



A University of Sussex DPhil thesis

Available online via Sussex Research Online:

<http://sro.sussex.ac.uk/>

This thesis is protected by copyright which belongs to the author.

This thesis cannot be reproduced or quoted extensively from without first obtaining permission in writing from the Author

The content must not be changed in any way or sold commercially in any format or medium without the formal permission of the Author

When referring to this work, full bibliographic details including the author, title, awarding institution and date of the thesis must be given

Please visit Sussex Research Online for more information and further details

The Application of Workflows to Digital Heritage Systems

by

Abdullah Al-Barakati

A thesis submitted in fulfilment of
the requirements for the degree of
Doctor of Philosophy
at the University of Sussex

Interactive Systems: Computer Graphics Centre
School of Informatics
University of Sussex
Brighton, England

February 2012

Declaration

The work presented in this thesis, unless otherwise indicated, was carried out in the Interactive Systems: Computer Graphics Centre at the University of Sussex and is that of author and has not been submitted in any form for any other degree at this or any other University.

Signed: _____

Abdullah Al-Barakati

Copyright © 2012

University of Sussex

Interactive Systems: Computer Graphics Centre
School of Informatics,
University of Sussex
Brighton
BN1 9QT

Acknowledgements

The work presented in this thesis would have not been possible without my supervisors: Dr Martin White and Dr Natalia Beloff. I would like to thank them both for their help, support and guidance over the course of my degree. I would like to thank my colleagues in the Computer Graphics Centre whose help was of the utmost importance to the research carried out.

I would also like to express my highest gratitude to my family members and friends who sincerely helped and supported me throughout my PhD studies, and gave me the drive to carry on and realize my dream of submitting this piece of work.

University of Sussex

ABSTRACT

Digital heritage systems usually handle a rich and varied mix of digital objects, accompanied by complex and intersecting workflows and processes. However, they usually lack effective workflow management within their components as evident in the lack of integrated solutions that include workflow components. There are a number of reasons for this limitation in workflow management utilization including some technical challenges, the unique nature of each digital resource and the challenges imposed by the environments and infrastructure in which such systems operate.

This thesis investigates the concept of utilizing Workflow Management Systems (WfMS) within Digital Library Systems, and more specifically in online Digital Heritage Resources. The research work conducted involved the design and development of a novel experimental WfMS to test the viability of effective workflow management on the complex processes that exist in digital library and heritage resources. This rarely studied area of interest is covered by analyzing evolving workflow management technologies and paradigms. The different operational and technological aspects of these systems are evaluated while focusing on the areas that traditional systems often fail to address.

A digital heritage resource was created to test a novel concept called DISPLAYS (Digital Library Services for Playing with Antiquity and Shared Heritage), which provides digital heritage content: creation, archival, exposition, presentation and interaction services for digital heritage collections. Based on DISPLAYS, a specific digital heritage resource was created to validate its concept and, more importantly, to act as a test bed to validate workflow management for digital heritage resources. This DISPLAYS type system implementation was called the Reanimating Cultural Heritage resource, for which three core components are the archival, retrieval and presentation components. To validate workflow management and its concepts, another limited version of these reanimating cultural heritage components was implemented within a workflow management host to test if the workflow technology is a viable choice for managing control and dataflow within a digital heritage system: this was successfully proved.

List of Publications

Al-Barakati, A.; Patoli, M. Z.; Gkion, M.; Zhang, W.; Newbury, P.; Beloff, N. and White, M. (2008): “A Dynamic Workflow Management Framework for Digital Heritage and Technology Enhanced Learning”. Proceedings of the 14th International Conference on Virtual Systems and Multimedia (VSMM 2008) dedicated to Digital Heritage, 20-26 October 2008.

Al-Barakati, A.; Zhang, W.; Patoli, M. Z.; Gkion, M.; Newbury, P. and White, M. (2009): “An Integrated Workflow Management Solution for Heritage Information Mashups” ASONAM 2009, First International Symposium on Mining Social Networks, held in Athens, Greece between 20-22 July 2009. The proceedings are published in IEEE explorer, ISBN: 978-0-7695-3689-7/09, DOI 10.1109/ASONAM.2009.40

Gkion, M.; Patoli, M. Z.; **Al-Barakati, A.;** Zhang, W.; Newbury, P. and White, M. (2009): “Collaborative 3D Digital Content Creation Exploiting a Grid Network”, Third International Conference on Information & Communication Technologies ICICT2009. The proceedings are published in IEEE explorer, ISBN: 978-1-4244-4609-4/09.

Patoli, M. Z.; **Al-Barakati, A.;** Gkion, M.; Zhang, W.; Newbury, P.; Beloff, N. and White, M. (2007): “A Service-Oriented Approach for a Digital Library System focused on Portable Antiquities and Shared Heritage”, Vast 2007, 26-29 November 2007, submitted to the 8th International Symposium on Virtual Reality, Archaeology and Cultural Heritage (2007), ISBN 978-963-8046-89-5 (pp. 15-18).

Patoli, M. Z.; Gkion, M.; **Al-Barakati, A.;** Zhang, W.; Newbury, P. and White, M. (2008): “How to Build an Open Source Render Farm based on Desktop Grid Computing”, presented in International Multi-topic Conference IMTIC, Published in CCIS-20, pp. 268-278, 2008.© Springer-Verlag Berlin Heidelberg 2008. ISBN: 978-3-540-89852-8.

Patoli, M. Z.; Gkion, M.; **Al-Barakati, A.;** Zhang, W.; Newbury, P. and White, M. (2009): “An Open Source Grid Based Render Farm for Blender 3D”, presented in IEEE Power Systems Conference & Exhibition (PSCE) 2009, held in Seattle, Washington, USA, 15-18 March 2009, published in IEEE, ISBN: 978-1-4244-3811-2/09.

Zhang, W.; Patoli, M. Z.; Gkion, M.; **Al-Barakati, A.;** Newbury, P. and White, M. (2009): “Reanimating Cultural Heritage through Service Orientation, Workflows, Social Networking and Mashups”, International Conference on CYBERWORLDS 2009, published in IEEE Computer Society, ISBN: 978-0-7695-3791-7/09, DOI: 10.1109/CW.2009.45

Poster Presentations

Al-Barakati, A.; Patoli, M. Z.; Gkion, M.; Zhang, W.; Newbury, P.; Beloff, N. and White, M. (2008): Poster presentation on “A Dynamic Workflow Management Framework for Digital Heritage and Technology Enhanced Learning”, presented in VSMM 2008 - Virtual Systems for MultiMedia dedicated to Digital Heritage, 24 October 2008, <http://www.vsmm2008.org/>

Gkion, M.; Zhang, W.; **Al-Barakati, A.**; Patoli, M. Z.; Newbury, P. and White, M. (2009): ‘*Mashing up*’ *Digital Worlds for Collective and Exploratory Learning*, CAL09 - Learning in Digital Worlds, Brighton, UK, 23-25 March 2009.

Patoli, M. Z.; **Al-Barakati, A.**; Gkion, M.; Zhang, W.; Newbury, P.; Beloff, N. and White, M. (2007): Poster presentation on “Digital Library Services for Portable Antiquity and Shared Heritage”, Informatics Open day, University of Sussex, 29 January 2007.

Workshop Proceedings

Patoli, M. Z., White, M.; Gkion, M.; Zhang, W. and **Al-Barakati, A.** (2009): “Touching the Untouchables”, presented at a workshop on Touching the Untouchables at University Exeter, UK, <http://www.exeter.ac.uk/scienceheritage/MartinWhite.html>, 29-30 May 2009

Glossary

ACMA	ARCO Content Management Application
AHRC	Art and Humanities Research Council
AMS	ARCO Metadata Element Set
ARCO	Augment Representation of Cultural Objects
ARIF	Augmented Reality Interface
BPEL	Business Process Execution Language
BPM	Business Process Management
BRICKS	Building Resources for Integrated Cultural Knowledge Services
CSS	Cascading Style Sheets
CMS	Content Management System
CMU	Carnegie Mellon University
DCA	Digital Content Archival
DCC	Digital Content Creation
DCE	Digital Content Exposition
DCI	Digital Content Interaction
DCP	Digital Content Presentation
DHR	Digital Heritage Resource
DISPLAYS	Digital Library Services for Playing with Antiquity and Shared Heritage
DL	Digital Library
DLI	Digital Library of India
DLMS	Digital Library Management System
DLS	Digital Library Systems
DRIVER	Digital Repositories Infrastructure Vision for European Research
DRM	Digital Rights Management
EDA	Event-Driven Architecture
ERIC	Education Resources Information Centre
HCI	Human-Computer-Interaction
HP	Hewlett-Packard
HTML	Hypertext Markup Language
ICT	Information and Communications Technology
IDLS	Integrated Digital Library System
IES	Institute of Education Sciences
IIITH	International Institute of Information Technology, Hyderabad

IISc	Indian Institute of Science
IS	Information Science
J2EE	Java 2 Platform Enterprise Edition
LAN	Local Area Network
LMS	Learning Management System
MIT	Massachusetts Institute of Technology
MVC	Model-View-Controller
NDAP	National Digital Archives Program
NDLP	National Digital Library Project
OAI-PMH	Open Archives Initiative Protocol for Metadata Harvesting
OPAC	Online Public Access Catalogs
PHP	Hypertext Preprocessor
RCH	Reanimating Cultural Heritage
RCMS	RCH Content Management System
SO	Service Oriented
SOA	Service Oriented Architecture
TDA	Taiwan Digital Archives
UI	User Interface
URL	Unfired Resource Locator
WAN	Wide Area Network
WF	Windows Workflow Foundation
WfM	Workflow Management
WfMC	Workflow Management Collation
WfMS	Workflow Management System
WfRM	Workflow Reference Model
XHTML	Extensible Hypertext Markup Language
XML	Extensible Markup Language
XSLT	Extensible Stylesheet Language Transformations
X-VRML	XML Virtual Reality Modeling Language
YAWL	Yet Another Workflow Language

Table of Contents

1	Introduction.....	1
1.1	Problem Statement	2
1.2	Research Questions	3
1.2.1	Digital Library Systems and Digital Heritage Resources	3
1.2.2	Workflow Management Systems	4
1.3	Proposed Solution	5
1.3.1	Digital Heritage Resource Workflow Management Development Model....	5
1.3.2	Workflow Management for DLSS.....	5
1.4	Contribution to Knowledge.....	5
1.4.1	Validating the DISPLAYS Framework	5
1.4.2	Identifying a WfMS Development Model for DLS	6
1.4.3	Design and Implementation of a WfMS for DLS	6
1.5	Organization of the Thesis	6
2	Digital Library and Workflow Management System Paradigms.....	10
2.1	Digital Library Systems (DLSS)	10
2.1.1	An introduction to Digital Library Systems.....	11
2.1.2	The Anatomy of a DLS	13
2.1.3	DLS Functionality and Components.....	14
2.1.4	Digital Library System Implementation Challenges.....	19
2.1.5	Digital Library Systems Design and Implementation Models.....	21
2.1.6	DLS Implementations	25
2.1.7	Comparison of DLSS.....	29
2.1.8	The Role of DLSS in Digital Heritage Preservation.....	30
2.2	Workflow Management Systems	32
2.2.1	What is a Workflow Management System?.....	33
2.2.2	WfMSs Functionality	34
2.2.3	A Brief History of WfMS	35
2.2.4	The Importance of Workflow Management Systems	36
2.2.5	Workflow Management Standards.....	37
2.2.6	Workflow System Examples.....	39
2.2.7	Workflow Management System Comparison	43
2.3	WfMS in DLS	45
2.4	Summary	46
3	DISPLAYS Framework.....	48
3.1	Introduction	48
3.2	DISPLAYS Concept	50
3.3	DISPLAYS Objectives	53
3.4	DISPLAYS Services	55
3.4.1	DISPLAYS Digital Content Creation (DCC) Services.....	56
3.4.2	DISPLAYS Digital Content Archival (DCA) Services	58
3.4.3	DISPLAYS Digital Content Exposition (DCE) Services	60
3.4.4	Digital Content Presentation Services (DCP)	62
3.4.5	Digital Content Interaction Services (DCI)	63
3.5	The DISPLAYS Architecture	64
3.5.1	DISPLAYS Grid Infrastructure	67
3.5.2	DISPLAYS Workflow Management	68
3.6	DISPLAYS Operational Scenario.....	71
3.6.1	Utilization of the DISPLAYS DCC and DCA Services	71

3.6.2	Utilization of the DCE, DCP and DCI Services	72
3.6.3	Empowerment of Content Sharing through DCA	73
3.6.4	User Generated Contents.....	73
3.7	Summary	73
4	RCH - an example DLS System	75
4.1	Introduction	76
4.2	RCH Context.....	77
4.3	RCH Objectives	79
4.4	The RCH Model.....	80
4.4.1	RCH Architecture.....	82
4.5	RCH Services	83
4.5.1	The Archival Services	83
4.5.2	The Presentation and Interaction Services	84
4.6	RCH Web 2.0 Mashups	85
4.6.1	RCH Mashups Implementation.....	86
4.7	Social Networking Integration	88
4.8	Heritage Workflow Management in RCH	89
4.9	Operational Scenario	91
4.9.1	Content Sharing and Distribution	91
4.9.2	Object Retrieval	91
4.9.3	Integration of Web 2.0 Mashups.....	91
4.9.4	Integration of Social Networking Functionality	92
4.10	Summary	92
5	RCH Archival and Presentation.....	94
5.1	Introduction	94
5.2	RCH MVC Model.....	96
5.3	RCH Archival Components	98
5.3.1	The Archival Tools	99
5.3.2	Data Mapping Tools.....	100
5.3.3	Design and Analysis of the Archival Mapping Tool	103
5.3.4	The Data Mapping Scenario.....	115
5.3.5	The Mapping Workflow.....	117
5.4	RCH Object Presentation	121
5.4.1	Design and Analysis of the Presentation Components	122
5.4.2	Dynamic Content Generation.....	127
5.5	RCH Object Retrieval Components	129
5.5.1	RCH Object Retrieval Design and Modelling	129
5.5.2	RCH XSLT Dynamics	135
5.5.3	The RCH Search Interface	137
5.6	Summary	140
6	Hosting RCH as a Workflow.....	143
6.1	Introduction	143
6.2	Solution Formulation	145
6.2.1	The Adopted Workflow Model.....	148
6.3	RCH WfMS Implementation	149
6.3.1	WfMS Implementation Technology	149
6.3.2	RCH WfMS Components	152
6.3.3	The RCH WfMS Design.....	153
6.4	The Workflow System Implementation	157
6.4.1	Workflow Runtime Construction.....	158

6.4.2	The Archival Components Workflow Runtime Services.....	158
6.4.3	The Retrieval Components Workflow Runtime Services.....	163
6.4.4	The Presentation Components Workflow Runtime Services.....	166
6.5	RCH Content Management System (RCMS)	170
6.5.1	RCMS Illustration	170
6.6	Workflow Running and Testing	177
6.6.1	Example Testing Scenario: the Object Mapping Process	178
6.6.2	Message Passing Verification	179
6.7	Results Analysis	181
6.8	Future Adaption of RCH WfMS	187
6.9	Summary	189
7	Conclusion and Future Work	192
7.1	Results Analysis	192
7.1.1	Validating the DISPLAYS Concept and WfMS Integration	193
7.1.2	DLS Hosting within a WfMS.....	195
7.1.3	Effective Communication between the RCH Components.....	196
7.1.4	WfMS Integration Gains and Advantages	196
7.2	Future Work	197
8	Bibliography	199
	Appendix A	211
	RCH New Interface.....	211
	Appendix B	217
	Museum Metadata Examples.....	217
	Appendix C	221
	RCMS Code Samples.....	221
	Workflow Management Code Samples.....	223
	Appendix D	224
	RCH Website Code Samples.....	224

List of Figures

Figure 2-1 Typical Structure of a DLS	14
Figure 2-2 The DELOS DLS Tiered Approach	24
Figure 2-3 Typical WfMS Components	35
Figure 2-4 The WfRM Categories	38
Figure 2-5 WF's Extensible Model	42
Figure 3-1 Examples of DISPLAYS Type Implementations	52
Figure 3-2 DISPLAYS Services and Their Associated Tools	56
Figure 3-3 DISPLAYS Digital Content Creation Services (DCC)	58
Figure 3-4 DISPLAYS Digital Content Archival (DCA) Services	60
Figure 3-5 DISPLAYS Digital Content Exposition (DCE) Services	61
Figure 3-6 Digital Content Presentation Services (DCP)	63
Figure 3-7 Digital Content Interaction Services (DCI)	64
Figure 3-8 DISPLAYS Architectural Components	66
Figure 3-9 Integration of a Grid Render Farm with DISPLAYS	68
Figure 3-10 Typical DISPLAYS Workflow-managed Scenario	71
Figure 4-1 RCH Overview	79
Figure 4-2 The RCH Data and Integration	82
Figure 4-3 RCH Architecture	83
Figure 4-4 RCH's Web Interface (presentation services)	85
Figure 4-5 RCH's Web 2.0 Mashup Scrapbook	87
Figure 4-6 Facebook Social Networking Integration with RCH	89
Figure 4-7 Abstraction of a RCH Type System	90
Figure 5-1 RCH Components	96
Figure 5-2 RCH MVC Implementation	97
Figure 5-3 A Snapshot of the Managed Digital Heritage Object Data	102
Figure 5-4 Mapping Tool Components	105
Figure 5-5 RCH Mapping Tool Use-Case Diagram	107
Figure 5-6 RCH Mapping Tool Class Diagram	111
Figure 5-7 Data Export Sequence Diagram	113
Figure 5-8 Data Import Sequence Diagram	114
Figure 5-9 Adding a New Schema Sequence Diagram	115
Figure 5-10 The Mapping Process	116
Figure 5-11 The Mapping Workflow	118
Figure 5-12 The Online Mapping Tool	119
Figure 5-13 RCH Frontend	122
Figure 5-14 Presentation Components Use-Case Diagram	124
Figure 5-15 The Adopted Presentation Model	126
Figure 5-16 RCH Presentation Components Sequence Diagram	127
Figure 5-17 Object Retrieval Use-Case Diagram	130
Figure 5-18 Object Retrieval Class Diagram	133
Figure 5-19 The Object Retrieval Sequence Diagram	135
Figure 5-20 The RCH Search Interface	137
Figure 5-21 The Search Results	138
Figure 5-22 Enlarged Object View	139
Figure 5-23 Detailed Object View	140
Figure 6-1 Hosting RCH Components in a WfMS	145
Figure 6-2 RCH Components' Interaction with the WfMS	148
Figure 6-3 Interaction between the RCH, RCH WfMS and RCMS	155
Figure 6-4 RCH Message Exchange Illustration	156

Figure 6-5 Archival Mapping Workflow Runtime	162
Figure 6-6 RCH Search Interface.....	164
Figure 6-7 Retrieval Workflow Runtime	165
Figure 6-8 Gallery View Presentation Example	168
Figure 6-9 Presentation Workflow Runtime	169
Figure 6-10 RCH Content Management System	171
Figure 6-11 The Archival Management Interface.....	173
Figure 6-12 The Object Retrieval Management Interface	174
Figure 6-13 The Object Presentation Management Interface	175
Figure 6-14 The Detailed Object Display Mode.....	176
Figure 6-15 The Object Presentation Frontend.....	177
Figure 6-16 Object Mapping Confirmation	178
Figure 6-17 The RCH Workflow Monitor	179
Figure 6-18 Passing the Source Museum Data from the Workflow Host.....	180
Figure 6-19 Passing the Destination Museum Data from the Workflow Host	180
Figure 6-20 Workflow Runtime Service Invocation.....	180
Figure 6-21 Using the Parameters within the Called Runtime Service	180
Figure 6-22 Workflow Integration Benefits.....	182
Figure 6-23 The latest RCH Search Interface	187
Figure 6-24 Enhanced Object Browsing View	188
Figure 6-25 Community-oriented Multimedia Feeds	189
Figure 6-26 Mapping the WfMS to the RCH Problems	191

List of Tables

Table 2-1 Comparison of Example Digital Library Systems.....	30
Table 2-2 WfMS Implementation Tools Comparison	44
Table 5-1 The RCH Archival Services Goals	99
Table 6-1 WF Features.....	150
Table 6-2 The Utilized WF Code Constructs.....	157
Table 6-3 The Presentation Component's workflow-managed Services.....	166
Table 6-4 RCH WfMS Testing Criteria and Results	183

CHAPTER I

1 Introduction

Advancements in software and hardware technologies (for example, online museum content management systems, desktop computing systems, Web 2.0, 3D modelling, etc.) transformed the ways that traditional libraries operate. Such advancements led to the advent of Digital Library Systems (DLS) and Digital Heritage Resources (DHR). Consequently, the functions of traditional libraries were further enhanced by means of giving them more interactive elements (Web 2.0 websites, social networking, mashups, touch screens, etc.) that enabled them to better achieve their goals and interact more closely with their users. This transformation was driven by a mix of software and hardware innovations in conjunction with a proliferation of digital data formats that facilitated the process of digital object preservation and distribution. As a result, a rich range of DLS implementations have emerged to serve different needs and purposes.

Technology advancements led to the advent of DHRs in the form of specialized DLSs that are dedicated to the purpose of ‘digital heritage object preservation and sharing’ [1]. This discipline is directly related to the concept of creating “Digital Library Cultural Heritage Resources” [2]. Digital Library Cultural Heritage Resources represent a subset of DLS implementations, which have their own unique characteristics and technical requirements in light of the complex digital objects that they have to handle (images, videos, 3D models, etc.), and the distributed environments (usually the Internet) in which they usually operate.

The proliferation of DLSs was consequently accompanied by parallel advancements in the standards and conceptual models that are associated with them. These models include the DELOS Digital Library Reference Model [3] among other standardization attempts that accompanied the rapid growth of DLSs. This proliferation allowed for the availability of different DLS and DHR implementations that are designed to meet the needs of their communities of practice, such as museums.

The role that advanced workflow management applications play in modern software systems can never be overestimated due to the profound ways in which they have

transformed a wide spectrum of software system implementations [4]. Workflow Management Systems (WfMS) are becoming an integral part of a variety of systems, especially those that possess convoluted business processes as well as data and control flows [5]. However, despite the rapid growth that WfMSs have witnessed over the last few years, it can be seen that their utilization in Digital Repositories and DLSs is still minimal. This limitation is due to a number of implementation challenges that limit the ways in which they can be integrated and utilized within such environments.

One of the main focus areas of this research is to investigate current DLS and WfMS implementations, while identifying the underlying gaps and shortcomings in DLSs to come up with an appropriate implementation model that can address such problems. This analysis is then followed with an investigation into how WfMS can be applied to make such DLSs more efficient in terms of workflow control, efficiency and resource management. The presented work is based on a methodological approach that combined the conducted literature review with a series of prototypes. These prototypes contributed to the validation of an innovative proposal for a new DLS concept called DISPLAYS (Digital Library Services for Playing with Antiquity and Shared Heritage) [1] [5] [6] [7] [8] [9]. More importantly, validating the concept of DISPLAYS is carried out in conjunction with a novel approach to workflow management for the control and management of DLS infrastructures based on a uniquely devised WfMS model.

1.1 Problem Statement

Despite the relative maturity that DLSs have reached [10], the issue of managing their workflows is one that still needs to be adequately addressed by effective WfMS implementation models. The integration of WfMSs within DLSs is an issue that has always been hallmarked by technical hurdles due to the inherent complexity and dynamicity associated with DLSs infrastructures [11]. Moreover, despite the fact that traditional WfMSs were originally based on some of the conceptual elements adopted in document management systems [12] cited in [13], they still fall short of the requirements for effective management of the convoluted workflows that usually exist in DLSs: these requirements are discussed further in Section 2.3. In essence, these traditional WfMS packages often fail to handle the demands of such complex environments, for example, adapting to dynamic process changes, simultaneous process running, etc. This shortcoming is caused by the range of runtime processes, business

processes, data and control flow, and the dynamic system and user events that any devised WfMS should adequately handle. These hurdles also led to the common trend that workflow management in DLSs is usually achieved through application-specific models, which are commonly inflexible and have limited customizability and scalability capabilities. Therefore, it is a challenging task to design and build WfMS models that are generic enough to meet the needs of different DLS implementations, while having the right level of flexibility to adapt to the dynamics of these implementations.

1.2 Research Questions

The research presented in this thesis is based on a number of research questions that form the basis on which the practical research activities were carried out. This spanned a number of technical areas in relation to the process of devising viable WfMS implementations for managing DLSs of various contexts as outlined below.

1.2.1 Digital Library Systems and Digital Heritage Resources

Online digital repositories, especially in the form of DLSs, are becoming more commonplace as they are being utilized to serve the purposes of digital content preservation and distribution. The proliferation of such systems was the result of the rapid developments that hardware and software systems have witnessed over the last two decades. These developments paved the way for DLSs that are more capable of meeting the goals and objectives of their user communities. This trend was forecast by Adam *et al.* [14], who predicted that DLSs would have the capability to impact on society in a number of areas related to the way society produces and shares digital content, leading to parallel developments in related disciplines. These disciplines include Digital Rights Management (DRM) and the laws and regulations that accompany content distribution and sharing. This is now a reality as can be observed in the advanced DLS implementations that can be seen virtually everywhere.

However, despite the developments that DLSs have witnessed, it can be observed that the use of WfMSs within DLSs is still limited and their utilization is minimal; this is due to a number of reasons, technical or practical. This is notably evident in the lack of literature that discusses such a combination, i.e. WfMSs integrated with DLSs, despite the fact that WfMSs have always accompanied the development of traditional document

management systems. Therefore, one of the important aspects of this thesis is to cover this overlooked research area. Fundamental questions concerning this research are:

- What are the major hurdles and technical challenges that prevent the full utilization of WfMSs within DLSs?
- What are the main considerations that WfMSs should address to successfully meet the demands of DLSs?
- What are the measures that can be used to assess the success of any WfMS in meeting the requirements of DLSs?

1.2.2 Workflow Management Systems

It is a notable trend that WfMS are becoming an integral part of a wide spectrum of enterprise systems due to the vital role that they play in enhancing, improving and managing task execution [4]. This notion is evident in the range of tools used to manage and track down the services and processes of the managed systems [4]. The area of automated workflow management is a rich research area involving a number of technical aspects that span a varied mix of design and implementation disciplines. The optimal utilization of WfMSs within existing software infrastructures (for example, document management systems, archiving systems, etc.) aims at better efficiency, speed and resource management and utilization. However, it is not as straightforward a task as it seems to build a WfMS; this is due to the fact that it has to address the needs and requirements of a complex mix of application and user needs. This fact highlights the importance of developing sound WfMS conceptual implementation models to suit a wide variety of enterprise applications that are in need of effective workflow management.

One of the focus areas of this thesis concentrates on devising a WfMS implementation model to suit the very nature of cultural heritage DLSs as will be highlighted below. This necessitated carrying out a comprehensive survey of current WfMSs and the current design and architectural paradigms that accompany them. As a result, the following research questions have been identified as a part of that process:

- What are the characteristics that a viable digital heritage resource WfMS implementation model should provide?

- What is the best approach for building generic WfMS solutions that can meet the needs of a wide range of DLS implementations?

1.3 Proposed Solution

This thesis presents an integrated workflow management solution for online digital resources that is constituted of two major parts as discussed below.

1.3.1 Digital Heritage Resource Workflow Management Development Model

One of the fundamental parts of the presented thesis is the design of a generic WfMS development model that is specifically designed to address the needs of DLSs. This model will adopt some of the concepts adopted in notable existing standards and paradigms such as the Workflow Management Collation (WfMC) [15], while complementing them with domain-specific features. The ultimate goal here will be the provision of a solid framework on which unique individual WfMS implementations can be based to meet the needs of a varied range of DLS implementations. The details of the adopted model of implementation can be viewed in Chapter 6.

1.3.2 Workflow Management for DLSs

A major portion of the work presented in this thesis is based on the implementation of a novel WfMS for the purpose of managing the convoluted and intersecting workflows of an innovative digital heritage resource for reanimating cultural heritage. The conducted work is based on the creation of a number of prototypes to examine a conceptual DLS called DISPLAYS (Digital Library Services for Playing with Antiquity and Shared Heritage). DISPLAYS offers a number of digital library services including digital heritage object content creation, archival, exposition, presentation and interaction services.

1.4 Contribution to Knowledge

The research and practical work conducted as a part of this thesis contributed to knowledge in a number of areas as outlined below.

1.4.1 Validating the DISPLAYS Framework

The author of this thesis contributed to validating the DISPLAYS Framework [5] [7] [1] by building archival, retrieval and presentation components for a DISPLAYS based

DLS. These components were built from scratch as a part of the Reanimating Cultural Heritage (RCH) system, which is a DHR shared between three museums: the British Museum, Brighton Museum and Art Gallery, and the Glasgow Museum [1]. The archival component comprised a data mapping tool designed to facilitate mapping the digital objects' metadata (XML based) from one museum to another. XSLT files played a major role in the retrieval component as they were utilized to extract search results from the archival XML files. The search results are presented to the end-user through the XSLT-rendered HTML code. Details of this contribution can be found in Chapter 5. The built DLS components were then used as the test bed to validate the proposed WfMS as detailed in Chapter 6. Contributions can also be found in [1] [5] [6] [7] [8] [9].

1.4.2 Identifying a WfMS Development Model for DLS

One of the major aims of this thesis is to develop a valid implementation model to accommodate the workflow management needs of DLS implementations. The developed model is based on the idea of developing a modular WfMS that is able to fit within the SO RCH components (see Chapter 4). This WfMS model closely interacts through message passing with two entities: the DLS (RCH) itself and a host application that provides UI elements to the end-users. The proposed WfMS model was designed and implemented as outlined in Chapter 6. Contributions in this regard can be found in [5] [6].

1.4.3 Design and Implementation of a WfMS for DLS

A novel WfMS was built (see Chapter 6) to manage the components of the RCH system, which acted as an example DLS (see Chapter 4). The archival, retrieval and presentation components of RCH were managed through the WfMS that was hosted in a host application that interacted with the end-users. The host application was called the RCH Content Management System (RCMS) and represented the medium through which the users utilized the system's workflow-managed components. Contributions to this work can be found in [1] [6] [5] [7].

1.5 Organization of the Thesis

Chapter 2 of this thesis is divided into two main sections that cover the literature related to the most fundamental aspects of the conducted research and the proposed WfMS

implementation. The first section outlines some of the core concepts that are related to DHRs, DLSs and Digital Library Management Systems (DLMSs). In this regard, the history of libraries in general is briefly outlined as a preamble for the topics of DLSs and distributed DHRs. Special emphasis is placed on the gradual improvements that modern technology has introduced to traditional libraries.

The above discussion is further expanded by covering the technical aspects that are associated with DLSs while focusing on their implementation models, as well as the standardization initiatives that contributed profoundly to shaping the current DLSs landscape. Moreover, the technical challenges that such systems should overcome to arrive at fully functional models are highlighted and scrutinized by means of examining some of the most prominent literature in that regard.

The above conceptual and technical overview is further enhanced by showcasing some of the most notable DLS implementations while focusing on their different technical and operational aspects, characteristics, issues, and the paradigms that hallmark each of them. The first section of Chapter 2 concludes by covering a very important aspect of DLSs, which is the utilization of DLSs for the purposes of Digital Heritage Preservation and Digital Curation.

While the first part of Chapter 2 focuses on DLSs as they form the medium on which the proposed workflow management solution is to be implemented, the second part focuses on the literature and previous work done in relation to WfMS implementations. This is primarily concerned with the technical and operational aspects of such convoluted systems while covering their early implementation in the form of office automation systems leading to the latest state-of-the-art in this area. In a similar approach to the one adopted in the first section, the standards and conceptual models that govern WfMS are also discussed while emphasizing the technical and practical impacts that they impose on WfMS implementations in different contexts. This section then embarks upon highlighting some of the most prominent open and closed-source WfMSs while outlining their key features within their respective implementation contexts. Finally, Chapter 2 concludes by discussing the issues related to the utilization of WfMSs within DLSs, while focusing on the current implementations and the challenges that that accompany them.

Chapter 3 discusses DISPLAYS [7] [8] [9] [16] which is a DLS framework whose functionality revolves around the concept of digital heritage resource sharing and distribution. Chapter 3 highlights the most important aspects of the DISPLAYS framework, which are its digital heritage content: Creation, Archival, Exposition, Presentation and Interaction Services for digital heritage collections.

Chapter 4 of this thesis covers a specific digital repository implementation that is based on the model provided by the DISPLAYS framework. In view of this, the Reanimating Cultural Heritage (RCH) Project is discussed as it presents an innovative digital heritage resource that formed a useful model on which the proposed WfMS can be implemented and tested. The synergy between the DISPLAYS framework and the RCH system is highlighted, especially in relation to the Service-Oriented (SO) system services and the techniques they adopt within the devised loosely coupled architectural model. The unique aspects of the RCH system are covered especially in relation to the tools it provides to reanimate cultural heritage objects by means of utilizing different media and technologies such as videos, 3D Models, social networking and Web 2.0 mashups.

Chapter 5 elaborates on the RCH Archival, Retrieval and Presentation Components due to the pivotal role that they play and the workflow management implications that they impose. Chapter 5 focuses particularly on the technical details of those components and the services that they provide. These components acted as the test bed for the RCH WfMS prototype as detailed in Chapter 6.

Based on the three chapters described above, the DISPLAYS framework acted as a DLS concept, RCH acted as validation architecture to better refine the WfMS prototype, and the Archival, Retrieval and Presentation components were used for the actual implementation, validation and testing of the developed WfMS.

Chapter 6 is dedicated entirely to discussing the underlying details of the proposed WfMS in relation to the systems elaborated in the three previous chapters. This begins with exploring the core concepts and ideas behind the proposed system, and the adopted paradigm that was devised to suit the demands of the targeted DLS. Moreover, a number of advanced workflow implementation concepts are discussed and examined within the context of the proposed WfMS. These concepts include workflow runtime services, workflow design considerations, workflow technology utilization and

workflow integration within an existing software infrastructure which is in this case is the RCH system.

Chapter 7 presents the conclusions and future work in regard to the work carried out as a part of this thesis. An evaluation of the conducted research is presented in relation to this thesis's research questions and its goals. The WfMS implementation is also analyzed and possible improvements and enhancements are proposed as a part of the intended future work.

CHAPTER II

2 Digital Library and Workflow Management System Paradigms

This chapter explores two main topics that are related to the proposed novel DLS and WfMS systems presented in this thesis. The reviewed work includes some positioning and peer reviewed papers that led to this thesis. The author's contributions that were reviewed include the work presented in [1] [5] [6] [7] [8] [9]. Prominent literature is also reviewed in relation to the main themes of the thesis. This chapter discusses DLSs and the different technical and practical aspects related to their design, implementation and operation. The focus here is to cover the history of DLSs while paying special attention to their implementation paradigms.

The chapter then explores WfMSs in general, especially in relation to their history, technologies and notable system implementations. Moreover, the benefits of WfMS implementations are explored while focusing on the latest and most innovative technologies to implement such systems. Furthermore, emphasis is placed on their application in distributed software systems and the implementation challenges that they usually have to overcome. Workflow management standardization efforts and initiatives are also discussed due to the pivotal role that they play in association with the process of designing and implementing modern WfMSs. This chapter concludes with a discussion of the implementation of WfMSs within DLSs.

2.1 Digital Library Systems (DLSs)

Libraries are, and always have been, among the most prominent sources of information for societies. Furthermore, libraries are traditionally considered to be one of the most important sources on which communities heavily depend to access information of all sorts and types. As McGrory *et al* state [17], “libraries have always been a community’s ‘portal’ to information, knowledge and leisure. Beyond their shelves, libraries are a community’s gateway to information from many sources nationally and internationally”. Moreover, libraries are rather meaningfully described by Byrne [18] to

be “storehouses of knowledge”; a description that highlights the pivotal role that libraries play in modern societies, where human heritage data is preserved over the years to be accessed by knowledge seekers, researchers and learners to serve their respective goals.

Libraries in general play the imperative role of protecting any society’s heritage by preserving its data in a retrievable highly accessible format. Libraries also offer (by the knowledge that they make available) individuals and societies alike the opportunity to improve their lives and contribute to the development and wellbeing of their respective communities [18]. Murray *et al.*, in their book *The Library: An Illustrated History* [19], argue that libraries play the profound role of recording the human cultural achievements being preserved from previous generations; this is a worthy goal that contributed to the rapid advancements that hallmarked the existence of libraries over the years. Therefore, the importance of libraries can never be overestimated, especially in regard to spreading knowledge and preserving human heritage across generations.

From an historical perspective, libraries span a long and rich history that accompanied almost every step of human development as “the collection of written knowledge in some sort of repository is a practice as old as civilization itself”, Krasner-Khait [20]. Libraries can be traced as far back as 1300-1200 BC when the Ancient Egyptians used to hold collections of papyrus scrolls to preserve records related to the different aspects of their civilization [20]. More developed forms of libraries and document collections date back about 4000 years, represented in the collections of clay tablets that were used to preserve the heritage and culture of the highly developed society of ancient Mesopotamia [21]. Libraries evolved over time and their improvement was fuelled by the rapid proliferation of the mediums used to hold information (books, audio tapes, CDs, etc.), and the improvements that libraries witnessed in a parallel fashion, as will be further showcased in this chapter.

2.1.1 An introduction to Digital Library Systems

The advancements that information technology has brought to modern societies paved the way for traditional libraries to evolve into more advanced and interactive mediums while, at the same time, being able to better serve their goals and objectives [21]. Such goals include better information access, better search and retrieval operations, and more interaction between users and the managed collections. This revolution was driven

mainly by the sheer mix of digital data formats and types that technology advancements made available for the purposes of preserving data objects whether they are textual, visual, animated or audible.

This notion of better interaction and access to digital libraries was further enhanced by other key technological innovations (advanced data storage media, distributed systems, networking technologies and web technologies) that allowed the development of more complex models of libraries. These advancements in software technologies were coupled with equally powerful hardware innovations (desktop computing, servers, LANs, WANs, etc.), which opened new horizons for better and more advanced forms of library systems. Such developments allowed for better digitization and reanimation processes for library collections within their respective contexts. This also allowed for better availability and accessibility opportunities for knowledge seekers through the utilization of carefully devised DLS services.

2.1.1.1 DLS Definitions

According to Abu Bakar [22] the term “Digital Library System” means different things to different scholars and researchers. This is partially attributed to the fact that “the proliferation of digital libraries over the past decade has produced so varied an array of digital collections and services that the term digital library defies a precise definition”, Stefanelli and Aldrich [23]. Moreover, DLS implementations mainly represent a meeting point [24] between a number of computing and library disciplines. For example, data management, information retrieval, library sciences, document management, information systems, the web, image processing, artificial intelligence, human-computer interaction, and digital curation [24].

Therefore, such an inherent multidisciplinary nature is a factor that manifested itself in a variety of definitions that can be applied to DLSs. For example, according to Pavlova-Draganova and Paneva [25] a DLS is a managed collection of information associated with a number of services, whereby the stored information is in digital format while being accessible online through a distributed networked computing system. On the other hand, Lesk [26] cited in [22], has defined a DLS as “organized collections of digital information. They combine the structure and gathering of information, which libraries and archives have always done, with the digital representation that computers have made possible”. Furthermore, Janssen [27] considers the World Wide Web to be a DLS

of a generic form due to the digital data sharing and distribution possibilities that it made available to its users. This definition is based on the opportunities that the presence of the Internet has made available to computing systems, paving the way for supporting the idea that digital collections can be made available to a variety of users over a common distributed platform, as argued by Greenaway [28]. A definition that seems to capture the technical and functional aspects of a DLS is provided by Janssen [27] as he states that a DLS is considered to be “an integrated set of software, hardware, and protocol elements that together provide a means of storing, managing and accessing documents in digital form”.

Despite the varied set of definitions for what constitutes a DLS, all point to a common goal. This goal is presented in knowledge preservation and delivery in various digital formats to the members of any given community, whether it is an organization, a cultural institution or a nationwide body. In view of this, it seems that DLSs have profoundly transformed the ways in which people share, preserve and view knowledge. This was foreseen by Adam *et al.* [14] who predicted that DLSs would impact societies, necessitating changes to laws and regulations while effectively benefiting a number of areas such as economics, intellectual property and medicine. Interestingly enough, 14 years after that forecast, it can be seen that DLSs are becoming commonplace and a prominent popular medium for information distribution and sharing, thus realizing the above vision. This was inspired by the rapid growth of technological advancements that hallmarked every step of the gradual evolvement of DLSs.

2.1.2 The Anatomy of a DLS

DLSs have undergone a number of important and notable developments over the years, and they are now considered to be one of the most powerful means for enabling public access to information [14]. They have also reached a good level of maturity in terms of their development paradigms and the ways in which they can be devised to meet the demands of any given development environment [24].

Figure 2-1 is an illustration of the core components that may constitute a typical DLS. It can be seen that an integrated combination of data and software services contribute to the delivery of the overall DLS functionality. Such services include archival and indexing services, data retrieval services, presentation services, etc. It can also be seen that effective system functionality is achieved through message passing between the

system's components, which tend to be complex and intersecting in large enterprise implementations. Custom-made data models and user interfaces play a key role in such implementations as they contribute to achieving the rich functionality that DLSs deliver to their user communities. Supported digital objects may include images, videos, audio, text and 3D objects. Some example DLSs will be discussed in Section 2.1.6 of this chapter.

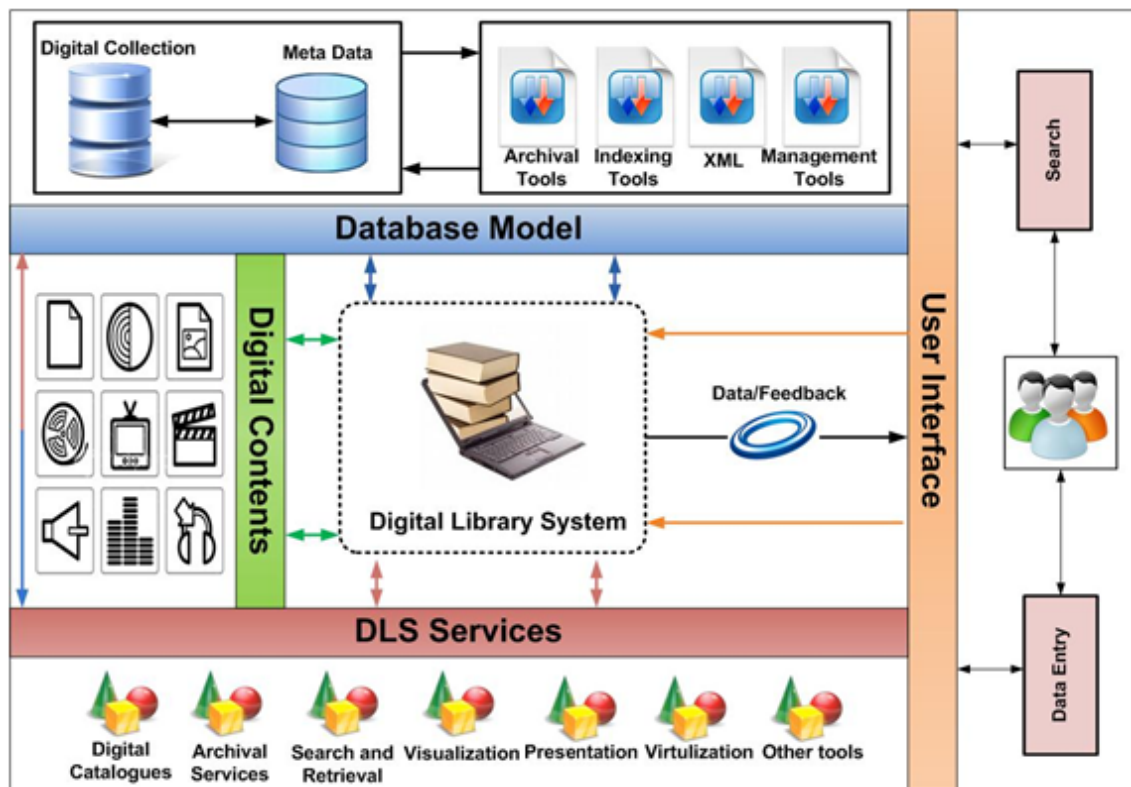


Figure 2-1 Typical Structure of a DLS

2.1.3 DLS Functionality and Components

A DLS can be thought of as a combination of services and contents that complement each other [29]. Despite the various DLS implementations that can be seen everywhere, they virtually cover a common set of core functionality based on a range of different interactive services. There are certain user expectations in relation to the functionality that a DLS should effectively cover; this is consequently reflected in the set of services seen in most DLSs, such as customized user interfaces and advanced archival, search and retrieval capabilities.

McGrory *et al.* [17] indicate that a DLS is always expected to serve the particular needs of its users by employing the right mix of services complemented with the support of the user's preferred technological mediums for accessing information. Such access

methods can be in the form of a computer monitor, a touch screen, a screen reader or a Braille keyboard for instance. Therefore, there is a strong user-centred focus in most modern DLS implementations.

On the other hand, there is always a certain level of expectation in regard to the protection of the users' privacy in connection with their personal information and the need to make them totally private and secure when using a DLS [17]. The significance of security and privacy is enhanced by the fact that the vast majority of DLSs are networked systems that exist in online environments. Such implementations impose a certain set of security and privacy concerns in regard to the user actions and the protection of personal data circulated throughout the system.

2.1.3.1 DLS Functionality

DLSs share a number of common features driven from the purposes that they usually serve. The first of those common-ground features is represented in the fact that a DLS is typically expected to provide its users with instantaneous access to information. According to Adam *et al* [14], it is always expected that a DLS should provide instantaneous access to information that totally replaces the need for a knowledge seeker to go physically to a library to retrieve the needed pieces of information. Hence, such a goal is emphasized in most of the notable DLS implementations in place; for example, the Goleman Library DLS has the stated prime goal to “get library materials into the hands of users”, Schermerhorn [30]. This is one of the most important goals that most of DLSs serve and relate to in one way or another. This goal greatly contributes to the final user perception in regard to the effectiveness and convenience of the DLS in use, as information access is one the factors used to gauge the success of any given DLS implementation.

Another distinctive feature that sets DLSs apart from conventional libraries is the unlimited possibilities they offer in terms of the ranges and formats of data they can support and deliver to their users. Technology advancements in relation to data processing, modelling, visualization, virtualization and presentation made such a feature commonplace in most DLSs. Therefore, it is possible to present the stored data in a variety of formats that suit different user needs and software/hardware capabilities in place, making DLSs one of the most convenient mediums for data preservation and retrieval. In view of that, a DLS may typically support a comprehensive and varied set

of digital data formats including text, audio, video, 3D models, and other multimedia objects. Hence, DLSs go beyond traditional text-based processing [31] to an almost unlimited range of supported formats that allow for comprehensive and highly accessible digital data repositories [32].

The Internet can be thought of as one vast DLS accessed by computers with its multitude of interfaces (e.g. media player, PDF viewer ... web browser) for accessing and presenting the Internet's information. Another example is the National Digital Library Project (NDLP) at the Library of Congress, which utilizes a combination of digitized text, audio and video to present the objects in its collections [33]. Such digital data objects are typically associated with the complementary data (metadata) necessary for the functionality of the archival and retrieval components leading to the advent of highly sustainable digital collections [32].

In terms of the sources of the contents that constitute the collections of a DLS, they can be either materials that have been digitized from their original formats, such as books, or alternatively they can span a wide spectrum of contents that were originally produced in digital format. Interestingly enough, Sreejaya and Sreekumar in their paper "Digital Library Initiatives and Issues In India: Efforts On Scholarly Knowledge Management" [34] have stated that more than 70% of the world's scholarly literature is born digital, which naturally necessitates the utilization of DLS to acquire and manage such objects, adding more importance to the role that DLSs play today.

A typical DLS services are usually implemented within an appropriate implementation model or framework where all the system services or modules are hosted and managed. It is therefore always a desirable characteristic in DLSs to adopt some sort of functional adaptively to meet the user needs [35]. Consequently, there are a number of different DLS implementations that range from highly distributable customizable packages to specifically-built versions to suit the underlying needs of the operational environment in place, some example models will be presented in Section 2.1.6.

2.1.3.2 The User-Centric Nature of DLSs

It can be noted that the user-centred nature of DLSs has manifested itself in the expectation that their users should be provided with certain levels of help and support to be able to fully utilize the provided services. This perception is reflected in the design of most of the notable DLS implementations as the concept of usability, help facilities and

user support are always reflected in the efforts to “design these digital libraries for effective use by different types of users”, Xie [36]. Ideally, this user-centred nature should be empowered with user-friendly interfaces that would make the process of utilizing the services of a DLS straightforward.

User interaction with DLSs is a prominent research area in modern DLS literature, as can be observed in the notable number of research and experimental ventures that covered that subject [24]. This discipline spans a number of interrelated areas of interest including “methodological, conceptual and theoretical support in some areas, such as Human-Computer-Interaction (HCI), for the usability studies, and Information Science (IS), for the studies about information needs and user’s behaviour during the information search and use processes”, Ferreira and Pithan [37].

In the paper entitled “Usability of digital libraries: a study based on the areas of information science and human-computer-interaction”, Ferreira and Pithan [37] have analyzed the usability of DLSs by means of cognitive usability evaluation while observing the search and retrieval aspects of user interaction. This process led to the conclusion that user satisfaction is largely driven by the level of support and ease of use with which a DLS provides its users. On the other hand, the importance of DLS usability issues are highlighted by the fact that Yu [38] attributed the lack of full utilization of DLSs in China to a group of usability issues that undermined a number of prominent local implementations. This finding gives a clear indication that it is a key success element to “understand users’ difficulties in working with information and particularly with DLs, and to equip developers with ways of thinking about users and their needs that help guide DL development and evaluation”, Yu [38].

2.1.3.3 DLSs Search and Retrieval Capabilities

Another distinctive feature of a DLS implementation is the existence of “Agents for Search and Selection”, Adam *et al.* [14], in other words search and retrieval tools and mechanisms. DLSs tend to support a variety of search and retrieval tools that suit the range of stored digital objects and the techniques used to archive and index them. Ideally, the provided search and retrieval capabilities should be “convenient and efficient”, Yu [38] to exactly meet the information retrieval needs of the end-users.

Search and retrieval is, and always has been, one of the major topics discussed in DLS literature. In view of this, Schatz [39] argues that the development of mechanisms for

information retrieval has been among the important elements that accompanied the historical development of DLSs. Therefore, efficient search and retrieval capabilities are considered to be integral parts of DLS implementation models as “the primary purpose of digital libraries is to enable searching of electronic collections distributed across networks, rather than merely creating electronic repositories from digitized physical materials”, Schatz [39].

Devising search and retrieval components often tends to be a complex process in modern DLS implementations due to the sheer variety of the digital objects being handled by a DLS’s components. Therefore, it is not a simple matter of text-based search operations as some collections require a specially-devised search and retrieval code infrastructure. For example, audio material requires the utilization of complex tune retrieval mechanisms to retrieve them in association with an appropriate user interface that enables users to adequately review the retrieved objects [40].

2.1.3.4 DLSs Archival Capabilities

The search and retrieval capabilities of a DLS implementation are just one face of the coin; the other face is the archival tools which manage the ways by which the DLS’s collection data is being stored and organized. Having effective archival capabilities is a pivotal feature in DLSs as “flexible organization of information is one of the key design challenges in any digital library”, Arms *et al* [41]. Furthermore, as digital collections form the data foundation on which a DLS operates [42], archival components do not only lead to the accumulation of data objects, they also contribute to the process of enhancing and maintaining the library’s contents by effective and meaningful indexing of metadata for example. This is directly related to the baseline concept of “digital preservation” Eakin [43], which all the other DLS functional areas revolve around. The content archiving process is further complicated by the necessity of managing the information that usually accompanies the stored data objects, such as the metadata which is used for the later search and retrieval operations.

It can be observed that DLSs employ different techniques to store and archive their contents while depending on a data model that is designed to suit the range and magnitude of the digital data objects being handled. A very important distinction should be made here between a conventional database and the data model of a DLS as “a

database contains representations of facts and of the involved objects”, Spyratos [44], whereas a DLS handles the actual objects and not a mere representation of them.

2.1.3.5 Other DLS Functionality

It is often the case that the features typically found in a DLS span a wide range of supportive functionality such as personalized user interfaces, content management tools, collaboration tools, and advanced augmentation and innovative mashup tools [32]. Such tools may include, for example, the innovative integration of information mashups and social networking tools [1].

2.1.4 Digital Library System Implementation Challenges

As outlined in Section 2.1.3, there is always a set of core services and functionality that a DLS is expected to provide its users with. However, the development of a DLS is not as a straightforward a task as it seems because such a development necessities overcoming a number of technical and functional challenges. In view of this, the fact that, on the one hand, a typical DLS has to handle diverse and highly distributed sources of digital contents and, on the other hand, a vibrant range of user communities is a challenge in its own right [45]. This is a key area of DLS competency as indicated by Greenstein [45] who stated that “digital libraries establish their distinctive identities, serve their user communities, emphasize their owned collections, and promote their unique institutional objectives by the way in which they disclose, provide access to, and support the use of their increasingly virtual collections.” Therefore, it is not only important to hold rich and large digital object collections, it is equally important to have the right tools to manage them and deliver the required contents to the end-user in the required format, within the targeted software platforms.

Fox *et al.* [46] argue that DLSs should be able to address a number of technical hurdles that directly stem from the need to handle technological disparities. These disparities are represented in the various systems that need to interact with a DLS and the different platforms that users tend to use. DLSs should also be able to deal efficiently with the changing paradigms in network architectures and protocols (Web 2.0, SOA, SOAP, etc.) with which they interact heavily. Furthermore, one aspect of the challenges that face DLS implementations was discussed by the work carried out by Hopkinson in his paper “Challenges for the Digital Libraries and Standards to Solve them” [47], in which

he argues that adopting a high level of digital content standardization would contribute to solving the problems of data mapping and distribution within a DLS environment.

While the capability of having efficient search and retrieval components is described as one of the most important features of a DLS as mentioned in Sections 2.1.3.3, it is also considered to be one of the challenging implementation areas that impose certain functional hurdles in the face of DLS developers. Furthermore, Hopkinson [47] states that “searching systems is one of the most difficult operations to achieve satisfactorily”. This complication can be attributed partially to the rich and varied set of digital data objects and formats that a DLS usually holds in its collection and the comprehensive set of data that accompanies them. Moreover, Hopkinson [47] further indicates that the search and retrieval of digital objects within the collections of a DLS has always been hallmarked with imprecision; an issue that DLSs should overcome to arrive at search and retrieval modules that are capable of delivering accurate search results to their end-users. A good example of complex search and retrieval operations within a DLS is represented in the work carried out as a part of the Reanimating Cultural Heritage (RCH) project [1]. This work is discussed in more detail in Chapters 4, 5 and 6.

Another challenging area that DLS literature covers is the issue of applying workflow management to the intersecting components of a DLS system as identified by McCord [48]. Based on that view, workflow management for a DLS can be a challenging task for a number of reasons. These reasons include the set of complex and intersecting system workflows, the multiple ranges of system stakeholders across a distributed system, or a semantic network and the inherent complexity and magnitude of the data objects being transmitted throughout the system. This aspect is discussed in more detail in Section 2.3 of this chapter.

Another area of challenge that prominent DLS literature covers are the issues resulting from the networked nature of such systems as “the planning and implementation of networked digital libraries poses new challenges and involves policy making regarding the members, content, content management, governance, maintenance and the technical know-how”, Fox *et al* [46]. Such issues involve technical aspects such as the case with service distribution and maintenance or organizational ones such as Digital Rights Management (DRM).

Another implementation challenge is the process of managing the access of the DLS’s collections by the different user groups that might be accessing the DLS. Therefore, the

complex user rights and security issues that might arise in such a scenario should also be taken into account [45]. This challenge necessitates the provision of appropriate administrative tools to enable the effective management of the different aspects of user access to arrive at fully functional secure DLS implementations.

2.1.5 Digital Library Systems Design and Implementation Models

There are a number of common issues that DLS models often aim to address including content management, content publishing, search and retrieval and content interpretation [49]. Such areas have always proved to be challenging ones due to their complexity, as well as their inherently intersecting processes. Moreover, it is often the case that DLSs are implemented on the foundation of a well-defined conceptual model that is based on a certain architectural design paradigm; this is evident in the many initiatives to develop such models, standards and frameworks, which are further discussed below.

A conceptual model of a DLS implementation can be thought of as a combination of the contents provided as well as a set of associated services and management tools that are hosted in an appropriate operational host [29]. A DLS model can be implemented based on a number of architectural design patterns and paradigms depending on the size and nature of the system and the functionality that it needs to fulfil to meet its goals. Therefore, the underlying details of a DLS conceptual model may vary from one system to another. This is because different DLS models adopt different approaches in devising their underlying components due to the unique nature of each system.

The WfMS prototype presented as a part of this thesis uses a DLS framework as a test bed for its implementation. This framework is called DISPLAYS (**D**igital library **S**ervices for **P**LAYing with Antiquity and **S**hared Heritage) which is described in detail in Chapter 3. The actual implementation of the proposed WfMS was integrated with a DISPLAYS-based DLS implementation called Reanimating Cultural Heritage (RCH), which was constructed to validate the proposed workflow management concepts and implementation approach. RCH components are detailed in Chapters 4 and 5.

2.1.5.1 Library 2.0

Library 2.0 is a relatively new term in the literature covering DLS, and is a paradigm that is fuelled by the latest advances that Internet technologies have recently been witnessing. Savastinuk and Casey [50], cited in [51], see Library 2.0 as a paradigm that

is centred on the concept of “user-centred change” [50]. Therefore, Library 2.0 is concerned with the provision of advanced and more interactive DLSs where user participation is empathized while providing a set of services that make such participation possible. It is useful to put this paradigm in context when discussing modern DLS literature as it represents one of the elements that encapsulates the latest thinking in this arena.

Library 2.0 elements are showcased in a number of modern DLS implementations that incorporate social networking functionality among other highly interactive features. Library 2.0 implementations have the tendency to treat DLSs as web applications in their own right, based on the fact that they all have the common feature of operating on a networked system which in most cases is the Internet. Hence, the concept of Library 2.0 represents the idea of combining the latest advances of Web 2.0 [52] with DLS services, resulting in highly interactive DLS implementations that go a step beyond the functionality provided by traditional implementations. Some of the distinct features that Library 2.0 implementations provide their users with include virtual references, different ranges of personalized public online access catalogue interfaces, and a variety of downloadable material that can be used and manipulated in different ways [50].

Combining the features of Library 2.0 in conjunction with DLS is largely considered to be a move beyond the static nature that hallmarked early implementations of DLSs. Early DLS implementations mainly depended on the older web infrastructure that provided limited content and user interaction capabilities [53]. An example of such a static trend is provided by Maness in his paper “Library 2.0 Theory: Web 2.0 and Its Implications for Libraries”, in which he indicates that the online public access catalogs (OPACs) represent a DLS implementation that lacks the range of interactive services that Library 2.0 has enabled. For example, OPACs require its users to carry out traditional search and retrieval processes without providing the kind of support that would normally exist in Web 2.0 implementations such as search suggestions, preferred search saving, and so on. On the other hand, good examples of Library 2.0 implementations include the Digital Library of India (see Section 2.1.6.1) and ARCO (see Section 2.1.6.6).

2.1.5.2 *DELOS Digital Library Reference Model*

A good example of a conceptual model that is generic enough to be applied to a wide range of DLS implementations is the DELOS Digital Library Reference Model [3], developed as a part of the Seventh Framework Programme, ICT Programme – “Cultural Heritage and Technology Enhanced Learning” adopted by the EU. In that rich DLS model, a group of DLS SO independent components are encapsulated into a generic framework that supports a number of different intersecting modules that can be utilized to act as the functional baseline of different types of DLS implementations.

The DELOS model itself is encapsulated in what is called the “DELOS Digital Library Reference Model” [3], which forms the technical basis of the DELOS project. The prime objective of the DELOS project is to “define and conduct a joint program of activities in order to integrate and coordinate the on-going research activities of the major European research teams in the field of digital libraries for the purpose of developing the next generation digital library technologies” [29]. Therefore, DELOS provides a set of highly standardized tools and design patterns that serve the needs of the participating communities of practice. Such tools are encapsulated within a conceptual model that includes common user interfaces, API interfaces, DLS management tools, etc. [29]. The above model is better known as the Digital Library Manifesto (DL Manifesto) and, as the name suggests, it formulates an elaborated effort to be presented as a “springboard” for the development of DLSs [54].

The DELOS Reference Model builds on the paradigm of a tiered DLS functional model, where a three-tier framework comprises three main DLS services that are related to three different core functional areas: these are Digital Library, Digital Library System, and Digital Library Management System [3]. Such a tiered approach represents “different levels of conceptualization of the universe of Digital Libraries”, Candela *et al* [3], leading to high levels of flexibility and interoperability between the DLSs that adopt it because of its modular nature.

Figure 2-2 illustrates the DELOS architectural approach where the interaction between the three DLS tiers leads to the realization of the system’s functionality while adhering to a flexible loosely-coupled implementation model. The Digital Library Tier represents the actual organization’s or community’s DLS implementation that manages a collection of digital objects. These objects may span a variety of formats including text,

images, video clips, audio clips, etc. The Digital Library Tier represents the interface through which the users interact with the provided DLS services. Finally, the Digital Library Management System (DLMS) Tier represents the most complex tier within the DELOS conceptual model [3]. This is due to the fact that the DLMS Tier provides the core system software components including its archival, search and retrieval, presentation, security, etc., components.

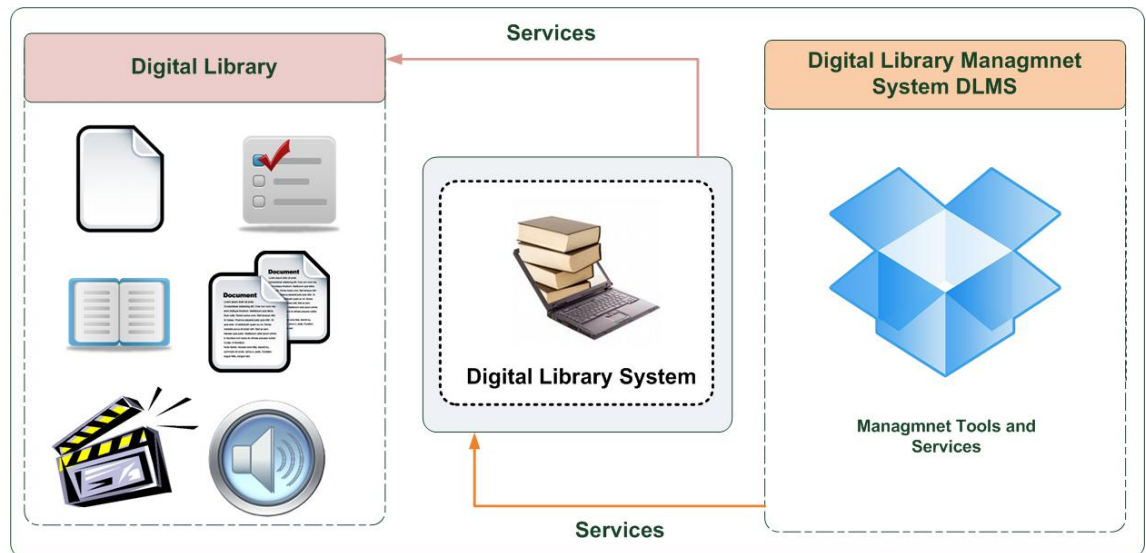


Figure 2-2 The DELOS DLS Tiered Approach

2.1.5.3 Ontology-Based DLS Framework

Another prominent DLS conceptual model can be seen in the framework that was proposed by Motta *et al.* [49], which is an Ontology-Based model that illustrates the idea of a limited-scale framework that targets a specific area of DLS functionality. The Ontology-Based Digital Library Server for Research Documents and Discourse [49] was based on the idea of providing an ontology-based DLS infrastructure that supports “scholarly interpretation and discourse”, Motta *et al.* [49]. This concept is centred on the various services that an ontology-based server can provide to a DLS implementation in the context of a semantic network that supports multiple user interfaces within a number of supported software platforms. Motta *et al.* [49] state that such a model addresses certain issues that are usually associated with digital preservation such as the issues that are related to the usability concerns that usually accompany the process of knowledge formalization within DLSs, as previously discussed in Section 2.1.3.

2.1.6 DLS Implementations

As mentioned in Section 2.1.3.2, on the whole DLSs are user-centred applications, as one of the major concerns that DLS initiatives should address is the fulfilment of the requirements of the “myriad of users that will access the collections”, Greenaway [28]. The term ‘collections’, refers to the set of digital objects that any given DLS may hold and manage. Therefore, there are a number of distinct and notable DLS implementations that span different system and operational environments while serving user and community needs.

The underlying DLS implementation in any given environment depends on the actual context it is being designed for. Consequently, this imposes different technical and operational requirements that any given DLS should be able to adequately address in order to arrive at a functional model that meets its user needs. In this regard, a combination of software and hardware components within an appropriate architectural model contribute to the establishment of a useable DLS model that supports its users at different application levels, typically within the lifecycle of a digital object (creation, archiving, exposition, retrieval, presentation). Furthermore, different DLS implementations drastically differ in the design and architectural paradigms that they adopt, leading to the availability of an interesting range of systems that have a number of distinctive architectural and operational characteristics.

Some DLS implementations adopt a totally flexible and scalable Service Orientation (SO) approach, while others are custom-built implementations that suit a specific environment. What follows is a summary of some of the notable DLS implementations that demonstrate the latest developments that DLSs have witnessed, and the range of services they provide their respective user communities with.

2.1.6.1 *Digital Library of India (DLI)*

One of the notable DLS implementations is the Digital Library of India (DLI) that is relayed on more than 20 content creation centres, including the Indian Institute of Science (IISc), Carnegie Mellon University (CMU), the International Institute of Information Technology, Hyderabad (IIITH), and many other local content sources [55]. DLI represents a good example of a research community’s initiative to transform a collection of books into a digitized collection that is accessible online through the network of a group of participating institutions. Moreover, one of the prime objectives

of DLI is to strengthen and promote the concept of public community access to digital materials [56]. Furthermore, Chandrashekara and Varatharajan [55] indicate that the DLI is one of the world leaders in enabling digital content access. It makes digital contents available to the interested audience for free while supporting a variety of technological means to view the managed digital collections and objects.

The ultimate objective of DLI is centred on the goal to “preserve all the knowledge of the human race in digital form and make that content searchable, independent of language and location, and to ensure that the cultural heritage of countries like India is not lost during the transition from paper to bits and bytes”, Chandrashekara and Varatharajan [55]. Hence, it is not just a matter of providing access to the collection of a single entity; it is also a nationwide initiative whereby a number of disparate institutions participate in the system to build its overall digital collection. This shared nature characteristically represents a typical implementation manner in DLSs, where the actual implemented architecture attempts at integrating heterogeneous digital content sources. These sources may be spread across a number of distributed databases and digital repositories [57].

At the time DLI was reviewed, it had no integration with social networking platforms. This is a disadvantage in an online DLS that aims to provide accessibility to digital media. Another observed disadvantage is the fact that it does not have a clear framework for reuse, customization or enhancement. The current implementation seems to be built to serve a specific purpose; it is not flexible enough to undergo major customization if needed.

2.1.6.2 Pergamos Digital Library System

The Pergamos DLS is a system that was developed specifically to handle a heterogeneous and complex set of data objects that belong to the collections of the University of Athens [58]. Pergamos is a good example of a DLS implementation where its contents are augmented from a number of disparate sources including the Senate Archive, the Theatrical Collection, the Folklore Collection, and the Papyri Collection, which are among the richest digital data sources in Greece [58]. Consequently, the end result is a unified collection that is in the form of a highly accessible DHR that employs a powerful digital object manipulation mechanism based on a number of custom-developed components [58].

Despite the complex workflows that this DLS is supposed to handle, it lacks independent workflow management components. This is a limiting factor in terms of its ability to handle its convoluted workflows as well as simultaneous user sessions.

2.1.6.3 DSpace

DSpace is the result of collaboration between Hewlett-Packard (HP) and the Massachusetts Institute of Technology (MIT) library, which resulted in the development of a unique and rich DLS that acts as an institutional document library [59]. DSpace encapsulates the concept of a custom-made digital repository that targets a specific range of digital contents to serve a particular community of users and knowledge seekers across a number of participating institutions. It was developed to serve as a generic package that can be customized to suit any particular operational environment in a paradigm that made it one of the popular frameworks to build customized DLS implementations, especially within an academic or scholarly context. Based on that, the DSpace model differs from the two above mentioned systems as it can be customized and tweaked to suit the particular needs of any given user community due to the flexible nature of its implementation framework. This flexibility was achieved through an SO approach that made it a straightforward process to develop different DLSs that are built by utilizing the DSpace services.

Despite the adopted SO approach, this DLS lacks any sort of workflow management components. This may result in limiting the efficiency of this DLS as its highly distributed operations may benefit from a customised workflow management engine for better management and synchronization. Moreover, it has no social media or information mashup (see Section 4.6) components despite its academic context.

2.1.6.4 Education Resources Information Centre (ERIC)

One of the most noteworthy educational-oriented DLS implementations is the model represented in ERIC (Education Resources Information Centre), which is an online DLS that is oriented around research and information-enabling. This system is sponsored by the Institute of Education Sciences (IES) of the US Department of Education [60]. The significance of ERIC is evident in the fact that it is considered to be “World’s largest digital library of education literature”, ERIC [61].

ERIC manages a huge digital collection of more than 1.2 million bibliographical records that mainly comprise textual contents in the form of journal articles and other related material [61]. It is a good example of a DLS that has a specific focus on a certain niche of users while serving their needs based on a specific type of content, which is in this case is full text journals and academic contents [61].

Similar to DSpace and Pergamos, this DLS does not have any sort of workflow management components or modules. It also does not have social media integration despite its rich contents and wide outreach to varied groups of users.

2.1.6.5 Building Resources for Integrated Cultural Knowledge Services (BRICKS)

Building Resources for Integrated Cultural Knowledge Services (BRICKS) is a DLS developed as part of the European Integrated Project [62]. The prime goal of BRICKS is to develop an open user SOA to facilitate knowledge sharing within a cultural heritage context. Moreover, the ultimate goal of BRICKS is to “build a Europe-wide distributed Digital Library in the field of Cultural Heritage”, Hecht and Bernhard [63].

BRICKS uses the Internet as the backbone of its services while fulfilling the requirements of “expandability, graduality of engagement, scalability, availability, and interoperability”, Risse *et al.* [64]. Such characteristics are highly desirable in DLSs as they lead to implementations that are highly flexible and are able to handle the evolving user needs. Hence, BRICKS adopts a similar approach to the one adopted in DSpace, as shown in Section 2.1.6.3.

Although BRICKS is a system that aims for high levels of scalability, availability, and interoperability, it does not have a workflow management backend. This could prove to be a shortcoming especially in large enterprise implementations of BRICKS where its operations and data flows become complex and intersecting. On the other hand, BRICKS contents are confined to the data objects obtained from the participating museums and cultural institutions. This limitation is imposed by the fact that BRICKS is not integrated with any information mashps or social networking applications, which usually act as valuable sources of added value contents to digital repositories.

2.1.6.6 ARCO

The Augment Representation of Cultural Objects (ARCO) Project is a project led by the Computer Graphics Centre at the University of Sussex. The ARCO DLS allows

museums and other cultural institutions to manage their own digital objects through the tools and services it provides. ARCO also empowers its users to build and maintain virtual museum exhibitions to be presented online. Such virtual exhibitions are built by utilizing a number of techniques including virtual and augmented reality [65]. It enables the production of virtual museums in various formats, including, for example, web pages that have embedded 3D objects [64].

ARCO comprises a number of DLS tools including a content management system called ACMA (ARCO Content Management Application) that constitutes a cultural object manager, a presentation template manager, and a presentation manager. ACMA utilizes a number of technologies that include XVRML (XML Virtual Reality Modeling Language), Augmented Reality Interface (ARIF), etc. These tools can be installed in any museum and used to manage its collections of digital heritage objects.

ARCO has no integration with information mashup applications; however, it can be used as a data source in independent mashup implementations (more on this in Section 4.6.1). It can also be observed that despite the possibilities that the richness of the ARCO contents can offer, it does not have any sort of social networking platform integration. This limitation confines user interaction with ARCO to the traditional data browsing and input/output operations.

2.1.7 Comparison of DLSs

Table 2-1 presents a comparison of the discussed DLSs with particular emphasis on their common features. In the context of this thesis, it is notable that all the discussed DLSs do not exploit WfMSs, and mostly do not take advantage of social networking technologies and Web 2.0 mashups. For this, and other practical reasons (such as the viability resource wise of implementing such large systems as test beds for the research presented in this thesis) this thesis presents another DLS system (DISPLAYS) as a conceptual model from which a validating DLS called the Reanimating Cultural Heritage (RCH) system has been implemented as a DLS to evaluate workflow and social network technologies for inclusion in DLSs. This work is presented in more detail in Chapters 4, 5 and 6 of this thesis.

Table 2-1 Comparison of Example Digital Library Systems

Feature	DLI	Pergamos	DSpace	ERIC	BRICKS	ARCO
User Community	Nation-wide	Nation-wide	One institution	Several institutions	Europe-wide	Museums in general
Digital Heritage	Yes	Yes	No	No	Yes	Yes
Workflow Management	No	No	No	No	No	No
Social Networking	No	No	No	No	No	No
Content Creation Services	Yes	Yes	Yes	No	Yes	Yes
Archival Services	Yes	Yes	Yes	Yes	Yes	Yes
Retrieval Services	Yes	Yes	Yes	Yes	Yes	Yes
Presentation Services	Yes	Yes	Yes	Yes	Yes	Yes

2.1.8 The Role of DLSs in Digital Heritage Preservation

The discipline of “Digital Library Cultural Heritage Resources” [2] represents a rich area within modern DLS implementations due to the pivotal role that DLSs play in that arena [2]. DLSs are utilized to preserve digital heritage objects due to the powerful features that they provide in relation to the management and preservation of digital assets. On the other hand, it can be observed from the reviewed DLS literature that the

concept of Digital Curation [2] is one that can be naturally linked to DLSs. Pennock [2] indicates that Digital Curation is a recent term that applies to the process of maintaining and adding value to a group of digital information for both current and future use. Moreover, according to Rusbridge [66] the Digital Curation concept is concerned with the idea of “communication across time”, and this raises the issue of preserving and interpreting digital heritage objects in a useful way within a suitable preservation and distribution environment, which can ultimately be a DLS.

It is fairly common to see Digital Curation attempts and initiatives that are usually accompanied by the implementation of digital resources in one form or another. Such an approach is described by the work carried out by Marchionini and Shah [67] who indicate that the process of Digital Curation primarily involves selecting, preserving and ensuring access to a repository of digital information. Moreover, the work carried out by Marchionini and Shah [67] involved what is called The Vidarch Project, which aimed to “develop policies and tools that help video curators discover and add contextual elements that will help future generations not only find but also make sense of video content”. This represents a good example of the utilization of an online DLS that aims at realizing the Digital Curation objectives of a certain community of practice, such as museums and cultural institutions.

The goals of “long-term access and use of meaningful and authentic digital resources” Lee *et al* [68], comes at the forefront of the goals and objectives of digital heritage preservation systems and their associated digital Curation attempts. Additionally, according to Ray [69], Curation is all about ensuring that “digital data will be preserved in meaningful form into the future and managed so that information can be found when needed by those who want it”. Therefore, Digital Curation mainly involves the entire life cycle of a digital object, and the efficiency of Digital Curation applications is measured by the effectiveness by which their management tools can handle such a complex lifecycle [68].

A number of digital heritage preservation projects are directly linked with the successful utilization of integrated Digital Curation and DLS implementations. One of the remarkable examples in this arena is the one cited by Ray [69] who covered the development of a Digital Curation infrastructure built by Purdue University in the US. Purdue University established a fully distributed Curation Centre that aimed at preserving and organizing digital objects, while facilitating the process of managing and

accessing them by interested groups of users and scholars. On the other hand, another notable initiative that can be found in modern DLS literature is the Digital Curation Centre (DCC), which was developed in the UK to realize the objectives of “long term stewardship of digital assets”, Rusbridge [66]. This project is a good example of an attempt that highlights the ways by which online digital resources in general, and DLSs in particular, are fully utilized to serve the needs and requirements of a particular Digital Curation initiative.

In the context of this thesis the RCH (Reanimating Cultural Heritage) system was built to validate the proposed WfMS implementation. RCH has a strong digital Curation element as it is used to preserve and reanimate the cultural heritage objects of Sierra Leone, as described in Chapter 4.

2.2 Workflow Management Systems

WfMSs are primarily designed for the goal of improving the businesses process of any given system by providing it with the necessary tools to automate and manage its processes [70]. Practical implementations of WfMSs are widely used to improve organizational performance and efficiency in terms of managing a set of system-specific workflows that are imposed by the managed system’s processes and data flows [71].

Aalst [4] states that WfMSs innovations formed a promising solution for an age-old problem, which is concerned with the optimization, monitoring and support of business processes in a given operational environment such as an enterprise software infrastructure. Such systems can also prove to be beneficial to DLS implementations as the author of this thesis showcases in [5].

The term ‘Workflow Management’ refers to “the ideas methods, techniques and software used to support structured business process”, Aalst and Hee [72]. The main objectives of workflow management are oriented around the concept of having better streamlining and ease of management and maintenance of the business processes in any given system. WfMSs primarily aim at helping users achieve their goals and objectives with high levels of efficiency by means of sequencing workflow activities and invoking the appropriate human, application or information resources that are associated with these activities [73]. This process ultimately aims to arrive at fully orchestrated controllable managed system services [73]. Workflow management activities are usually complemented by a number of services that contribute to achieving the above

goals while interacting with the underlying managed components. Typical workflow management services may include process monitoring and tracking services, and different process management and coordination tools that are usually associated with modern WfMSs implementations.

2.2.1 What is a Workflow Management System?

According to Schael [74], WfMSs are considered to be one of the innovative solutions for process improvement and optimization. WfMS play a pivotal role in improving the overall throughput of the system in question by means of managing its data and control flows – can this innovation be proved to apply to DLS architectures? To investigate this, this thesis presents a conceptual DLS in Chapter 3, followed by practical implementations of this DLS in Chapters 4 and 5, followed by a simple but stable test bed WfMS that implements the same DLS components as in Chapter 4, but as a WfMS managed one as shown in Chapter 6.

Aalst and Hee [72] define WfMSs in simpler terms by describing them as generic software packages that are dedicated for the purpose of business process management. Notably, WfMS implementations place a certain emphasis on the automation of task execution as indicated by Munaga *et al.* [75], who argues that “a workflow aims to automate business processes, where documents and information are passed between agents according to a set of rules to achieve or contribute to an overall business goal”.

A more contemporary definition of WfMSs is provided by the Workflow Management Coalition WFMC [76] cited in [77], where it is stated that a WfMS is “a system consists of process definition tools, administration and monitoring tools, client applications, invoked applications and workflow engines”. In such a paradigm tasks are performed either by the system’s end-users or its applications, based on a workflow design that is produced by process definition tools [76] cited in [77]. Moreover, according to Yu [78] a WfMS is responsible for defining, managing and executing workflows on a computing system at resource level. Therefore, WfMSs can reach high levels of complexity and sophistication due to the nature of the systems and business process patterns that they need to handle efficiently and manage fully.

What can be implied here is that WfMSs can be made an integral part of a wide variety of software systems that need efficient workflow management capabilities to improve their performance, have better management of their components and data and control

flows – and hence this should apply to DLSs. Furthermore, the way that a WfMS integrates with any software infrastructure varies according to the adopted implementation technology and architectural paradigm as well as the actual specifications of the system to be managed. This thesis proposes the integration of a WfMS within test bed SO DLS components, as detailed in Chapter 6.

2.2.2 WfMSs Functionality

A WfMS typically serves three main dimensions, which are process, organization and infrastructure [70]. These dimensions are detailed as follows:

- the term ‘**process**’ primarily refers to “the business logic that captures the activities, their inter dependencies, and associated people and applications required to meet the underlying business goals”, Lin et al. [70].
- the ‘**organization**’ model refers to the different components, which are either application-based or user-based that a WfMS has to deal with [70].
- the ‘**infrastructure**’ dimension encapsulates the technical and operational aspects that a WfMS must be able to integrate with while providing its services; this includes the network infrastructure and the associated applications that may exist in the concerned implementation environment [70].

Figure 2-3 illustrates a typical WfMS implementation where a number of baseline components and services work in coordination with each other. The illustrated model is based on the five services identified by Lin [70]: a Process Definition Tool, Process Definition Repository, Workflow Engine, Worklist Handler and Administration and Monitoring tools.

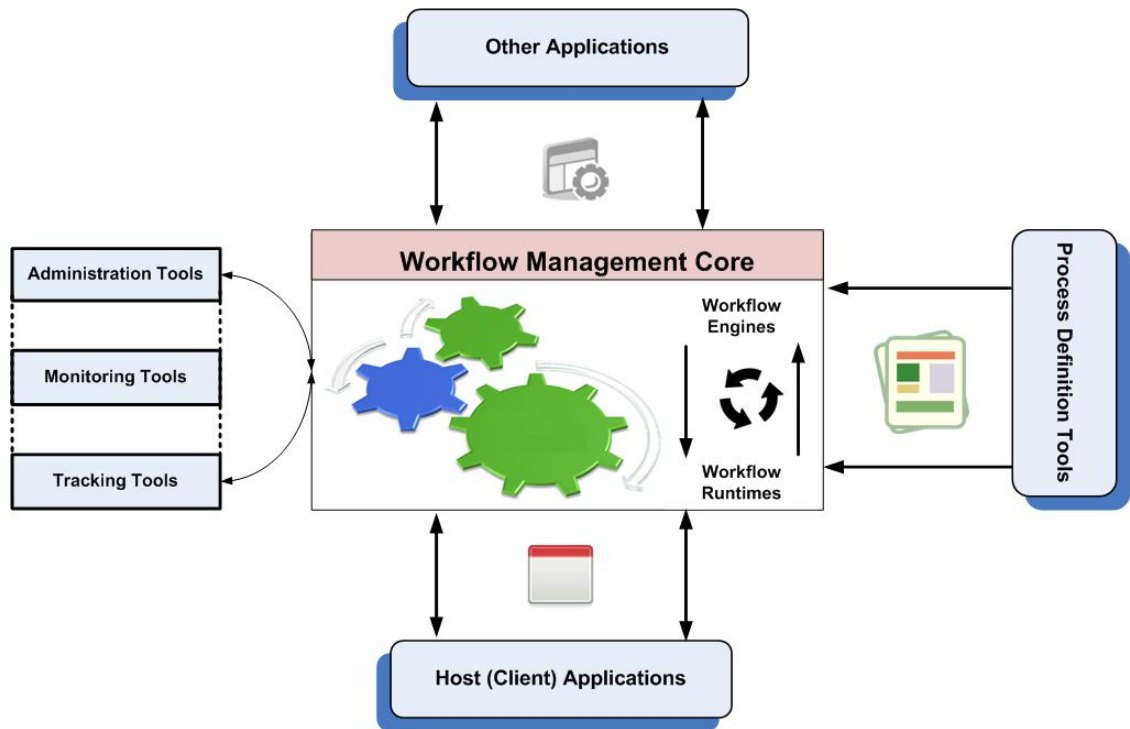


Figure 2-3 Typical WfMS Components

The reviewed workflow management literature shows that, although the above five services are equally important, there is a general consensus that a WfMS should at least provide the last three services to pave the way for a functional model that handles the system processes effectively. The importance of a workflow engine within a WfMS is evident by the fact that, through its built-in modules, it has the ability to “interpret the process definition, interact with workflow participants, and invoke the use of information technology tools and applications”, Zhan [79]. On the other hand, the importance of having some kind of Worklist Handler is evident in a number of prominent workflow management solutions. Such solutions provide the necessary capabilities to effectively handle the different work items or processes that flow throughout the managed system. It is equally vital to have an appropriate range of Administration and Monitoring services that provide a number of sophisticated tools that directly interact with the system’s work items and business processes. These tools provide different performance indicators as well as administration capabilities (invocation of a process, termination of a workflow, etc.).

2.2.3 A Brief History of WfMS

Automated WfMSs are not as recent as they seem, as commercial versions can be traced back to the early 1990s [71]. Additionally, the early implementations of WfMSs are

linked to the early attempts to automate as well as support business applications through devising specialized software applications to manage and monitor their operation [74]. WfMSs can be considered as a resemblance of the early office automation systems as “using workflow technology to support cooperative activities is an old idea, taking its sources in Office Information Systems”, Charoy *et al.* [80].

The earliest examples of WfMSs can be dated as far back as 1977 as seen in the work done by Zisman [81], cited in [74], which is considered to be among the pioneering stages in the development of WfMSs. On the other hand, on a conceptual level, the actual concept of devising specialized software packages to act as mediums for the purpose of workflow management dates back even earlier than that, as can be seen in the work carried out by Ellis [82] who discusses the different aspects of devising a Mathematical Model of Office Information Flow.

2.2.4 The Importance of Workflow Management Systems

The importance of WfMSs in modern distributed software implementations can never be overestimated due to the role they play in process improvement and resource utilization (software/hardware). According to Frey [83], WfMSs are becoming more of a common and prominent medium for handling business processes within software systems inside organizations of all sorts and types. The anatomy of a WfMS is mainly connected to the need to automate procedures in software systems where files and data flow between the system components according to a number of predefined rules [84]. A WfMS implementation can supplement and improve the functionality of an existing software infrastructure by providing it with better management and monitoring capabilities paving the way for more efficiency and flexibility. WfMSs typically support process execution by means of managing it and the associated flow of data, while ensuring that the individual workflows are executed at the right time by the right person and or the designated system processes [72]. Moreover, WfMS can be looked at from the perspective that they represent a drastic upgrade from traditional unorganized process management to a more structured and organized way of managing an organization's business processes [83].

2.2.5 Workflow Management Standards

There are a number of standards and frameworks that govern WfMS implementation and design paradigms according a set of guidelines that impose a high level of standardization on WfMS implementations, as detailed below.

2.2.5.1 *The Workflow Reference Model*

The Workflow Management Collation (WfMC) is one of the most prominent bodies in standardizing the processes involved with WfMSs design, implementation, operation and maintenance [15]. Moreover, WfMC is credited to be the first organization that became actively involved in promoting workflow standards [15]. The development of such standards adheres to WfMC's vision and strategic objectives as it represents "a grouping of companies trying to establish standards that will facilitate the interoperability between workflow systems", Eloff and Botha [85].

At the forefront of WfMC's standardization efforts comes the Workflow Reference Model (WfRM), which provides a highly advanced conceptual model used to govern the design and implementation of WfMSs. Hollingsworth [84] indicates that the WfRM provides a common WfMS implementation model that identifies their characteristics, underlying terminology and operational components, paving the way for a contextual model that can be adopted in any WfMS. Moreover, Eloff and Botha [85] indicate that the WfRM describes the fundamental concepts that are associated with workflow management, complemented by an architectural model that addresses the interfaces between the different components of a typical WfMS. More importantly, the WfRM is considered to be a generic overall model for WfMS implementations [86], with high levels of applicability to a variety of complex software systems. This paradigm is encapsulated in Hollingsworth's [87] view of the WfRM as he states that "the model attempted to construct an abstract view of the business process in terms of its core characteristics, separated from the technologies that could be used to deliver its functionality in a real world situation".

The WfRM defines a number of standards, guidelines and rules against which the efficiency of any given WfMS implementation can be measured according the way it is being designed and implemented. Furthermore, this reference model, that was published in 1995 [87], has three significant categories that are used to formalize an overall reference model that can be followed by WfMS implementations of different types and

natures. These categories as identified by Hollingsworth [87] are: “a common vocabulary for describing the business process”, in other words a workflow definition language, “a functional description of the necessary key software components” and “the definition, in functional (or abstract) terms, of the interface between various key software components that would facilitate exchange of information in a standardised way”. The core conceptual components of WfRM are illustrated in Figure 2-4.

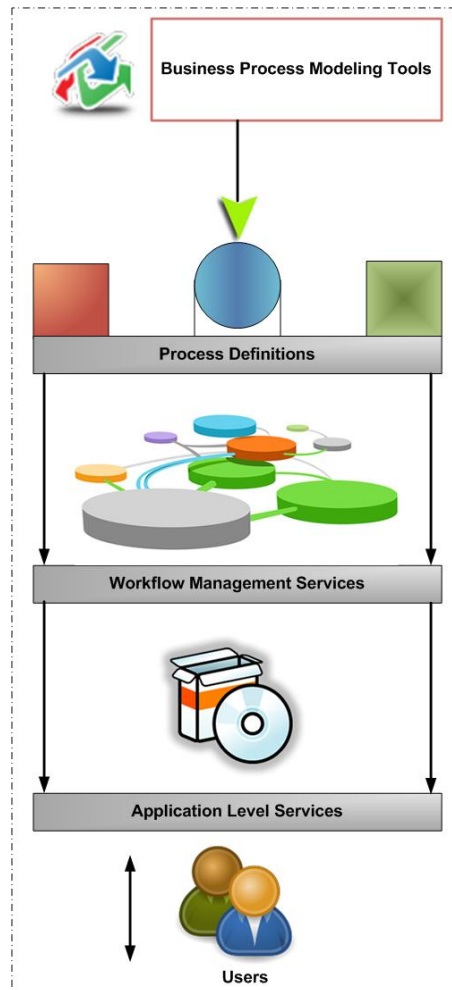


Figure 2-4 The WfRM Categories

Going into the details of the WfRM is outside the scope of this thesis. What should be explained here is that it plays an important role in contemporary WfMS implementations due to the wide acceptance that it was met with by the developers and vendors of WfMSs. Despite some sceptical views about the effectiveness of the levels of abstraction and generalization that the WfRM provides [87], it still can be considered as one of the most widely used WfMS standards. It is worth mentioning here that the WfMS prototype devised as a part of this thesis comprises some of the WfRM components while interacting with the system it manages (more on this in Chapter 6).

2.2.5.2 Business Process Execution Language (BPEL)

The Business Process Execution Language (BPEL) is a business process design standard that represents a model used for defining process execution rules within a WfMS [15]. BPEL is different from the WfRM because it merely represents an XML-based language that models a business process [88]. BPEL comprises a set of basic activities that constitute the workflow management tasks to be performed [88]. These rules are complemented with what is called ‘compound activities’ according to BPEL’s terminology, which are used to model the control flow activities of the managed system. Additionally, Fernandes [88] argues that what makes BPEL a unique as well as a powerful WfMS design standard, is the fact that it consists of a mixture of structured and unstructured representations of workflow patterns. This availability of workflow patterns makes BPEL suitable for virtually any WfMS implementation. According to Mendling [15], the heart of a BEPL implementation is the “so-called BPEL engine”, which has the full capability of executing the process definitions of a BPEL model.

One of the negative sides of BPEL is the fact that it offers limited support for an Event-Driven Architecture (EDA), and this poses a number of limitations when using it for systems that adopt that paradigm [89]. This may prove to be a serious shortcoming in complex implementations that “involve multiple applications (or application components) that run on distinct physical machines across an enterprise network”, Saini [89].

2.2.6 Workflow System Examples

There is a number of open-source and closed-source WfMS development tools and languages that vary in their implementation paradigms and the services that they provide. This is due to the inherent vibrant nature of WfMSs and the dynamic nature of the user and system requirements that require such a wide range of solutions. What follows is a brief outline of some of the noteworthy WfMS development tools and systems.

2.2.6.1 Yet another Workflow Language (YAWL)

Yet another Workflow Language (YAWL) [90] is one of the notable open-source WfMS implementation languages. As the name suggests, YAWL is considered to be a workflow management language rather than an implementation framework. The prime

goal of YAWL is to provide more powerful support to complex workflow patterns [10]. In addition, Garcês [10] states that YAWL was developed by “taking Petri nets as a starting point and adding mechanisms to allow for a more direct and intuitive support of the workflow patterns identified”.

YAWL was developed by Wil van der Aalst, Lachlan Aldred, Marlon Dumas and Arthur ter Hofstede, members of the Faculty of Information Technology of Queensland University of Technology [10], and based on their identification of a number of shortcomings that conventional workflow management development languages and paradigms fail to address. What is really significant about YAWL is that it provides for a flexible development model that can be used for the development of flexible and adaptable WfMS applications. Additionally, YAWL is considered to be one of the most expressive and mature open-source WfMS development tools when compared to other open-source WfMS development languages and frameworks [91]. The powerfulness of YAWL stems from its ability to overcome the limitations of traditional workflow modelling techniques by means of the innovative and direct support of virtually all workflow patterns [92].

YAWL is the result of the analysis of existing WfMSs and their related standards by utilizing a number of comprehensive workflow patterns. Moreover, Aalst *et al* [92] indicate that such analysis led to the conclusion that WfMS-relevant standards, as well as their associated theoretical models, have problems in supporting optimal workflow management patterns. This shortcoming inspired the development of YAWL to fill such gaps, and provide developers with the tools they need to build comprehensive WfMS solutions that can handle any level of workflow complexity [92]. Additionally, YAWL effectively covers the different areas of workflow management that a system might need, such as the collaboration, monitoring and the data and control flow aspects of a WfMS implementation [92].

2.2.6.2 Bonita

Bonita is an open-source system that features a number of innovative tools that can be exploited to deliver sophisticated WfMS implementations [93]. It differs from YAWL in that it is actually a standalone application that provides a set of built-in workflow design and management tools that can be used by developers to build customized WfMSs.

According to Siddiqui [94], Bonita relies on Java's J2EE as an implementation foundation. Its components are centred around three main tools: an innovative studio for process modelling, a powerful BPM & Workflow engine, and a user interface which can be utilized to build a variety of workflow designs [95]. Bonita provides a graphical tool for designing workflow patterns by means of using a number of easy-to-use controls that can be utilized to build complex workflow management models [93]. Moreover, Bonita is fully compliant with WfRM [10] and is capable of handling long-running and complex business and user oriented workflows [10].

2.2.6.3 Windows Workflow Foundation (WF)

Windows Workflow Foundation (WF) is considered to be one of the most prominent technological solutions for devising WfMSs. WF is based on Microsoft's .NET Framework and offers a range of development tools that support the full implementation of WfMSs of any scale. WF offers a number of advanced tools that are dedicated to building comprehensive workflow solutions by the utilization of a dedicated programming model, consisting of a number of .NET components supported by a powerful workflow engine [96]. Furthermore, according to Scott [97], WF represents a solution that encapsulates a set of tools that enables the process of defining, executing and managing workflows.

The development environment that WF delivers consists of a number of standard code controls that contribute to optimal development time and unlimited customization and integration possibilities. Although WF is a Microsoft technology, it can still be integrated with software solutions provided by other vendors, as can be seen in the scenario highlighted by Farahbod [96]. Farahbod [96] covered the process of integrating WF applications with an IBM database by utilizing Microsoft's Visual Studio (WF's development environment).

According to Bukovics [98] there are a number of advantages that WF provides WfMS developers. These advantages include the flexible and powerful framework that WF provides, the consistent development model that it supports and maintains, its support of a variety of advanced workflow models and patterns, its advanced support of domain-specific problem solving, infinite extensibility and its support of the development of complete workflow ecosystems [98]. Such advantages are complemented with an architectural approach that provides for a flexible implementation framework in an SO

manner, this ensures effective and modular workflow infrastructure implementation [99].

Figure 2-5 illustrates WF's extensible model showing a number of services from its baseline functional model. These services include the Persistence, Tracking, Scheduling and Transaction services. These services have the flexibility to be hosted within a host application that can be assigned the task of providing extra user functionality, as well as user interaction elements if needed. The provided services can also interact with a database model that can be either Microsoft's SQL Server or any of the database technologies that the .NET Framework supports.

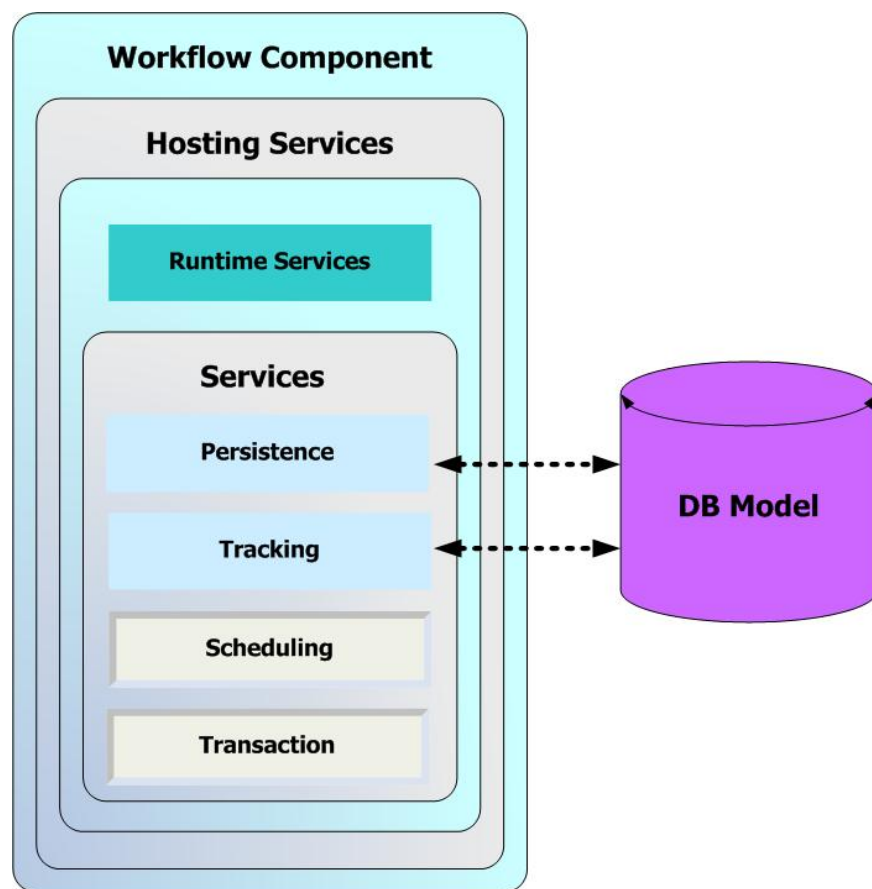


Figure 2-5 WF's Extensible Model

What is unique about the aforementioned paradigm is that a WF workflow management solution has the flexibility to be hosted in a number of software platforms and applications. This flexibility is evident in the fact that WF applications can run within a host, for example, a SharePoint Portal Server, a Windows Application, Console Application or an ASP.NET Application [100]. It is also worth mentioning here that what is fundamental to the whole WF model is its powerful Workflow Runtime Engine that is pivotal in providing the necessary workflow execution functionalities as well as

responding to the system's events. Furthermore, the WF Workflow Runtime Engine provides other services such as the workflow state management, threading, tracking and persistence services [30]. WF was the chosen technological medium to implement the proposed WfMS for a number of practical reasons, as detailed in Chapter 6.

2.2.7 Workflow Management System Comparison

When comparing WfMS implementation tools and frameworks, a number of similarities and differences can be found. In the context of this thesis, three sample WfMSs are illustrated (YAWL, Bonita and WF). Among these WF was chosen as the implementation medium for the proposed WfMS prototype (this will be discussed further in Section 6.3.1).

Wu *et al.* [101] indicate that YAWL (see Section 2.2.6.1) and WF (see Sections 2.2.6.3 and 6.3.1) are designed to be “general purpose” workflow management implementation tools. YAWL is inherently supported by a workflow execution engine as well as a graphical workflow design editor. It also supports an extensible model that allows external applications to connect to a devised workflow engine in an SO approach. On the other hand, WF supports a flexible programming model and “rehostable” [101] runtimes that can be used to implement complex and long-running workflows. In a similar fashion to YAWL, WF's workflow runtime services can be designed and edited by a visual workflow designer which is the Visual Studio application. Furthermore, according to Roy [102], WF is specifically designed to provide developers with the necessary base components that are needed for building workflow management and process automation modules in an SO manner.

Bonita (see Section 2.2.6.2) is more domain-specific when compared to YAWL and WF, as it is designed to be a browser-based system for implementing and hosting workflows within J2EE implementations [94]. Moreover, Bonita depends entirely on web services (also supported by WF) in conjunction with SOAP to provide workflow management functionality in distributed systems.

According to Jang *et al.* [103], one of the distinctive features of WF is the flexibility it provides by combining its graphical design tools with the ability to build lower-level code modules programmatically. It therefore can deliver more complex customized workflow management functionality supported by WF's extensible model. It is also possible to devise workflow components that are entirely written in code. These

independent custom-built components can be then hosted in either web-based or custom-built client applications (hosts).

Table 2-2 below compares YAWL and Bonita (open-source solutions) with WF (a closed-source solution) in relation to a number of workflow management functional areas, based on those identified by Garcês *et al.* [10]. These features are mainly related to workflow runtime creation, running and monitoring, which are pivotal to WfMS implementations.

Table 2-2 WfMS Implementation Tools Comparison

Feature	WF	YAWL	Bonita
Process definition tools and support	Yes	Yes	Yes
Built-in workflow client application	No	Yes	No
Administration and monitoring tools	Yes	Yes	Yes
Runtime Platform independence	Yes	Yes	Yes
Support for Web-based implementations	Yes	Yes	Yes
Support for Standalone implementations (i.e. no need for a host)	No	Yes	Yes
Other software required	No	Yes	Yes
DBMS Integration	Yes	Yes	Yes

Although WF and YAWL seem to share a number of similarities, there are a number of factors that favoured utilizing WF to build the proposed WfMS prototype. The main shortcoming of YAWL includes its limited support of workflow hosting in independent hosting mediums. This imposes certain limitations especially in terms of creating multiple views (interfaces) for the same workflow engine. As RCH needed a workflow management engine that would not interfere with its UI elements, WF seemed to be a better solution in this regard. More details about the factors that favour WF are discussed in more details in Section 6.3.1.

2.3 WfMS in DLS

It can be noted that the dominant paradigm in DLSs is to provide their services to a wide spectrum of users within an inherently complex distributed system. Despite the apparent need of effective workflow management capabilities within DLSs (due to their intersecting processes), WfMS literature rarely discusses integrated workflow solutions that are fully utilized in DLSs. This gap in the literature raises a number of questions in regard to this apparent lack of effective integration with all the performance and system management implications that this shortcoming might have.

According to Gang [11], traditional WfMs often fail to meet the dynamic demands of distributed systems, such as those that are web-based due to their inherent dynamicity and interactivity. Such complications require the adoption of flexible tools that are able to manage the vibrant processes and components of a DLS. Although it is a traditional trend that WfMSs are taking some conceptual elements of document management systems [12] cited in [13], conventional WfMSs solutions often fail to address the needs of the most fundamental components of a DLS due to their complexity. These components are the archival, retrieval and presentation components.

The research work that was conducted as a part of this thesis aimed to pave the way for a viable implementation model to integrate a WfMS within a DLS. The paper entitled “A Dynamic Workflow Management Framework for Digital Heritage and Technology Enhanced Learning” [5] written by the author of this thesis, was aimed at devising a WfMS prototype that integrated with a digital heritage system. The paper “An integrated workflow Management solution for heritage information mashups” [6] goes a step further and aims to develop a WfMS to accommodate rich information mashups.

These attempts formed part of the groundwork of the implementation model proposed in this thesis, which will be fully discussed in Chapters 6 and 7.

2.4 Summary

This chapter discussed the different practical and technical aspects of DLSs and WfMSs. It was evident that technological advancements made the utilization of DLSs a common trend, as they played a more integral role in the attempts to preserve and distribute digital data objects within digital resources and repositories [21]. The reviewed literature showed the DLSs evolved over time in parallel with the unlimited possibilities offered by the proliferation in data digitization formats and tools as well as computer networking technologies. It was also observed that DLSs are always accompanied by a set of user expectations that must be adequately addressed to arrive at fully functional models. Such expectations range from the need to provide an instantaneous access to information to the provision of highly interactive and efficient user interfaces.

In the literature review, it was also demonstrated that DLSs are complex in nature, and that their implementation often faces a number of technical and practical hurdles. These hurdles include the demanding needs of managing the rich digital collections that they usually hold, the challenges related to the process of providing rich archiving capabilities, the challenges of arriving at viable search and retrieval components, and the issues related to user actions and the need to maintain a certain level of security and privacy. Therefore, rather than resorting to generic ready-made off-the-shelf solutions that may fail to deal with such challenges, custom-made DLSs to address domain-specific problems were often created. This notion was highlighted with a number of representative DLS implementations that included, the Digital Library of India (DLI), Pergamos Digital Library System, the DSpace System, Education Resources Information Centre (ERIC), the Building Resources for Integrated Cultural Knowledge Services (BRICKS) project, and the Augment Representation of Cultural Objects (ARCO) system. As WfMSs play an effective role in improving the business process and efficiency of various software systems, their utilization within DLSs seem a viable option necessitated by their complex nature. However, such integration has to overcome a number of implementation challenges as will be explained in Chapter 6.

WfMSs were also discussed with a particular emphasis on their underlying functionality and the important role that they play in managing the business processes of any given system. Within that context, WfMS standardization efforts were discussed while covering the Workflow Reference Model (WfRM) and the Business Process Business Process Execution Language (BPEL). Moreover, representative WfMS solutions that comprised a group of open-source as well as closed-source development tools and techniques were discussed. The range of the examined systems provided different services in terms of the workflow management capabilities that they empower. The discussed systems included Yet another Workflow Language (YAWL), Bonita and Windows Workflow Foundation (WF).

This thesis proposes the utilization of an integrated WfMS solution within a DLS implementation. Based on that, the subsequent chapters of this thesis will outline the practical and technical aspects of devising the proposed solution. Chapter 3 will highlight the DISPLAYS system, which is a DLS implementation framework. Chapter 4 showcases the RCH system, which is an actual DLS implementation based on the DISPLAYS framework and which acted as the test bed for the proposed WfMS solution. Chapter 5 showcases the RCH components (archival, presentation and retrieval), while Chapter 6 illustrates how these components are being hosted within an integrated WfMS.

CHAPTER III

3 DISPLAYS Framework

This chapter presents an innovative digital library framework called **D**igital library Services for **P**LAYing with Antiquity and **S**hared Heritage (DISPLAYS). This DISPLAYS Framework [5] [7] [1] represents a digital resource model whose functionality revolves around the concepts of cultural heritage resource sharing and distribution. Chapter 4 then presents an implementation of a limited version of DISPLAYS as a validation architecture called the Reanimating Cultural Heritage (RCH) system. Chapter 5 takes a closer look at some of the key processes or components of the RCH system (the archival, retrieval and presentation components). These components are then implemented as a workflow based system in Chapter 6.

The current chapter gives a detailed description of the proposed DISPLAYS framework while emphasizing its main services. These are the **Digital Content: Creation, Archival, Exposition, Presentation and Interaction** services for digital heritage collections in addition to management services in the form of a set of workflow processes.

3.1 Introduction

Digital object management, sharing and distribution are major considerations when handling digitized cultural objects, due to their complexity and diversity on the one hand and the heterogeneity of the systems that usually handle them on the other. The task of creating common digital heritage application tools and repositories is further convoluted by the fact that different communities of practice use different standards, data formats and media to interpret and use their objects. Such complexities limit the ways by which such communities can share their collections unless they are provided with common shared tools (and standards) to create and maintain shared digital repositories. Such challenges make it necessary to devise effective DLS models that are capable of handling varied collections, while being flexible enough to be utilized by different communities of practice from the citizen and diasporas to museums.

It is proposed that the DISPLAYS framework provides a comprehensive conceptual model for building enterprise digital heritage data repositories that can be shared between a number of cultural institutions, leading to the organization of augmented data into meaningful information. This can be compared to the paradigm followed in some similar systems, such as the well-known Digital Repositories Infrastructure Vision for European Research (DRIVER) [104]. DRIVER promotes open access to European digital data across a network of physically distributed repositories throughout Europe while providing a set of specialized shared DLS services [105]. In a similar fashion, DISPLAYS offers a unique digital heritage resource infrastructure while exploiting a number of novel solutions that form its constituent components, as will be further illustrated in this chapter.

DISPLAYS functionality revolves around five core services: Digital Content: Creation, Archival, Exposition, Presentation, Interaction services, and an additional management service (or set of workflow processes) for digital heritage collections. Using the DISPLAYS framework, prototype DLS components and implementations were then built to allow the testing and evaluation of DISPLAYS concepts. One such implementation is the Reanimating Cultural Heritage (RCH) system [1], which is a DISPLAYS-based shared online DLS that is used in the context of this thesis to demonstrate effective workflow management (see Chapter 6). Thus, RCH is used as a ‘proof of concept’ system in the context of this thesis to validate its findings. A detailed description of RCH and its components is given in Chapters 4 and 5 of this thesis. Another example of a system that can be used to create a DLS implementation that adopts DISPLAYS concepts is the Augment Representation of Cultural Objects (ARCO) system [65] (see Section 2.1.6.6).

A DISPLAYS type implementation should build on the capabilities of Web 2.0 [106] by exploiting the Web as an effective medium for data-intensive operations. DISPLAYS does this by adopting a service orientation approach where the different system functions are organized into separate specialized web services. For example, DISPLAYS components could operate within a highly scalable distributed application framework [1] that acts as a functional baseline for the DISPLAYS services. DISPLAYS services should be flexible and customizable so that they can integrate with any existing digital heritage or knowledge management system. This flexibility paves the way for the creation of effective digital repositories that are capable of meeting the

digital object management and sharing needs of their communities of practice. Moreover, the complex and intersecting system workflows should be managed through specialized control and management services that control the different aspects of the system's data and control flows. A goal of the research presented in this thesis is to show that workflow management can be applied within a DISPLAYS type of system; this work is further detailed in Section 3.5.2 as well as Chapter 6.

3.2 DISPLAYS Concept

Cultural institutions, such as libraries, museums, archives, galleries, etc., are commonly considered to be natural partners due to the common grounds that they have with each other and the common interests that they share. Among those areas of interest, the process of digital content sharing and distribution is considered to be among the most significant [107]. DISPLAYS considers the process of 'sharing and distribution' of digital content to be refined into five distinct services that need managing through workflows. These are the Digital Content (1) Creation (DCC), (2) Archival (DCA), (3) Exposition (DCE), (4) Presentation (DCP), and (5) Interaction (DCI) services. This division generates a requirement for tools and services that can execute these processes for users (museum creators or visitors) to 'create', 'interpret', 'use' and 'explore' collections of objects in digital online spaces. Such services allow for better digital object management, distribution and presentation capabilities for the communities that collect and use digitized cultural objects.

The implementation of such tools and services is made possible through specialized software and hardware innovations, such as advanced open-source and commercial digital archive software, 3D modelling tools, touch screens, advanced web interfaces, etc. This trend of exploiting modern sophisticated technologies that can be utilized to build DLSs is further enhanced by the advent of Web 2.0, which itself was a milestone that profoundly changed the way that the web is used and utilized [106]. Through the exploitation of creative technical innovations (for example, web services, social networking, wikis, blogs, etc.), Web 2.0 transformed the Internet into to a common platform that can be used by its users to serve their common interests. Web 2.0 paved the way for the provision of effective and powerful online tools for users to connect, collaborate and share information in an environment that can handle various media such as, text, audio, videos, images, 3D objects, etc. This ability to share information online

using readily exploitable Web 2.0 technologies and sophisticated open-source (and commercial, but ideally not) software, can be exploited to build DISPLAYS type DLS for the communities of practice. Therefore, the opportunities that modern technology, especially web-related ones, has enabled can never be overestimated as “the emergence of web-based communities and services (such as social-networking sites), mobile and Wi-Fi technology offer new opportunities for citizens to create and share personal media and for cultural institutions to interact with their audiences”, Minerva [108]. Such advancements are of particular importance to heritage resource sharing applications as they usually depend on high levels of content sharing and delivery to enable effective digital content sharing services. These services are typically used among the disparate entities that may use the devised digital resource.

An example of a simple hypothetical implementation of a DISPLAYS type DLS is illustrated in Figure 3-1. Here, a simple DISPLAYS-based DLS can be implemented using common components such as Word and Excel to create appropriate text and metadata for digital objects, Adobe tools could be used to process images, videos, and even Flash presentations, and 3ds Max can be used to create 3D objects. This data can all be readily exported as XML data archives (requiring only appropriate XML schemas for validation). Such data can then be organized into expositions (folders giving meaning to the data) by, for example, Windows Explorer. Use of XML technologies, such as XSLT, can then be utilized to present these expositions to a web browser as a virtual museum or archive collection, etc. The web browser can have interactivity built in through other technologies such as Flash, or 3D web plugins, etc. This would make a very simple local DLS that could be further expanded by introducing web services to connect components and distribute each process across the Internet.

In such a scenario, the archival process or workflow would begin to use more sophisticated technologies such as appropriate database systems and even social networking to store and manage digital objects. A workflow management solution in such a context can be used to manage the main DLS services such as the archival and presentation services. The system services themselves can be utilized by different user groups such as content creators (e.g., 3D model creators), the content managers (e.g., archivists who manage a museum’s collections), the developers (e.g., web developers who build and maintain a museum’s website) and the users who use the system to view and explore the objects on display.

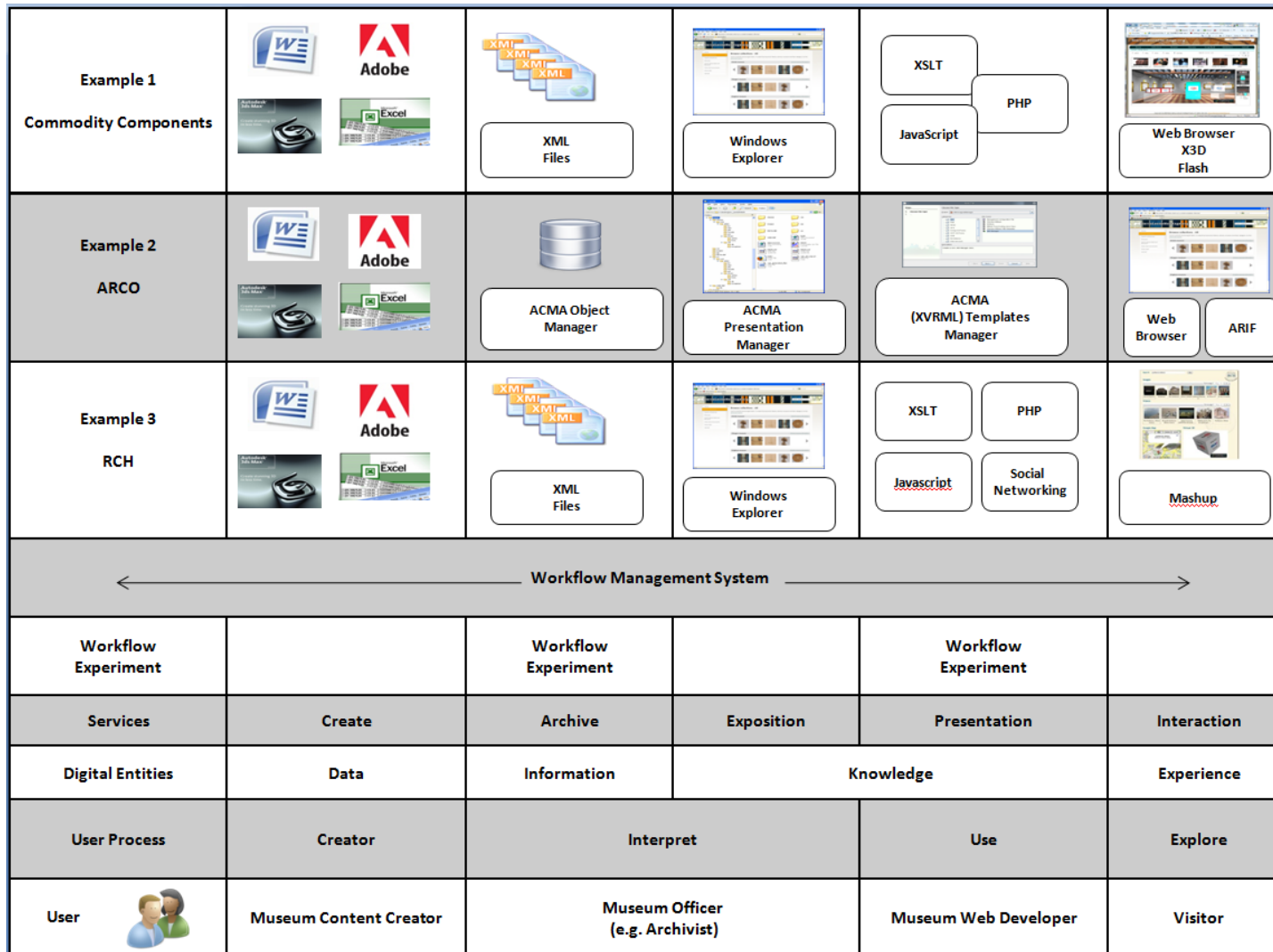


Figure 3-1 Examples of DISPLAYS Type Implementations

The DISPLAYS framework has been devised as a generic model that is a modular system based on a service orientation (SO) paradigm. It should exploit the latest advancements in commodity components (preferably open-source or low cost), networking and web technologies to allow for the creation of highly customizable online digital repositories and DLS implementations. The DISPLAYS framework essentially proposes a heritage-oriented model based on the concept of integrating five core services that map to digital entities, which in turn map to user process as illustrated in Figure 3-1. For example, a user creates digital object data with ‘create service’, using tools such as 3ds Max to create a 3D representation of the physical object. In this case, the user could be, for example, a museum digital content creator. Another user could be a museum visitor who explores the digital object through an interactive digital experience. Thus, these example implementations represent a heritage data model that benefits from the DISPLAYS services: these are discussed further in Section 3.4.

As implied above, the framework presented by DISPLAYS has the flexibility of being implemented in a number of different ways. Another example is where all the system services can be used to construct a total DLS solution built from scratch as is the case with the RCH system [1]. Alternatively, some of its services, such as the DCC or DCA for instance, can be used to complement current DLS implementations that may need such advanced functionality. This process was made possible by the modular nature of DISPLAYS due to the SO approach that it adopts, as will be detailed in Section 3.5 of this chapter.

Another area that the DISPLAYS framework empowers is the use of social networking technologies such as Facebook, YouTube and Flickr, to create customized Web 2.0 information mashups. These mashups are used to present the managed digital objects in more meaningful ways to their communities of practice [1]. Additionally, integration with social networking platforms is also used to enable the collection and management of user-generated contents that are used to enhance the managed collections. Again, RCH is a good example of a system that makes use of such services and this will be detailed in Chapters 4 and 5 of this thesis.

3.3 DISPLAYS Objectives

The main objectives of the DISPLAYS framework span a number of areas that revolve around the idea of enabling effective sharing and distribution of digital cultural objects.

The first of these objectives is to develop a set of services for a DLS, i.e. archival, exposition, presentation and interaction services [7]. DISPLAYS services are meant to have the flexibility to handle digital objects that may span a number of different formats and come from a variety of sources.

As another objective, DISPLAYS aims at empowering its communities of practice to create, interpret and use their own digitized cultural objects by utilizing the possibilities offered by multimedia, 3D, virtual and augmented realities [1]. This objective aims at paving the way for enhancing learning and data sharing through the exploitation of advanced innovative technologies, e.g. social networking, within an appropriate architectural framework. This objective can be achieved through the development of a set of DCC services that are designed to empower the communities of practice to create and maintain their own digitized collections. Such collections may involve the creation of complex multimedia-oriented data and 3D models to suit the different needs that a community of practice might have. For example, communities would submit their user-generated contents to the DLS via a social network service, Facebook for instance, using a Facebook web service (service orientation).

Based on these two objectives, the DISPLAYS framework needs to have effective digital object creation and archival capabilities. DISPLAYS DCA services aim at allowing users to easily index, retrieve and manipulate digitized cultural objects data within a central digital library service, while emphasizing on optimal digital preservation [7]. Content creation is complemented with the development of DCC services [7] that enable participating communities of practice to distribute their own multimedia-based heritage experiences to the interested bodies such as local cultural institutions. Furthermore, the DISPLAYS DCP and DCI services are there to enable the users of the devised DLS to visualize their own multimedia heritage objects within multi-lingual and multi-disciplinary contexts, by means of exploiting various visualization techniques including virtual, mixed and augmented realities on the web [5].

There are many other services that are of pivotal importance to a DISPLAYS type system, such as the creation of a set of Digital Library Management Services (DLMS) that include a number of custom designed components. These components include rights description and management schemes, identity, trust and security services, discovery services that use semantic descriptions (of data and application) and file

system management services. These services are used to provide advanced capabilities in terms of the distribution and storage of digital Portable Antiquity repositories in enterprise environments. However, this is beyond the scope of this thesis; instead the discussion of DISPLAYS is limited to that necessary to set the contributions of this thesis in context.

3.4 DISPLAYS Services

The operation of DISPLAYS is based on a number of specialized modular services that serve different purposes within the developed DLS model. Each service is intended to be a component that is accessible via a web service (e.g. the ARCO ACMA Cultural Object Manager and Presentation Manager are based on a SOAP web service to access the ARCO database (see Section 2.1.6.6), while most typical social networking APIs use a RESTful web service to provide access to their API functionality). These services may interact with the other modular system components to achieve the required functionality. What follows is an outline of each of the DISPLAYS services.

Figure 3-2 illustrates the DISPLAYS services (DCC, DCA, DCE, DCP and DCI) alongside the tools and processes associated with each of them. The focus of the work of the author of this thesis is related to workflow management of data-intensive operations within the DISPLAYS Framework.

The bottom row of Figure 3-2 highlights the key areas that the workflow management application created as a part of this thesis aimed to manage (see Chapter 6 for more information about the created WfMS). These areas of focus are mainly concerned with the data flow between the DISPLAYS Framework's components within its practical implementations (Such as the RCH prototype, see Chapters 5).

The sequence of green arrows at the bottom of Figure 3-2 highlights the key areas where the workflow management system implementation presented in this thesis gets involved. These areas are related to data-intensive operations by nature starting with content rendering through to digital content archiving, digital content exposition via XML files, query processing within retrieval services and digital content presentation through online web interfaces. It was decided to focus on these areas as they present a number of the most challenging use-cases to manage within a DLS.

The DISPLAYS services are further detailed in Sections 3.4.1, 3.4.2, 3.4.3, 3.4.4 and 3.4.5 respectively.

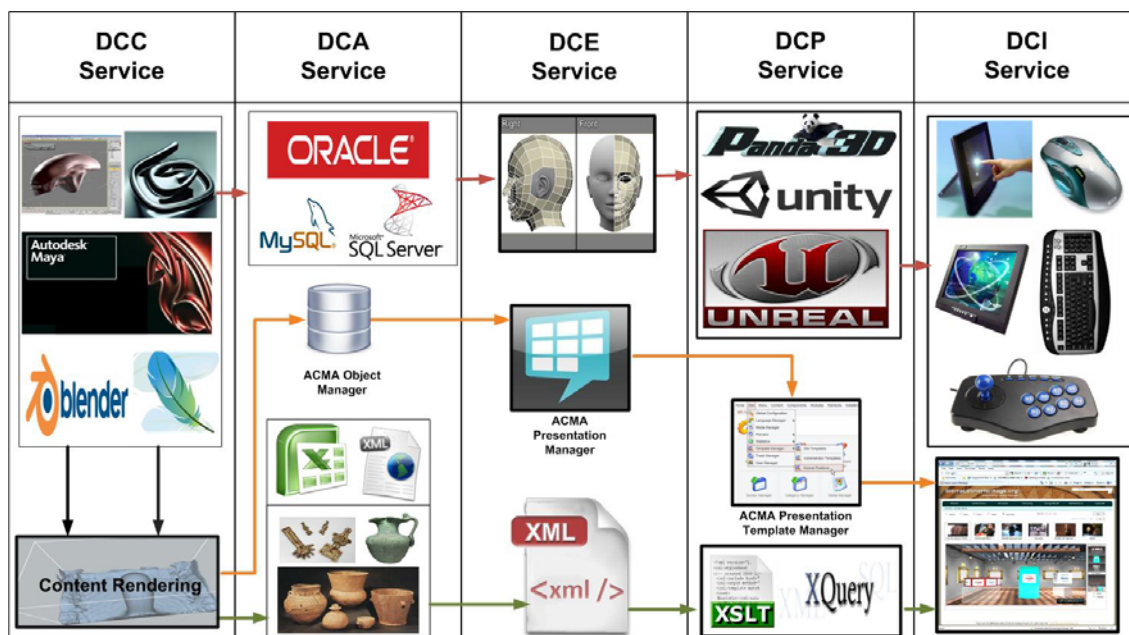


Figure 3-2 DISPLAYS Services and Their Associated Tools

3.4.1 DISPLAYS Digital Content Creation (DCC) Services

The concept of building specialized modules for digital object creation is one that is commonly used in DLSs, and it greatly contributes to building unique and sustainable digital repositories [109]. The digital objects that usually circulate around a DLS tend to be rich and varied in terms of their formats, and the data that accompany them, which consequently necessitates the existence of the appropriate tools to create and manage them. Furthermore, Radoslav and Pavlov [110] indicate that it is a key feature in modern DLS implementations to have powerful as well as flexible content creation services. This is because they play a key role in enhancing the digital collections being held by the shared resource in use: this applies to the model adopted by DISPLAYS.

Effective content creation is particularly important in the case of DISPLAYS due to its interaction with cultural objects that are rich by nature and require content creation services that are able to adequately model and store them. DISPLAYS DCC services are considered to be among the fundamental services in the DISPLAYS framework as they form the main source through which the system's digital collections are built. DCC services contribute to building the collections of the created DLS, regardless of the digital object formats used by content creators. When talking about DISPLAYS DCC services, it can be observed that they cover the full workflow of digital content from

creation to interpretation to use and exploration through the DISPLAYS operational model [5].

According to Patoli *et al* [5], the created services deal with DISPLAYS content creation needs within three broad major categories that are designed to cover the range of media that the built system is likely to handle. These services are as follows:

- Document Creation Services
- Video and Image Creation Services
- 3D Graphics Creation Services

Therefore, the DCC services contain a number of sub-services that are dedicated to the creation of digital objects of different types and standards. The created objects can be sourced from either non-digitized contents, such as normal books, or contents that are originally in digital format but converted to other formats that suit the ones used by the concerned community of practice.

Data indexing and retrieval is maintained through the assignment of appropriate metadata to accompany the created contents: this serves the needs of the communities of practice to meet their long term data preservation requirements. This process adheres to the common standards used in contemporary DLS implementations, as indicated by the principles outlined in the ‘Framework of Guidance for Building Good Digital Collections’ as set out by NISO [111] stating “collections should be described so that a user can discover characteristics of the collection, including scope, format, restrictions on access, ownership, and any information significant for determining the collection’s authenticity, integrity, and interpretation”. The importance of maintaining effective content preservation and retrieval practices is further highlighted by NRGL [112] that states “various metadata formats are important for indexing, upload of content, making it accessible and content protection”.

Figure 3-3 illustrates some of the tools that may be used in content creation. These tools may include 3D model creation tools such as 3ds Max, 3D animation creation tools such as Blender, image editing software such as Adobe Photoshop, etc. The contents created by these tools are rendered so that they can be appropriately handled by the DCA services.

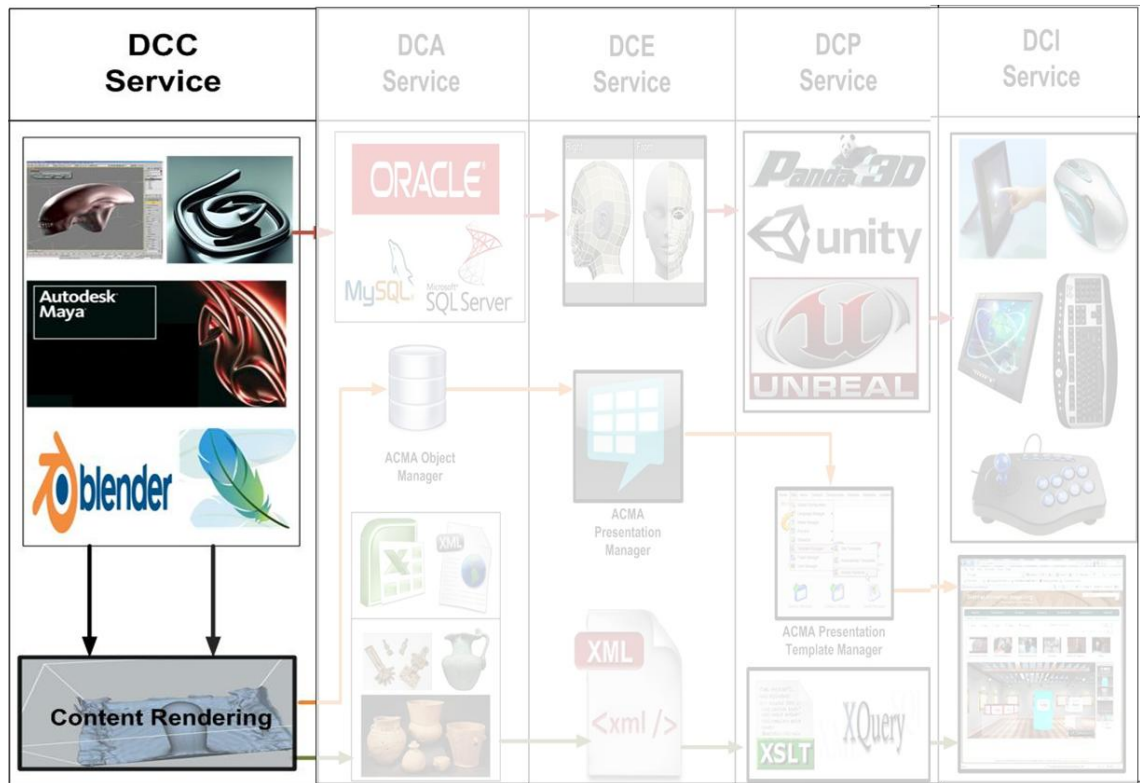


Figure 3-3 DISPLAYS Digital Content Creation Services (DCC)

3.4.2 DISPLAYS Digital Content Archival (DCA) Services

Archival services play an important role in digital resources and DLS implementations, as these systems are actually considered to be the natural progress from the traditional forms of digital archives [113]. Archival services have always been an integral part of DLS implementations, as explained in Section 2.1.3.4, due to the key role they play in managing the collections held in digital repositories. Likewise, DISPLAYS heavily depends on an advanced archival services that are utilized towards the goals of providing advanced digital object archiving and management capabilities.

The DISPLAYS DCA services primarily work on providing DLS users with a set of archival application tools that can be used to archive the digital contents being used in any given context. This process covers the different data types and media that an organization is likely to use, such as digitized documents, audio, video, images and 3D graphics. Furthermore, the underlying DISPLAYS DCA services are primarily aimed at facilitating the storage and management of the contents created by the DCC services, paving the way for re-using them by the system's communities of practice [5].

The functionality of the DISPLAYS DCA services revolve around the processes of inserting data (create), getting existing data (retrieve), modifying existing data (update)

and deleting data (delete), so called CRUD operations, which directly interact with the system's digital heritage objects [7]. So, in the context of a DISPLAYS implementation the CRUD operations for archiving a digital heritage object would require interfaces for:

1. Inserting a digital object into the DISPLAYS repository

- Examples are the ARCO ACMA Cultural Object Manager, or an Excel XML generator to create an XML file for a flat file repository.

2. Modifying existing data in the DISPLAYS repository

- Examples are the ARCO ACMA Cultural Object Manager, or the Excel interface to an XML file.

3. Retrieving digital objects from the DISPLAYS repository

- Examples are the ARCO ACMA Cultural Object Manager, or a website that queries XML files through XSLT files.

4. Deleting digital objects from the DISPLAYS repository

- Examples are the ARCO ACMA Cultural Object Manager, or a website that enables the deletion of objects' data by sending SQL delete commands to a database that holds the managed objects.

DCA services are ideally customizable by the communities of practice so that they can be adapted to suit their evolving needs. An example of this would be to build social networking functionality such as comments boxes, and photo and video upload, etc., into the DLS [1]. More specifically, an existing digital object in a DISPLAYS system could be modified using the Facebook comment API to add a comment to the digital object record by a visitor; this is actually implemented in the current version of RCH described in Chapter 4 of this thesis. More importantly, the DISPLAYS DCA services have built-in metadata, ontology and resource discovery standards to facilitate the data preservation and retrieval operations by means of the effective interpretation of the metadata introduced by the DCC services [7]. DISPLAYS implementations can utilize a variety of metadata standards, for example the RCH implementation uses the Dublin Core standard. DCA services are further illustrated in Figure 3-4. Again, a typical DCA service may be associated with a variety of archiving tools such as relational database packages (SQL Server, MySQL, Oracle, etc.). The ACMA Object Manager is used in

this instance as the management medium within the DCA (see Section 2.1.6.6). Other simpler (lite) archiving methods can also be used, such as Excel and XML files.

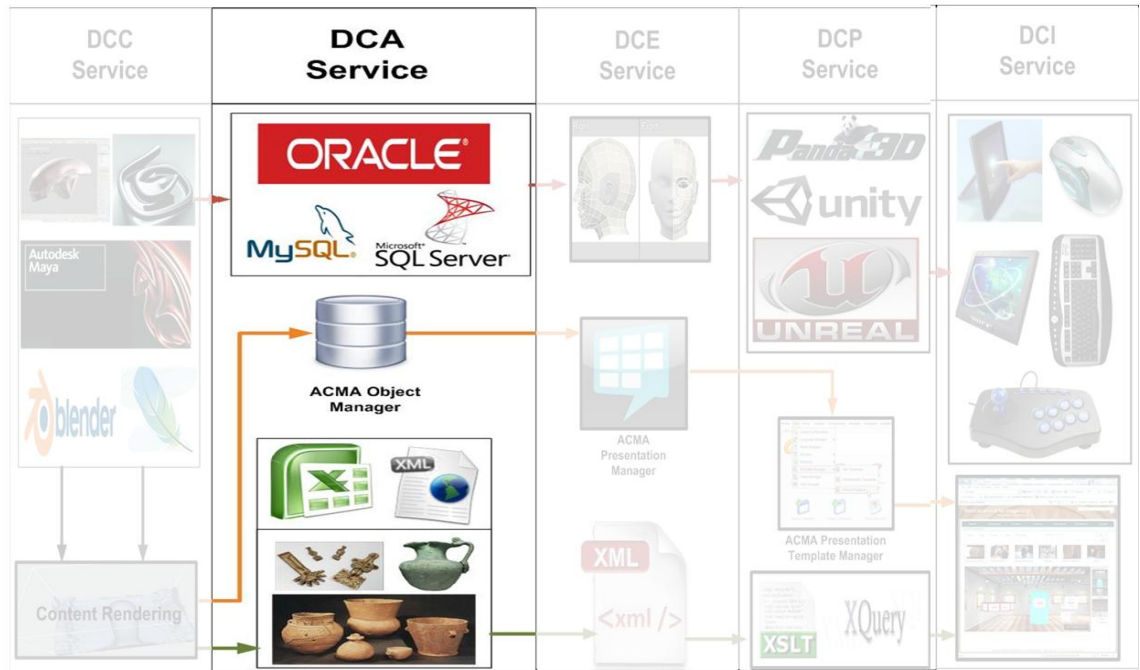


Figure 3-4 DISPLAYS Digital Content Archival (DCA) Services

3.4.3 DISPLAYS Digital Content Exposition (DCE) Services

The utilization of virtual digital object expositions is a common notion that accompanies cultural heritage applications due to the need to virtually expose the managed collections in more meaningful ways [114]. In view of this, a project with a similar implementation that can be mentioned in this context is the Biblioteca Universalis Digital Library Project [115], which aims to “make the major works of the world’s scientific and cultural heritage accessible to a vast public via multimedia technologies” [115]. DISPLAYS virtually shares the same above object exposition goals, and this is reflected in the range of exposition functionalities that it provides its DLS implementations with.

DISPLAYS DCE services represent a set of pivotal services that heavily interact with heritage objects that are rich by nature. Moreover, the DCE services are utilized towards the goal of providing highly customizable heritage object virtual exposition services that can be manipulated in a number of ways by the system users, especially in relation to the process of building virtual museums – ARCO for example has a major goal to provide DCE services that allow museums to create virtual museums. Another aspect that highlights the importance of DISPLAYS DCE services is the fact that they act as

agents to create knowledge out of the underlying cultural heritage context in place. That is, it is the DCE services that museum professionals use to capture their interpretation of the managed digital objects via themes, exhibitions or virtual museums (online). Thus, DISPLAYS DCE services allow the communities of practice to build and publish their own custom-made multimedia-based heritage expositions on the networks of shared enterprise systems [116]. This fulfils one of DISPLAYS principle functional goals, which is concerned with the objective of creating virtual expositions especially in the form of virtual museum expositions. Such expositions can then be used and explored through DCP and DCI services respectively by the community, thus sharing this knowledge leading to more understanding of heritage in the collections [116].

The DISPLAYS DCE services can be based on template technologies [7] in association with Extensible Markup Language (XML) based languages such as X-VRML, VR-BML and VRML/X3D [1]. This variety of formats aims to allow for the correct presentation of the handled heritage objects. The functionality of the DCE services is complemented with the DCP and DCI services as discussed in Sections 3.4.4 and 3.4.5. The DCE services may comprise a number of content exposition tools such as 3D exhibition creators as shown in Figure 3-5. The ACMA Presentation Manager is utilized here (as an example) to present the results of the exposition process to the targeted end-users.

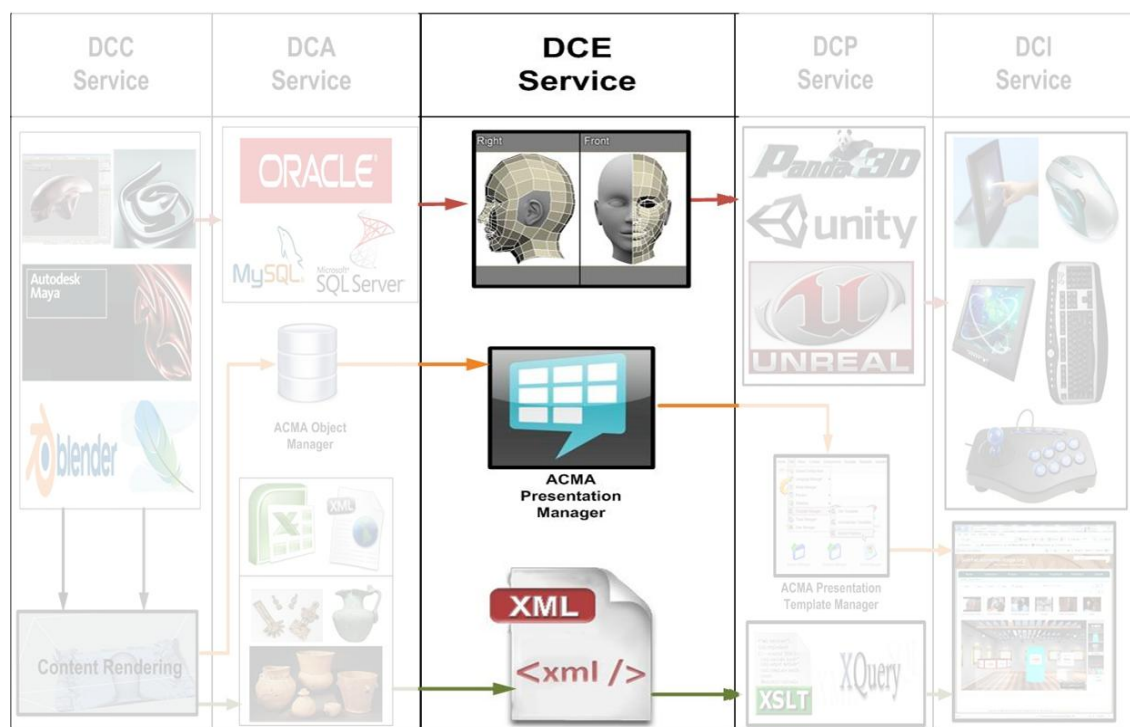


Figure 3-5 DISPLAYS Digital Content Exposition (DCE) Services

3.4.4 Digital Content Presentation Services (DCP)

Presentation needs in a DLS are notably addressed in a number of similar systems and implementation models, such as the DELOS Digital Library Reference Model [117] that was discussed in Section 2.1.5.2. DELOS provides its users with a personalized object browsing experience by exploiting proprietary information visualization via some custom devised novel user interfaces. Another example is the ARCO Presentation Manager, which is typically integrated into a museum's website [65].

Avis *et al* [118] define the process of digital object presentation as an interdisciplinary activity that involves the use of computing resources to visualize data objects. In a DLS context, the object presentation process involves visualizing the handled objects into formats that can be viewed by the system users. One of the main motivations behind the development of the DISPLAY DCP services is to address a long-standing problem in digital heritage contexts, which is the lack of coordinated digital resources and tools to “access, analyze and visualize archaeological data for research and publication”, Pettersen [116].

DISPLAY DCP services provide highly innovative application tools that have the capability of enabling the communities of practice to effectively visualize their own heritage objects within multi-lingual and multi-disciplinary contexts. This process is achieved by using different techniques including virtual, mixed and augmented realities on the web [7].

Figure 3-6 illustrates a typical DISPLAYS DCP service. A number of tools are used to present the results of the DCE services. Such tools include, for example, Panda3D for 3D object presentation. Another example is the ACMA Presentation Template Manager that can be used to define the layout of the virtual museums to be used in conjunction with the DCI services.

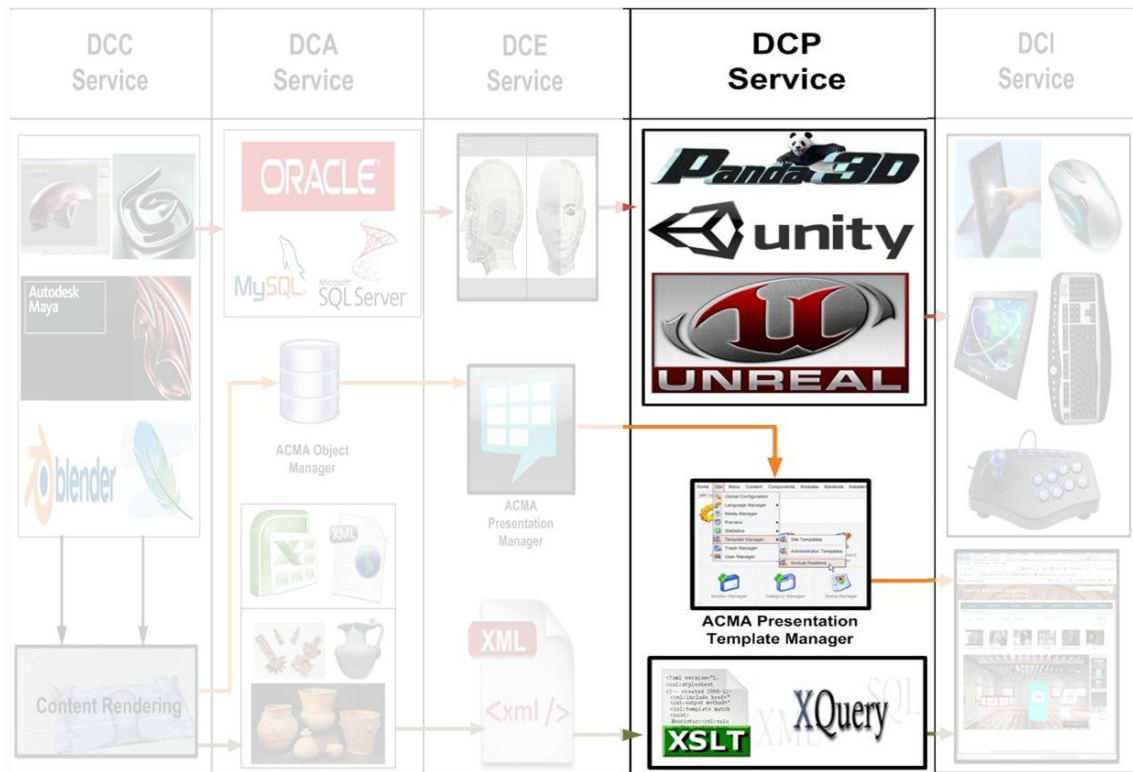


Figure 3-6 Digital Content Presentation Services (DCP)

3.4.5 Digital Content Interaction Services (DCI)

The importance of effective digital object interaction services in the context of shared DLs is discussed by Clubb *et al* [107] who state “institutions are now looking at outreach and other ways to be more relevant to their communities and their customers’ daily lives. The focus is now on the experience, both real and virtual, of the institution itself, as well as the institution’s collections”.

The DISPLAYS DCI services empower the communities of practice to interact with and manipulate their own objects by using some innovative multimedia interfaces, such as gaming interfaces that add a ‘user interactivity’ dimension to the managed cultural heritage object collections [7]. DISPLAYS DCI services support a variety of interaction modes such as website interfaces, touch screens, 3D Models, etc., within a variety of environments (Intranet, the Internet, computer kiosks, etc.).

Figure 3-7 illustrates a typical DISPLAYS DCI service where a number of mediums can be used by the end-users to interact with the displayed contents. For example, users can interact with virtual museums by using touch screens and joysticks. Alternatively, contents can be displayed in interactive websites that comprise interactive 3D virtual museums for instance.

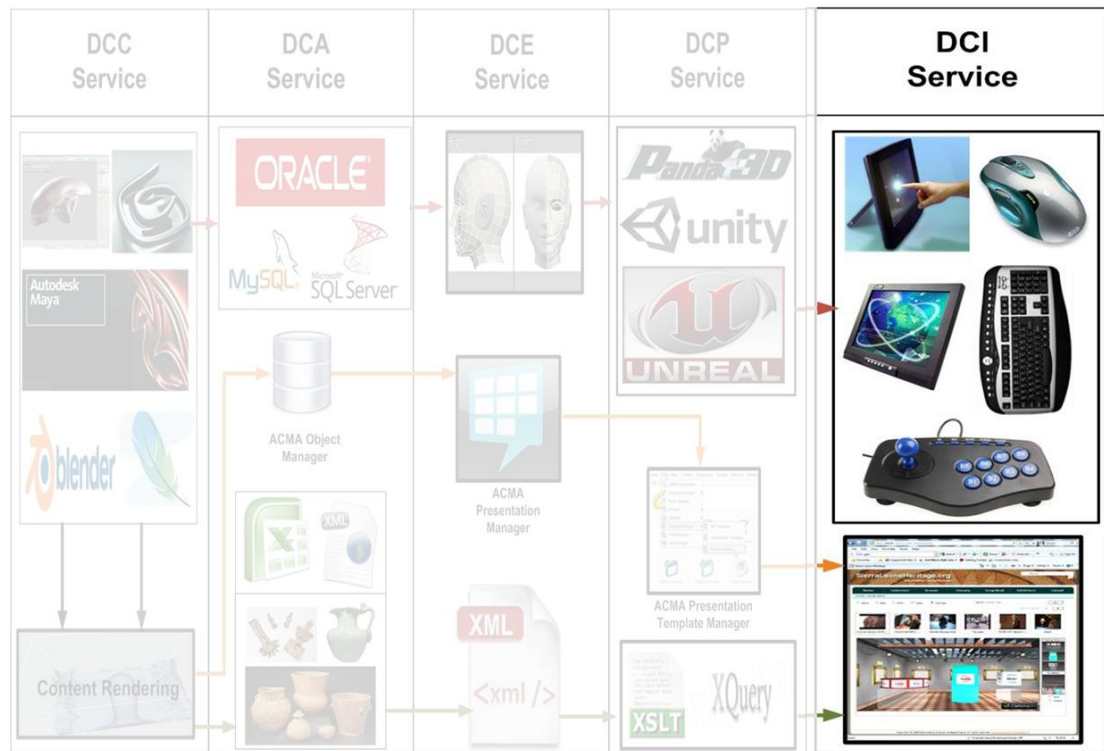


Figure 3-7 Digital Content Interaction Services (DCI)

3.5 The DISPLAYS Architecture

The DISPLAYS framework is based on a service orientation (SO) approach where the different system services are provided by highly customizable independent code modules, or off-the-shelf components, or any combination in-between [1]. This loosely-coupled approach paves the way for unlimited opportunities in terms of the services that a DISPLAYS based system can provide for its communities of practice, as stated by Patoli *et al* [1] “by using an SO approach it is envisaged that each element of the heritage data model can be created with either specifically designed components or existing components that are distributed and whose functionalities can be accessed through web services to create heritage applications”.

The adopted paradigm has manifested itself in the DISPLAYS services, as explained in Section 3.4. For example, the DCA services have the sole role of providing innovative and customizable archival facilities for the communities of practice while being flexible enough to meet their individual needs. In a typical DISPLAYS DLS, the required functionality (in this example DCA services) can be achieved via a web services model that is consumable by the communities of practice through a specific interface for that service. More concretely, a DCA service can be a social network component such as a Facebook photo or video storage and retrieval mechanism. In this example, by

definition, a DISPLAYS implementation is distributed – this has actually been implemented in the latest RCH system (see Chapter 4).

The very nature of DISPLAYS, and its aims and objectives, makes it necessary to adopt a distributed implementation approach, where interaction between the system services can be complex at times, especially when it comes to the process of handling the data and control flows of the system [5]. In an ideal world, a DISPLAYS implementation would make use of semantic web services, together with associated services such as semantic service descriptions [7], service level agreements, workflow and orchestration, management, trust and security, and transport and messaging. However, this is beyond the scope of this thesis, which focuses on showing how workflow management can be applied to DLS (see Chapter 6).

However DISPLAYS sets up its services, a community of practice, such as a museum, should be able to store and retrieve the information of the artefacts that it possesses, including their textual descriptions, photographs, videos, 3D models and audio clips, by utilizing the DISPLAYS DCC and DCA services. DISPLAYS can utilize several different architectures to implement its storage mechanism. For example, DISPLAYS data could be stored in a distributed implementation based on the use of web services to connect the distributed components, as implied in the discussion above [7]. Another method could be to adopt a data Grid approach. This paradigm can be observed in some similar ventures such as the work done by Larson *et al* [119], which involved the creation of a Grid-based DLS with advanced distributed archival and retrieval capabilities. As the system's (any DISPLAYS based DLS) archive builds up over time, the data gets organized into more meaningful information that is beneficial to the communities of practice for the purposes of digital object preservation and re-use.

The DISPLAYS components themselves can be thought of as being built in a tiered manner, where the different system modules are grouped in separate layers [7] as shown in Figure 3-8. In this paradigm, there are two main layers, which are the application layer and the SOA layer. The application layer provides the actual DISPLAYS user interfaces (for example, the Facebook interface mentioned above) that directly interact with the DLS users to achieve a variety of functions, such as content creation and archiving (of photos and videos). The SOA layer comprises the main system's web services that act as functional components providing a number of specialized services such as workflow management, security framework, database file systems, etc. The

DISPLAYS repositories are accessed through service orientation, and this approach has the flexibility of adding more repositories if the need for that arises.

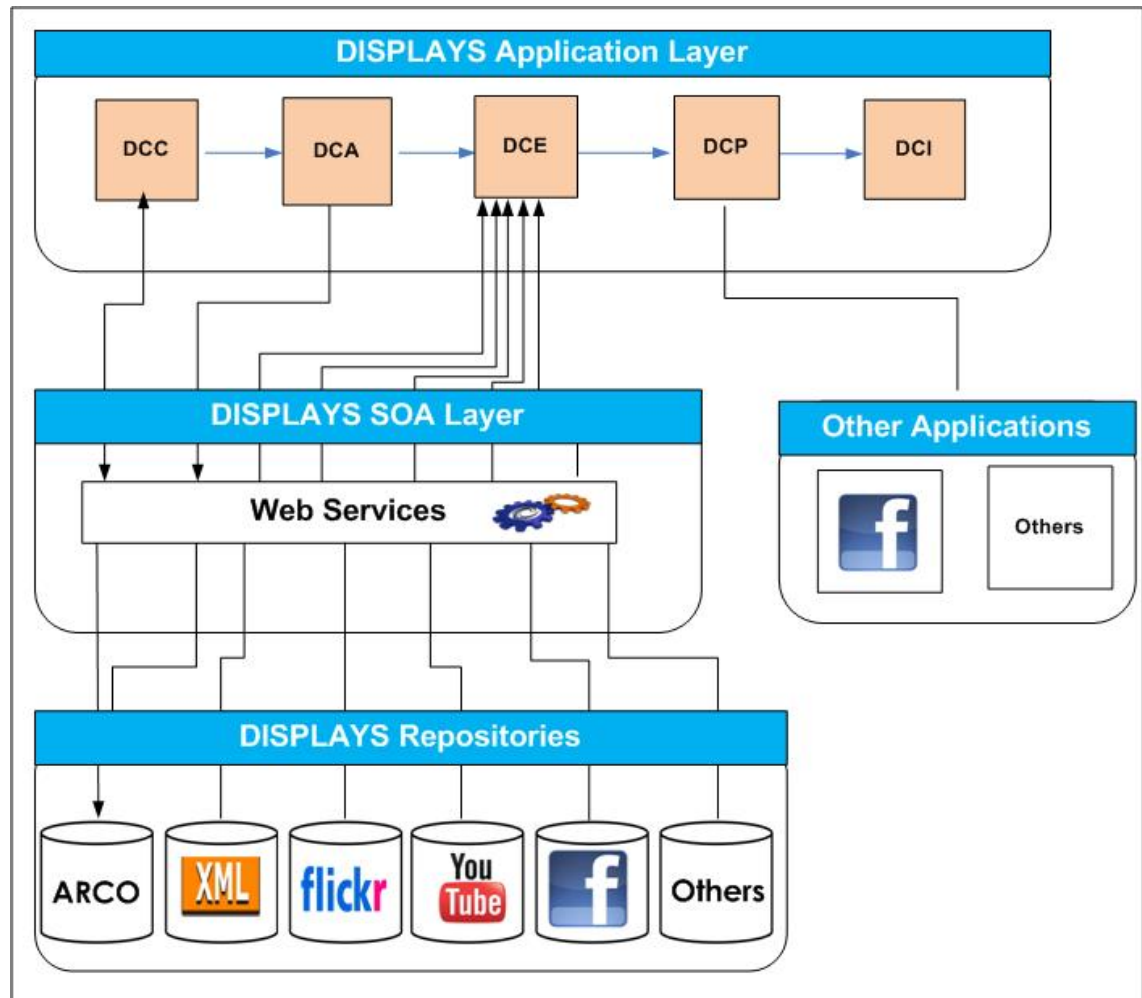


Figure 3-8 DISPLAYS Architectural Components

A DISPLAYS implementation is envisaged as being distributed in nature, thus the layers illustrated in Figure 3-8 above imply that these components can be distributed. DISPLAYS could also be implemented using a P2P Grid approach for storage and retrieval of data [113] [91].

A more specific example where Grid based services could be useful in a DISPLAYS implementation, which is distinct from the concept of implementing DISPLAYS as Grid architecture itself, is the use of a Grid to build a rendering facility [16]. Here, the computing resources of the DISPLAYS system users may work collectively to produce complex 3D models, as will be further showcased in the DISPLAYS operational scenario in Section 3.6.

3.5.1 DISPLAYS Grid Infrastructure

Grid computing has been identified as a significant emerging technology that has been rapidly introduced into software systems [120]. Computer Grids help virtual organizations, where resources are geographically and physically distributed, to tackle complex computational challenges by means of integrating efforts and resources that are available within the enterprise. Moreover, Grid-based implementations particularly suit virtual systems where intensive data sharing and modelling functions are required within an enterprise that spans multiple actors who possess different computational powers.

The nature of the DISPLAYS framework requires the existence of efficient and cost-effective techniques to serve the data processing and visualization needs of the system. This is further necessitated by the fact that the system's communities of practice often use complex multimedia-based objects, such as 3D and animated models, that require adequate computing resources to render and process. The use of Grid-based technologies meets the DISPLAYS framework needs and requirements as it provides efficient computational, storage and access control capabilities that are much needed by the system services.

Grid solutions perfectly suit the adopted DISPLAYS SOA approach where there is a need for Distributed Resource Management in conjunction with the operation of the different system services and workflows. For example, the system resources can be pooled to handle computationally strong problems, such as 3D modelling. In this scenario, custom designed Grid-based Render Farms [16] are utilized to produce the required computational power that is necessary to undertake such resource-intensive tasks. By adopting this infrastructural model, the system resources are utilized very efficiently as they can be orchestrated into achieving certain tasks within a reasonable timeframe and resource consumption.

Figure 3-9 below shows how a Grid infrastructure can integrate the DISPLAYS services with the ability to perform different tasks across the system layers. A typical Grid render farm is illustrated that can be utilized in different DISPLAYS scenarios such as the production of 3D models as mentioned above.

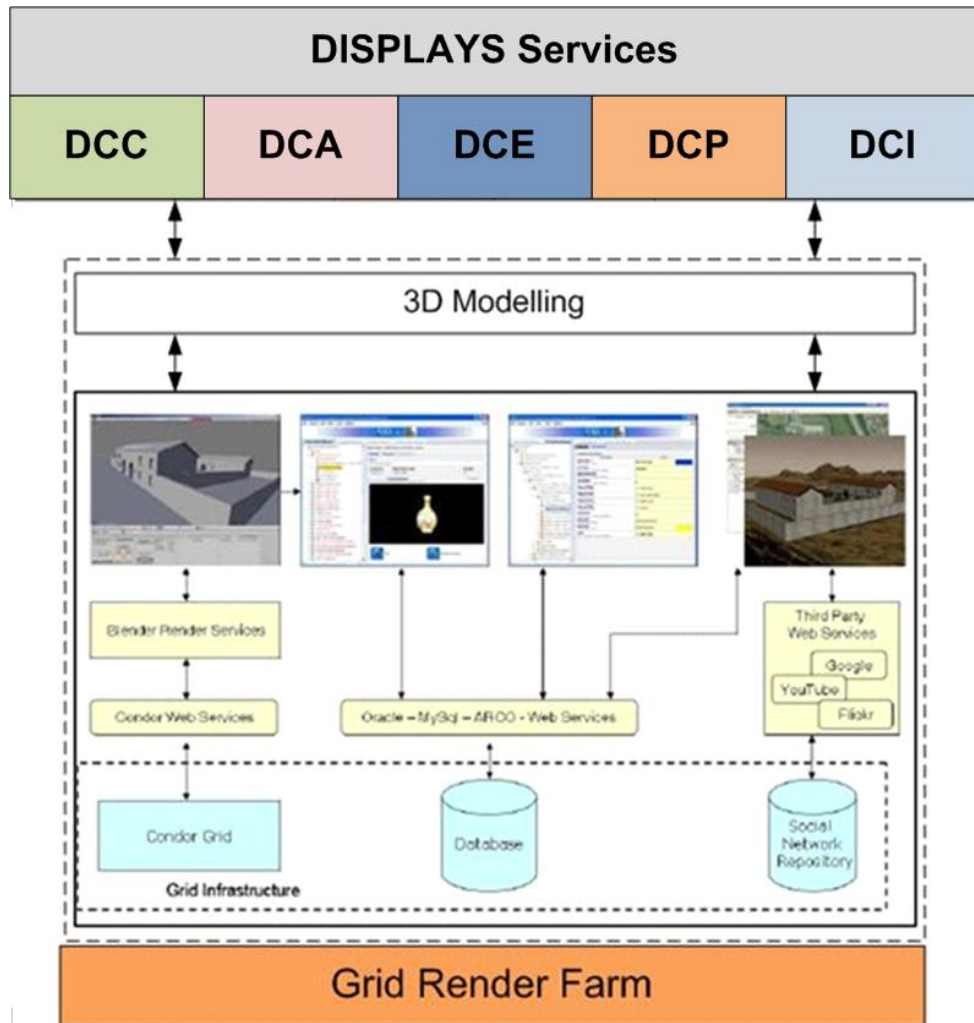


Figure 3-9 Integration of a Grid Render Farm with DISPLAYS

Going into the detail of DISPLAYS Grid implementation is outside the scope of this thesis, but its importance is further highlighted in the DISPLAYS operational scenario in Section 3.6.

3.5.2 DISPLAYS Workflow Management

Devising viable workflow solutions for the DISPLAYS framework was a major part of the research and practical work conducted as a part of this thesis. The integration of a workflow management component within the DISPLAYS architecture is based on the idea of integrating a dynamic workflow solution with a heterogeneous DLS infrastructure. This integration is aimed at providing a number of advanced workflow management and control services, as proposed by the author of this thesis in the paper entitled ‘A Dynamic Workflow Management Framework for Digital Heritage and Technology Enhanced Learning’ [5].

The DISPLAYS concept has been practically validated in the case of the RCH DLS as further outlined in Chapters 4 and 5 of this thesis, thus DISPLAYS as a concept or framework now became a practical implementation, at least in part. This practical implementation has implicit workflows that can now be investigated to see how a formal workflow management service can be applied. For this the archival, retrieval and presentation components (see Chapters 5 and 6) have been chosen to validate and test the workflow management service.

Workflow management forms an important part of the DISPLAYS Web Services layer where there was a need to effectively manage the complex data and control flows of the system. The proposed workflow solution for the DISPLAYS framework was discussed further in the paper ‘A Service-Oriented Approach for a Digital Library System focused on Portable Antiquities and Shared Heritage’ [7], which formed a part of the research contributing to the theme of this thesis. The work done as part of the above mentioned paper describes the DISPLAYS framework as a DLS implementation model that was “implemented through a set of user tools and services that are managed through a workflow management service. This workflow is a set of web services that implement data and control flow between the user tools and services and the underlying distributed application framework (e.g. Grid and P2P infrastructure)”. Such a perception makes it necessary to have a well-formed custom-made workflow management engine that is capable of orchestrating the different system services and workflows within the DISPLAYS convoluted infrastructure.

In a typical DISPLAYS scenario, workflow management starts from the point where a new object is created, through use to the point where it is actually consumed and manipulated by the system services such as the DCP and DCI services. As a result, workflow management should be taken into consideration to provide total automation and control tools by the workflow management component, leading to autonomous workflow monitoring and tracking capabilities within the adopted implementation, a Grid environment for instance.

Two major characteristics are considered in the DISPLAYS workflow management engine, which are the integration and interoperability between the application and infrastructural components of the system [5]. This means that the devised workflow management solution should integrate seamlessly within the SO components of the

system as a service layer that provides specialized workflow management and monitoring services.

In this way, the system workflows can be manipulated in different ways to achieve application-wide objectives that involve the coordination and management of the system services and code modules. Such tasks may include, for example, 3D modelling, object retrieval, visualization, etc. [9]. By adopting this paradigm, specific system workflows can be adapted to provide the system with the capability to distribute tasks and data between the internal system components, or among the resources of the users who interact with the system.

The DISPLAYS workflow management services are categorized into specialized sub-services that manage the main DISPLAYS services. These workflow management services manage all DISPLAYS services, starting from content creation and ending with the presentation and interaction services. Therefore, a custom-built workflow management framework should be integrated within the system in the form of a dynamic integrated workflow solution that is hosted within the system's Grid (assuming DISPLAYS is implemented in a Grid). This model of implementation should achieve separation between the business logic [5] and the control elements of the system in a paradigm that suits the adopted SO approach.

The process of workflow management within the DISPLAYS framework is one that works in line with all the system processes by coordinating their operations and providing the necessary tracking and control tools. Furthermore, sequential workflows [5] are used intensively within the proposed WfMS, whereby the system's actions are handled as workflow tasks that can branch into other related paths based on the pre-defined workflow rules and conditions. This was employed effectively within the RCH DLS implementation, as will be further detailed in Chapter 6 of this thesis.

An example of a typical sequence of DISPLAYS workflow management is illustrated in Figure 3-10, where a number of the functional points that workflow management is involved with are highlighted. This scenario shows a simplified typical digital heritage object retrieval process within a DISPLAYS implementation.

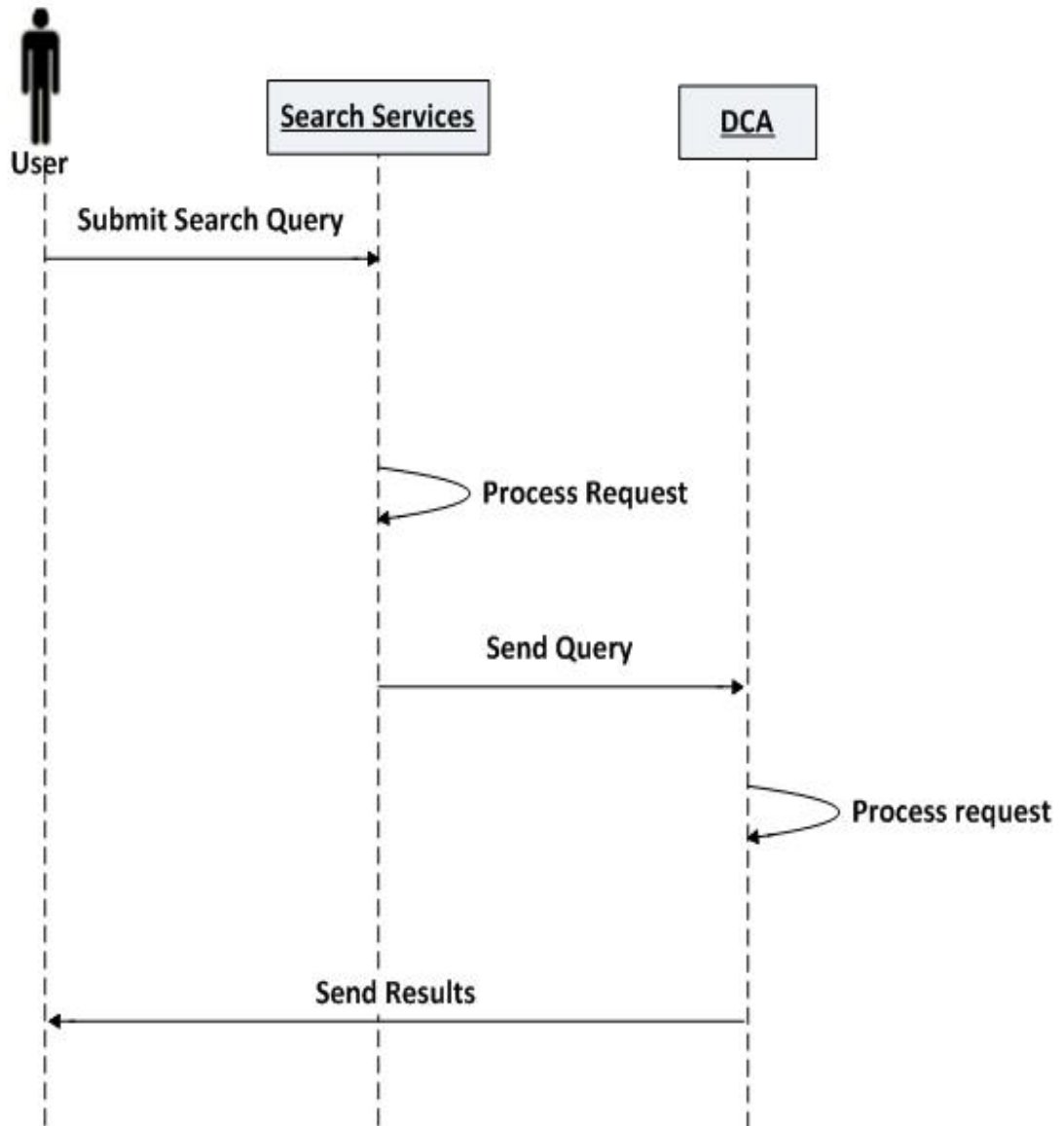


Figure 3-10 Typical DISPLAYS Workflow-managed Scenario

3.6 DISPLAYS Operational Scenario

The following scenario highlights the different operational and practical aspects of the DISPLAYS framework. It is based on the scenario whereby digital archaeological artefacts data is being shared between four museums.

3.6.1 Utilization of the DISPLAYS DCC and DCA Services

Four British museums share an interest in the cultural heritage of a certain African country. When a research archaeologist in any of the four museums finds an artefact that is related to that country, he initially records its details in a raw format at the site where it was found. Photographs are taken of the artefact and a brief textual description

is written about it. Then, when delivered to the museum, the DISPLAYS DCC and DCA Services, accessible through a web interface, are used to record and archive the details of the found artefact.

The other three museums also managed to find other artefacts in different locations in that country, and used the same archival services to record the findings of their expeditions. Any additional material, which can span a number of different digital formats such as videos, images and digital documents, is usually associated with the necessary metadata that is used for indexing and retrieval purposes. The shared collection between the four museums gets larger as time passes, as more objects are added to it on a regular basis.

3.6.2 Utilization of the DCE, DCP and DCI Services

Each museum uses the DISPLAYS DCE services to render the found objects. One of the participating museums has the necessary expertise to create accurate 3D models of the discovered artefacts, but it lacks adequate computing resources to create the actual models. This problem is resolved by utilizing the DISPLAYS Grid that provides the necessary processing power by means of combining the computing capabilities of the participating museums. DISPLAYS presents this service in the form of a shared Grid-based 3D rendering engine. The created models are then utilized in different ways in each museum. For example, one of the museums uses the DISPLAYS DCP and DCI services to display the objects on a touch screen that the museum's visitors can interact with.

One of the museums decides to run a workshop about its findings so far. The head of the workshop uses DISPLAYS object retrieval services to find all the objects that had been found over the last year. The retrieval process returns a number of objects with a variety of data formats including images, text, video and 3D models. These objects are augmented into a digital virtual museum exposition that will be an integral part of the workshop activities and discussions. The virtual museum is created by means of exploiting the DISPLAYS DCE services that provided the necessary tools to organize the pooled data into a virtual museum space that can be accessed on the web. Once the customized virtual museum space was designed, the DCP and DCI services came into play by giving it interactive features that made it highly interactive.

3.6.3 Empowerment of Content Sharing through DCA

An archaeological researcher who works in one of the museums is interested in finding objects that are similar to the ones that he found, to draw meaningful comparisons and determine their geographical distribution within the country of interest. DISPLAYS makes this possible by the techniques it uses to organize data into meaningful knowledge that is stored in highly accessible digital repositories. He views the other objects by interacting with the DISPLAYS DCP and DCI services that enable him to get an exact accurate view of the found objects, including their 3D models, images and textual descriptions. Moreover, the system enables advanced object retrieval by means of utilizing the custom-made retrieval tools that can be used by all the museums sharing the system.

An American museum starts excavation operations in the same country and asks to join the digital repository shared by the four museums. It seamlessly joins the network based on DISPLAYS Grid-based scalability that allowed for the expansion of the system with ease, paving the way for the rapid aggregation and organization of the data that come from different sources.

3.6.4 User Generated Contents

User-generated contents are collected by allowing the interested researchers and archaeologists to add their findings via a public Web 2.0 website. This website provides a number of tools that allow the users to interact with the managed collections. The provided services include the provision of highly flexible Web 2.0 mashups, and integration capabilities with social networking platforms such as Facebook, YouTube and Flickr. Such a rich web interface allows the interested users to share their experiences and interests on the one hand, and interact with the DLS objects on the other.

3.7 Summary

This chapter has described the DISPLAYS framework and its services. It was observed that the DISPLAYS framework was devised to meet a number of goals and objectives including the creation of highly accessible and sustainable digital heritage repositories. The DISPLAYS functionality is achieved based on a number of SO services that should operate on a distributed architecture, e.g. a Grid, but other solutions are possible, such

as a simple client server system for a small implementation – web services make this possible.

The DISPLAYS framework is based on five main services, which are the Digital Content: Creation, Archival, Exposition, Presentation and Interaction services, in addition to a Workflow Management service for digital heritage collections. Furthermore, the DISPLAYS framework is implemented in a layered approach that encompasses two main system layers, the application layer and the SOA layer. This allowed for total modularity within the adopted SO approach.

The modular nature of DISPLAYS makes it an easy task to customize it to work in a variety of environments and scenarios to serve different data sharing and distribution needs. It helps convert heritage data objects, that are varied in their nature and types, into meaningful knowledge that can be shared by communities of practice such as museums, digital libraries and cultural institutions.

The DISPLAYS framework utilizes social networking platforms for greater user participation and content sharing opportunities. Web 2.0 and social networking platforms are also used to produce rich information mashups that are used to present the managed collections in more meaningful ways: this is demonstrated in Chapter 4.

It is a necessary requirement to develop an appropriate workflow management solution for DISPLAYS due to the complexity of the data objects that it handles and the intersecting nature of its components. The devised dynamic integrated workflow management solution provides the necessary tools for managing the sequential heritage workflows that comprise the system's functionality. These workflows can be either data or control workflows that interact with the system services throughout its layers. The Reanimating Cultural Heritage (RCH) DLS implementation in the following chapter is a DISPLAYS-based DLS that will be used as a test bed to show that workflow management is a viable component to achieve its goals (see Chapters 5 and 6).

CHAPTER IV

4 RCH - an example DLS System

This chapter discusses the RCH system, which is a DLS implementation that effectively employs complex heritage workflows in its operation. These workflows need to be understood before attempts can be made to design effective workflow managers. This chapter therefore describes the RCH implementation in terms of its workflows. Chapter 5 then takes a more detailed look at some of these workflows: archival, retrieval, and presentation of digital heritage objects in an RCH website (a virtual museum). Chapter 6 implements this workflow in a workflow hosting environment to test the validity of workflow management in the context of DLS components.

RCH is based on the generic DISPLAYS framework that was discussed in Chapter 3. In this chapter, the services of the online digital heritage resource that RCH provides are discussed while covering the adopted architectural approach. RCH exploits various technologies, such as Web 2.0 mashups and social networking platforms, to enhance its functionality. Such functionality is accessed through web services connecting user generated contents to the RCH system. RCH is implemented as a digital heritage resource that holds the digital collections of several museums, i.e. the British Museum, the Brighton Museum and Art Gallery, and the Glasgow Museum (other museums are also adding their collections). Thus, RCH highlights the DISPLAYS concepts within a practical scenario that showcases the technical and functional capabilities of the adopted framework. RCH services are covered in more detail in Chapter 5 while focusing on their technical aspects.

Finally, the role that workflow management plays within RCH is covered. This is done while illustrating where and how workflow management should be applied to the RCH system to contribute to managing, coordinating, monitoring and tracking its complex heritage workflows.

4.1 Introduction

Contemporary museum practices tend to complement the textual information that accompanies museum collections with animated media. This association of media with descriptive text is used to display cultural heritage objects in exhibitions that closely depict the actual contexts that originally animated them [121]. Emerging software technologies and innovations, such as social networks, can provide cultural institutions and shared interest groups with new opportunities to create their own shared ‘reanimated collections’ of cultural heritage objects. This sharing of reanimated collections (i.e. museum collections integrated with user generated content) is achieved through an RCH mashup interface while exploiting social networks. Users or communities are able to connect their diverse social networks to augment data related to a specific area of interest such as a certain group of artefacts (a museum collection for instance). Additionally, the advent of interactive digital technologies, especially those that are web-based, have paved the way for the creation of virtual digital heritage environments, i.e. a virtual museum, particularly if it has 3D digital contents to better justify the term virtual. These environments can effectively reanimate cultural heritage objects in digital space as discussed in the ARCO system in Section 2.1.6.6.

The concept of ‘Reanimating Cultural Heritage Objects’ [1] represents the main focus of the RCH project where a combination of innovative technological solutions are utilized to achieve that goal. This is done through the development of a flexible and unique DLS that is capable of addressing the complex requirements of such an implementation. Therefore, RCH represents a system that provides sophisticated online heritage data repositories as a highly accessible digital resource. RCH is a DLS that adopts the DISPLAYS framework by means of a range of custom-made components that are based on the DISPLAYS SO conceptual services outlined in Section 3.4.

RCH is actually based on an Arts and Humanities Research Council (AHRC) funded project to create a digital heritage repository of Sierra Leone’s cultural heritage objects distributed through the networks of a number of international museums. In the context of this thesis, this has afforded the ideal opportunity to implement an RCH prototype (described in this chapter) to test workflow concepts. Reanimating the handled (physical) objects is done through various media such as images, videos, audio, 3D models, etc. The main objective here is to link the created digital objects with the actual oral and performative contexts that originally animated them [1]. By doing this, the gap

between the object and its original context is breached allowing for richer, more accurate, and interactive user experiences. The presented information is further complemented with social networking features and Web 2.0 mashups that contribute to adding more information to the managed objects, as illustrated in Sections 4.6 and 4.7.

Zhang *et al* [1] indicate that RCH depends heavily on a number of innovative software solutions. These solutions include Web 2.0, Library 2.0, Social Networking technologies and information mashups. The utilized technologies accompany a number of custom-made SO functional components. The end result is a number of tools and applications including Social Networking Data Repositories, Web 2.0 Dynamic Heritage Mashups, Content Management tools, Data Visualization and Presentation tools, and other customized management and tracking tools. Moreover, RCH components are managed through a custom-made dynamic integrated workflow management engine. This engine is based on the conceptual DLS workflow management model that was discussed in Section 3.5.2. RCH components and its operational scenario are discussed in the subsequent sections of this chapter as well as in Chapter 5, the WfMS solution is discussed in Chapter 6.

4.2 RCH Context

Sierra Leone is a country that has always been renowned for the diversity and vibrancy of its cultural heritage, including the music, dance, masquerade and storytelling practices of its various ethnic groups [122]. Despite its rich cultural heritage, Sierra Leone is one of the least developed countries in the world, impacted with a series of civil wars and violence that profoundly affected its economic and cultural infrastructure [123]. The RCH project operates within the context of the Sierra Leonean cultural heritage and the attempts to reanimate it by utilizing modern software technologies. It forms an important part of the research work that led to this thesis as it represents the environment in which the final proposed workflow solution was implemented, as detailed in Chapter 6. RCH has two major focus areas as indicated by Basu [123] as follows:

- to examine and validate the practical aspects of utilizing ICT innovations for the purpose of ‘reanimating cultural heritage through digital repatriation’;

- to conduct an anthropological study utilizing the resulting ‘digital heritage resource’ that the ICT solution provides to explore “knowledge networks and the strengthening of civil society in post-conflict Sierra Leone”, Basu [123].

RCH is a multidisciplinary project that involves a number of interrelated specialized disciplines including anthropology, museum studies, and informatics. Moreover, the concept of reanimating cultural heritage objects mainly revolves around the process of reanimating the objects that became totally isolated from the contexts and environments that originally animated them [119]. An important goal here is to facilitate “personalizing heritage exhibitions”, Zhang *et al* [1]. This goal aims at providing a better contextual connection between the presented cultural heritage objects and their source communities, hence enabling the provision of a more meaningful set of museum objects.

Three museums are participating in the RCH project; the British Museum, Brighton Museum and Art Gallery, and the Glasgow Museum. These museums supplied the objects that formed the digital collection that is handled by the RCH system. The participating museums used the RCH system to share and exchange their digital objects by using its data mapping and distribution tools to convert their digital objects to RCH digital objects and vice versa: this process will be detailed in Section 5.3.2. Moreover, the digital heritage resource that RCH represents has an online user interface that allowed interested users to interact with the managed objects, as will be elaborated in Section 5.4.

Figure 4-1 illustrates the concept of RCH while highlighting its functional model. RCH typically handles a variety of Sierra Leonean cultural objects that come from different disparate sources. The aggregated data belongs to a variety of media that reflects the richness of the represented objects; this includes videos, audio clips, text, and potentially 3D objects. The digital resource that is the focal point of the system is formulated by adopting the DISPLAYS framework, where service orientation is employed as the main service provision paradigm. Additionally, RCH utilizes a number of custom-made tools and social networking platforms to achieve its goals, as will be detailed in Sections 4.5, 4.6 and 4.7.

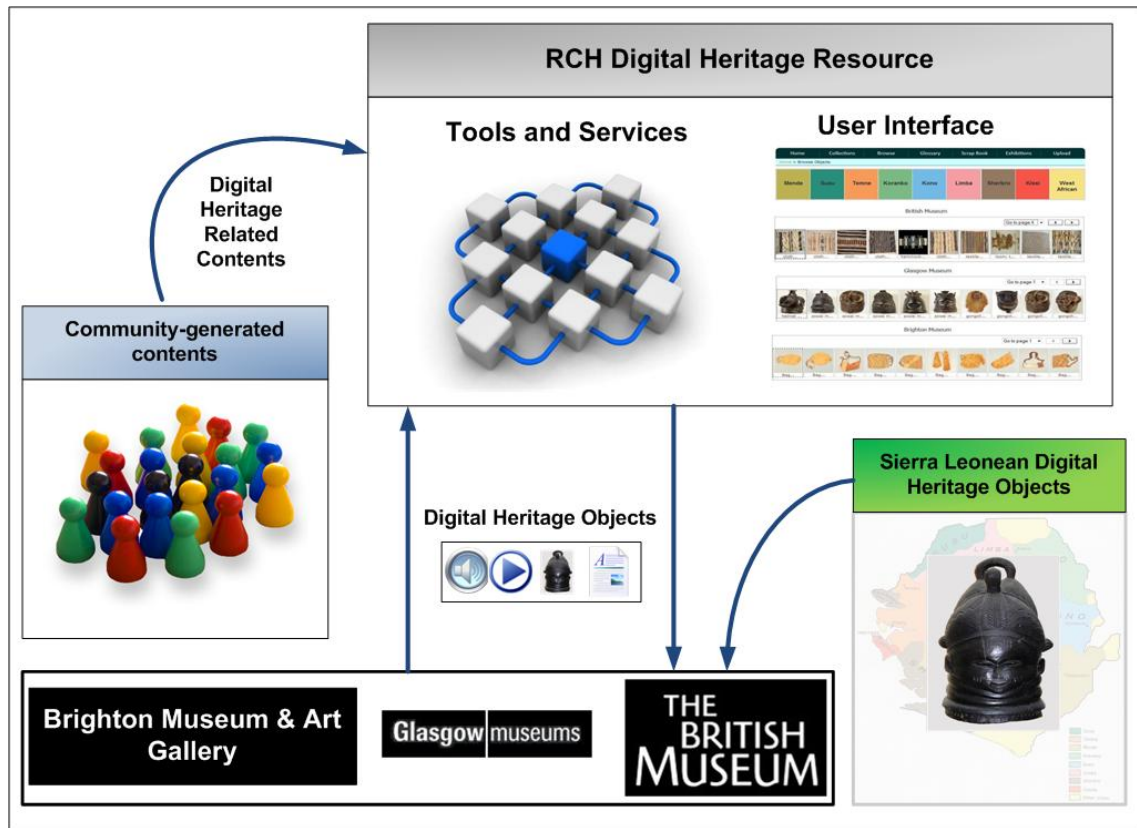


Figure 4-1 RCH Overview

4.3 RCH Objectives

The first goal of RCH is to explore and survey the latest state-of-the-art in digital museology, including the utilization of new media in cultural applications such as virtual exhibitions and repatriation applications [124]. RCH also aims to explore how the reanimation of cultural objects in digital environments can impact on ethnographic collections and their associated knowledge [1].

RCH aims to investigate the relationships that can be drawn between the elements of cultural sensory experience and knowledge/memory transmission in Sierra Leone. This investigation is meant to explore the impact of conflict and political unrest on such relationships. This goal is compounded with the need to explore the relationships between material culture, sensory experience and knowledge/memory transmission in Sierra Leone, which form the actual context of system's collections [1]. Discussion of these anthropological issues is beyond the scope of this thesis, instead this thesis focuses on the ICT needed to implement the project's goals, which is modelled on a DISPLAYS concept.

One of the important goals of RCH is to create a highly accessible digital online resource that enables the sharing of Sierra Leonean cultural heritage objects. This goal is meant to be achieved through the use of advanced web technologies such as Web 2.0, social networking platforms, information mashups, etc., to enable the creation of highly accessible knowledge networks.

4.4 The RCH Model

RCH has two broad technical areas:

- the provision of an **innovative ICT solution** for the reanimation of cultural heritage objects through the use of digital repatriation;
- the creation of an online **digital heritage resource** that can be shared among a number of users within a distributed DLS environment.

RCH's implementation is based on the DISPLAYS Framework. Therefore, its components are built as a number of specialized code modules or services based on an SO approach. DISPLAYS represented a suitable framework to meet the objectives of the RCH system. Implementation of RCH will validate the concepts, at least in part, behind the DISPLAYS framework. The DISPLAYS framework was specified to deal with complex heterogeneous DLS implementations, as was discussed in Section 3.5. Each of the DISPLAYS services is represented in a number of application tools and interfaces that supply a DISPLAYS-based system with a set of functionalities that serve its different needs. Such services can take the form of either standalone or networked components depending on the purpose for which they were devised.

In synergy with the DISPLAYS framework, the underlying RCH implementation comprises four of the five main DISPLAYS services. These are the digital heritage object: archival, exposition, presentation and interaction services. The archival services (e.g. database, Excel, XML import and export) provide the necessary tools to archive, exchange and retrieve the digital cultural heritage objects as detailed in Section 4.5.1. On the other hand, the exposition (e.g. XML data and configuration files, windows folders) and presentation (e.g. XSLT processing, JavaScript, PHP and web servers such as Apache) and interaction (e.g. the website configured as a virtual museum) services provide a variety of ways by which the RCH contents can be visualized and displayed via different technological mediums, as will be explained in Sections 4.5.2 and 4.5.2. Finally, a workflow management infrastructure is necessary to manage the system's

convoluted workflows, as will be explained in Chapter 6. The WfMS component is based on the model that was outlined in Section 3.5.2 as a part of the DISPLAYS framework.

The participating museums share an interest in Sierra Leone's cultural objects, and use RCH's tools and services as a medium for exchanging the cultural object data that they have in their collections. RCH gave these museums a common Web 2.0 online interface (a shared collection) that is accessible by various user groups. This is a practical example of a combination between the concepts of Library 2.0 and advanced distributed DLSs that were discussed in Chapter 2.

Figure 4-2 illustrates the RCH data components and workflow. It can be seen that the different RCH services are at the heart of the system as they form its functional core. These services run on the foundation of a networked distributed infrastructure that is accessible by the participating museums. The RCH services interact with the system's users through a number of custom-devised user interfaces. RCH interacts with its users mainly through a Web 2.0 website frontend that can be tailored to suit the needs of its users. Furthermore, RCH has the capability to import the data related to the digital collections held in each of the participating museums. This is achieved via a number of data mapping and distribution tools that are used to facilitate data exchange between the RCH data model and the models used in the participating museums. One of the most distinct characteristics of RCH is the innovative approach in which heritage object data representation was combined with some emerging social networking technologies and Web 2.0 mashups. This integration paves the way for a richer user interaction and a variety of ways in which the system's data can be presented, preserved, enriched and manipulated, as will be further showcased in Sections 4.6 and 4.7.

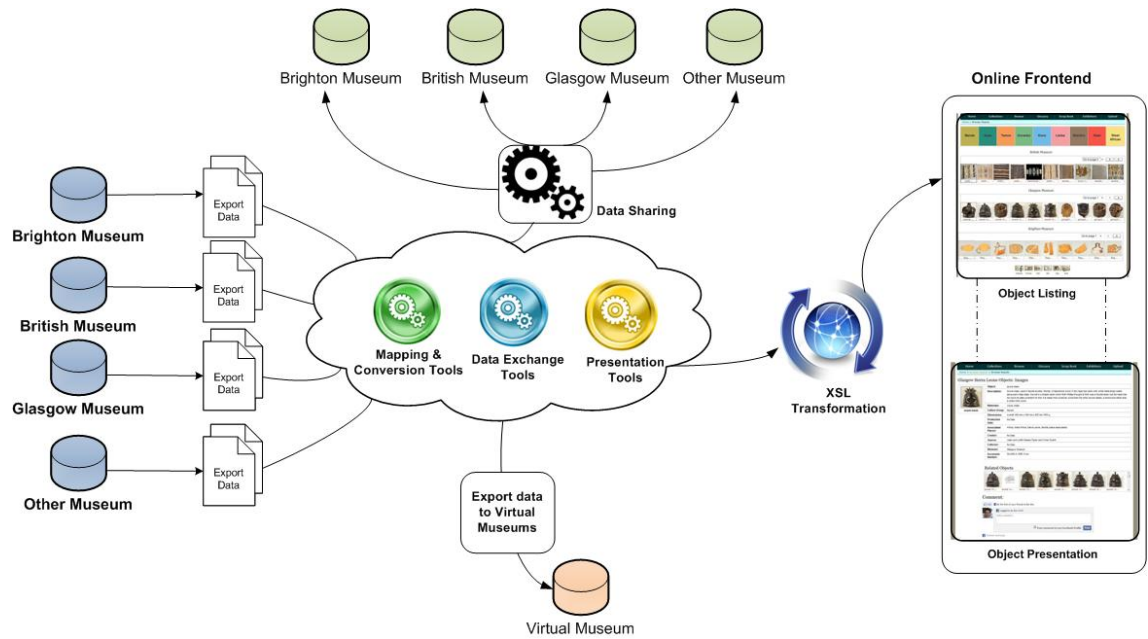


Figure 4-2 The RCH Data and Integration

4.4.1 RCH Architecture

The RCH system depends on an active database backend (XML based) due to its data-centric nature. The system's components operate on top of database services that are implemented in two main ways. An RCH implementation can have either a traditional database backend or customized XML-based data management services. An XML implementation exploits the universal nature of XML files [125], while allowing the participating cultural institutions to freely distribute and exchange their data. The utilization of XML overcomes the barriers of the various data storage formats and standards used in the participating museums. On the other hand, a traditional integrated database backend is also possible, which can act as a central data storage repository that can be used by the system users. This backend can be an SQL Server database, an MySQL database, an Oracle database, or any other database technology depending on the context of RCH's implementation. Hence RCH's generic data model can be adapted to a wide variety of data storage and management solutions that can be used by the cultural institutions that might be using its services.

Figure 4-3 illustrates the RCH architecture and its data model. An ARCO data model [65] is adopted as an example in the database-based implementation, and an XML based data model (the adopted one) is also shown. These in effect are two separate implementations with different associated services, because the database (ARCO), or the XML model, can be used to drive the data services. For example, if an XML

backend is adopted, it will consequently require the existence of appropriate XML management tools and Extensible Stylesheet Language Transformation (XSLT) files for the purposes of data retrieval and presentation in some cases. Whereas, the utilization of a traditional database model, such as a Relational Database, necessitates the existence of a set of supporting tools and technologies, such as the Structured Query Language (SQL), for the purpose of data retrieval. On the other hand, the RCH database backend interacts with the core system services and components, which in turn interact with the system's users via a number of custom-made interfaces, as illustrated in Section 4.5.2.

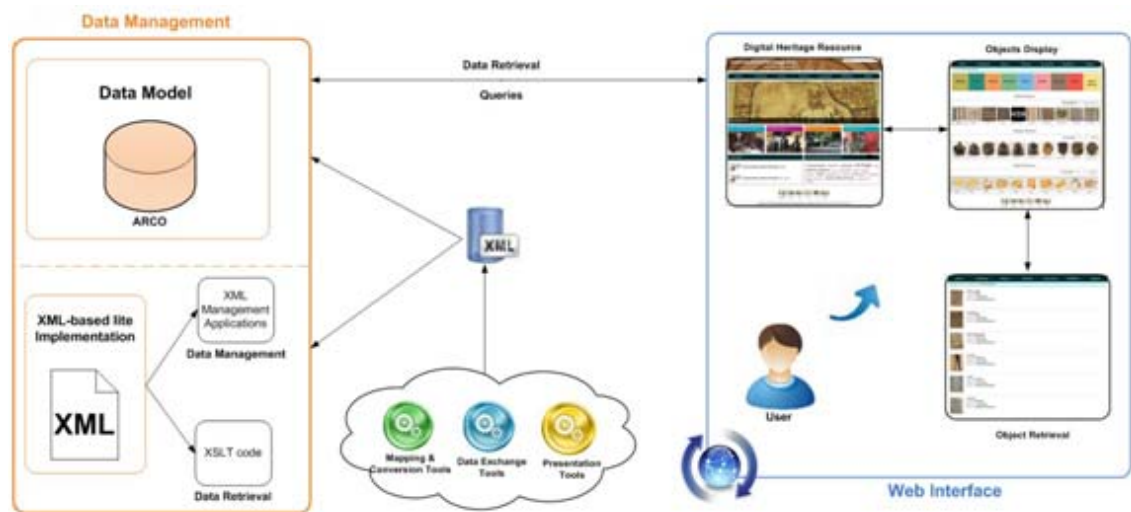


Figure 4-3 RCH Architecture

4.5 RCH Services

RCH comprises a number of services that perform different tasks according to the needs of the participating museums. The archival, presentation (and its associate retrieval services) and interaction services are among the ones that were used to prove the workflow concepts proposed in this thesis. These services were built (as actual DLS components) and then managed by the devised WfMS prototype as illustrated in Chapter 6.

4.5.1 The Archival Services

The RCH archival services consist of a number of application tools that are used by the participating museums to submit their data to the system, exchange data and retrieve cultural objects' data. RCH's archival services overcome the problem of the diversity of the data and storage formats used by the different RCH users. This is achieved by providing a set of common tools to be utilized by each museum. These tools include a

number of data mapping and conversion tools that allow for effective and standard-compliant data exchange operations, as will be explained in Section 5.3.2.

The archival services are designed as independent code modules that can fit within the existing collection management systems used in the participating museums. Consequently, two versions of the RCH archival services were created as follows:

- a **desktop** version that is DLL-based and is designed to suit users who have limited or no online access. So, this version allows them to perform data conversion operations locally;
- an **online** version that is based on web services and is designed to be distributed within RCH's enterprise to be shared by the participating museums.

4.5.2 The Presentation and Interaction Services

The RCH implementation uses web pages as the main medium for visualizing its contents including the different media types that are associated with each object. If an XML flat file system is used as the database backend, XSLT (and potentially XQuery) is utilized to dynamically present the system's objects within a template-based website frontend. The website UI is complemented with a number of additional features, such as Web 2.0 mashups and social networking functionality, to provide the RCH users with richer online experiences. Such experiences include the ability to create and preserve custom-made Web 2.0 mashups on the one hand, and the addition of community-generated contents on the other.

The presentation services are complemented with data retrieval services that aid the system users to retrieve the cultural heritage objects that they are looking for. As the data of displayed cultural objects is stored in XML files, the retrieval operations are based on the exploitation of XSLT logic to extract the required subset of cultural objects, as detailed in Section 5.5.

The RCH interaction services allow the system users to interact with the visualized objects within the system's web frontend. Such an interaction is enabled through a number of tools such as a 'drag and drop' functionality, touch screen interfaces, etc., or even augmented reality if the ARCO ARIF is deployed [65]. An RCH web interface is illustrated in Figure 4-4 where a snapshot of the presentation of the stored objects is shown. The aggregated data is displayed within a museum-specific categorization. Objects can be then more finely listed, either by browsing the different object categories

such as Mende, Susu, Temne, etc., or by using the provided advanced search functionality. The different characteristics of the RCH presentation and interaction services are discussed in Section 5.4.

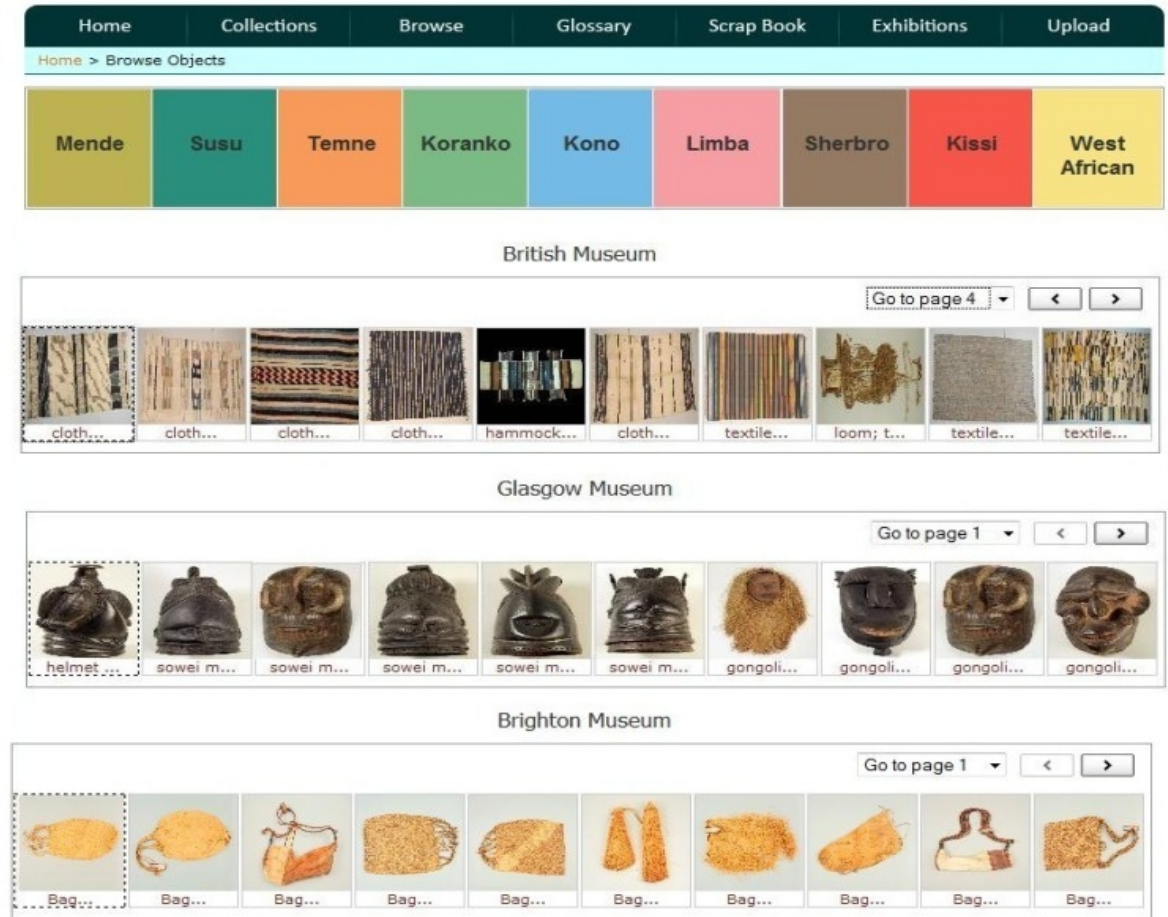


Figure 4-4 RCH's Web Interface (presentation services)

4.6 RCH Web 2.0 Mashups

The advent of Web 2.0 has transformed the web, paving the way for a number of innovative solutions that made the creation of effective data aggregation tools and applications a reality. Web 2.0 has allowed web users to effectively share contents of different media while being able to fully interact with other users who share the same interests [106]. Web 2.0 mashups are an emerging technology for the creation of dynamic data-rich web applications that aggregate open and subscribed data from a variety of sources, such as YouTube for videos, Flickr for images, Facebook for social contents, etc. [1]. Such rich contents can complement the contents or themes of knowledge-oriented websites by providing them with related contents that add more information to them [1].

RCH exploits the opportunities offered by Web 2.0 by the creative mix between the data it processes and enterprise Web 2.0 mashups. Integration of Web 2.0 mashups aims to enhance the displayed objects by means of pooling related data from different online sources. This is done to present the managed objects in custom-designed virtual museums or exhibitions that can be manipulated and preserved in different ways. Furthermore, integration of Web 2.0 mashups also allowed the end-users to build their own data-expositions that can be used for presentation and preservation purposes.

Coupling the museums' rich Sierra Leone's heritage objects with user generated contents in Web 2.0 mashups, contributes to the creation of unique online experiences within the created distributed online heritage resource (virtual museum). The resulting Web 2.0 mashups are not meant to be a replacement for the original museum contents; rather they are used to enhance them by giving the RCH users better tools to group museum and user generated contents together. In such a virtual museum, museum and user generated contents are aggregated into a single meaningful interface that can be used for different purposes. For example, Web 2.0 mashups are used to create virtual exhibitions that present digital objects collected from different sources, as will be further explained in the example implementation below.

4.6.1 RCH Mashups Implementation

An example implementation of Web 2.0 mashups in RCH is represented in the form of a 3D Scrapbook [1]. The RCH's Scrapbook was devised as a mashup that is used to retrieve data from different online sources. These sources include Bing (which is a search engine), Flickr (which is an image sharing website), YouTube (the video sharing website), and a museum's archive (ARCO for example). Data is retrieved through the web services and APIs of the respective data sources being used in the Scrapbook. It is then presented in virtual 3D exhibitions that can be designed by end-users using a set of interactive tools provided within the RCH's web interface.

The provided mashup can be utilized in different ways. For example, a participating museum may create its own virtual exhibition by using the scrapbook's mashup functionality. This can be achieved by grouping related objects in a virtual showroom that has a 3D interface. Furthermore, the RCH mashup functionality comprises a search and retrieval capability that is used to search the data collections present in the online sources being used. For example, the videos that are related to a specific object can be

retrieved by using the integrated YouTube search functionality. Moreover, the search capability is further complemented with a catalogue functionality that is used to aid users to select their desired objects and save them for later presentation purposes.

The Scrapbook interface is illustrated in Figure 4-5 that highlights its functionality. The RCH Scrapbook is based on two main operations: the object retrieval operation and the exhibition (exposition) creation process. The search capability is provided by means of utilizing the RCH mashup search engine to locate the data related to the objects in which the user is interested. The found objects, which may comprise text, images and videos, can then be presented within a 3D virtual exhibition. Users are equipped with a simple drag-and-drop interface that allows them to arrange the gathered object into a virtual 3D museum space, which can be saved, either online or locally, for preservation and presentation purposes.

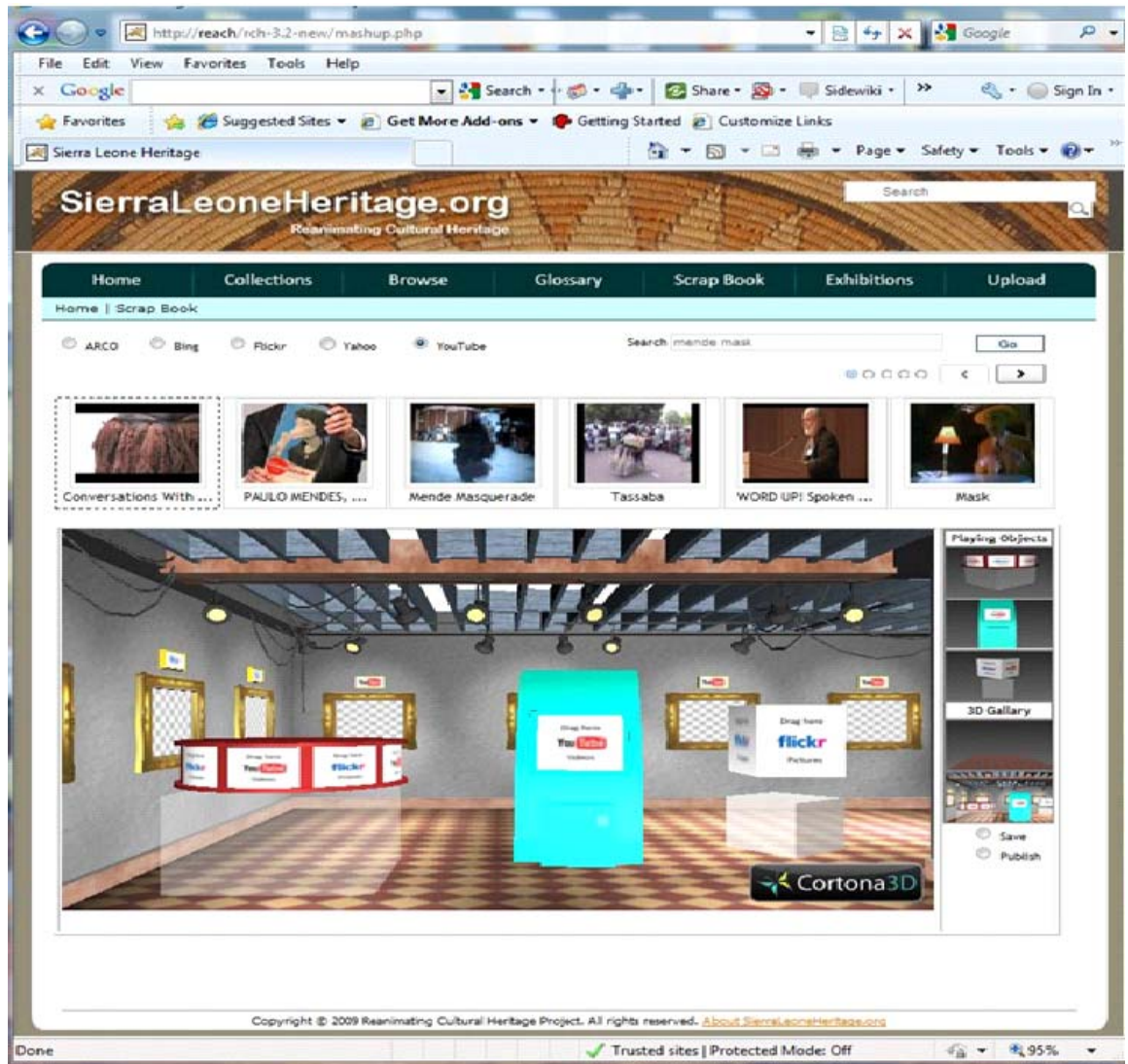


Figure 4-5 RCH's Web 2.0 Mashup Scrapbook

4.7 Social Networking Integration

The concept of ‘reanimating cultural heritage’ cannot entirely rely on curators’ interpretation; source communities and interested users should also have the ability to add their input to the visualized objects. By allowing communities to enhance the cultural and heritage object collections, they will be able to add their own perspective to the managed objects while preserving their curatorial integrity. Therefore, in addition to exploiting Web 2.0 mashups, RCH utilizes social networking technologies to give its contents an extra depth while enhancing user interaction and participation.

Social networking is an emerging computing paradigm that allows web users to interact and socialize online by means of using a number of shared tools and databases [126]. Social networking technologies allowed web users to communicate and collaborate beyond the technological and geographical barriers imposed by any given ICT infrastructure [1]. They also allowed for the creation of rich online communities that generate contents made by their users. Such contents evolve and become enhanced over time as the social network grows in size. RCH utilizes a number of social networking platforms to enhance the contents that it is presenting to its users by adding user-generated contents to them. An example of social networking integration can be seen in Figure 4-6, where users are able to add their input in regard to the displayed digital objects by using an integrated Facebook component (comments textbox).

Social networking integration with RCH is centred on the idea of a ‘Shared Interest Focus’ [1], as a number of users share their interest in a certain topic leading to sharing discussions and objects related to it. This process leads to the creation of user-generated digital contents where shared online repositories are generated based on the users’ focus. By doing this, the user communities are able to communicate and discuss their heritage objects’ data. They are also empowered to submit digital representations of their objects (for example images and videos) to appropriate social networking mediums such as YouTube and Flickr. Within RCH’s context, YouTube and Flickr components are utilized as a part of the RCH mashup services, as was illustrated in Section 4.6, thus they then become part of the virtual museum. Moreover, Facebook was also integrated to gather user-generated contents and feedback as will be highlighted below.

In a typical social networking scenario a user will search for a specific object through the system’s object browser. The RCH search engine will return a list of the objects that

match the entered keywords. The user then clicks on the object that he is interested in to view its full details. Being logged into his/her Facebook account, the user will be able to comment on the object on display and this will be associated with the object and the RCH's pages in Facebook. This scenario is illustrated in Figure 4-6.

The screenshot displays the RCH website interface. At the top is a navigation bar with links: Home, Collections, Browse, Glossary, Scrap Book, Exhibitions, and Upload. Below this is a breadcrumb trail: Home > Browse Objects > Browse Results. The main heading is 'Glasgow Sierra Leone Objects: Images'. On the left is a thumbnail of a 'sowe mask'. To its right is a detailed information table:

Object:	sowe mask
Description:	Sowe mask, used in Sande society, Mende, of blackened wood, 5 fan-ridge hair style with white metal strips nailed along each ridge edge. Carved in a simpler style which Ruth Phillips thought at first was a Gondei style, but the mask has too much tin plate ornament for this. It is made from a heavier wood than the other sowe masks, a hardwood rather than a cotton tree wood.
Materials:	wood, metal
Culture Group:	Mende
Dimensions:	overall: 400 mm x 190 mm x 220 mm 1853 g
Production Date:	No Data
Associated Places:	Africa, West Africa, Sierra Leone, Mende (place associated)
Creator:	No Data
Source:	Julian and Judith Massie-Taylor and Vivien Scarth
Collector:	No Data
Museum:	Glasgow Museum
Accession Number:	GLA:MG-A.1985.13.ac

Below the table is a 'Related Objects' section showing a row of eight thumbnail images of similar masks. Underneath this is a 'Comment:' section with a Facebook social plugin. It includes a 'Like' button, a message 'Be the first of your friends to like this.', a user profile picture, and a text box for 'Add a comment...'. There is a checkbox for 'Post comment to my Facebook Profile' and a 'Post' button. At the bottom left of the comment section is the text 'Facebook social plugin'.

Figure 4-6 Facebook Social Networking Integration with RCH

4.8 Heritage Workflow Management in RCH

In a digital heritage resource such as RCH, data flows can be very convoluted (they can be simple or complex) due to the different scenarios and user cases that are related to the operation of RCH's different components and services [5] [6], thus requiring many complicated and varied workflows. Such services may require sequential or simultaneous operations within the system's workflow hosting environment. These operations are compounded with a number of complex operational scenarios and workflows that necessitate the existence of effective tools to manage, coordinate, track and monitor them. Hence, within the context of RCH, a workflow management system (WfMS) needs to be designed to allow the different museums' digital collections to

integrate with each other while using the shared environment and tools provided by RCH. RCH's workflow management solution is based on the implementation model highlighted in Section 3.5.2, the details of its implementation are shown in Chapter 6.

RCH workflows can be either manual or automatic. An example of a manual workflow is the act of taking a photo of an object. A good example of an automatic workflow is the process of mapping the museums' data from one format to another, which is done through the system's archival services. These two categories encapsulate all the system's workflows (heritage workflows) that operate either sequentially or in parallel according to their underlying scenarios and user actions. It is possible to design many different groups of workflows depending on how the RCH system is implemented. Figure 4-7 shows an abstraction of an RCH (or DISPLAYS) type system, where we can see the five main services and a set of four layers: digital, process, system and user.

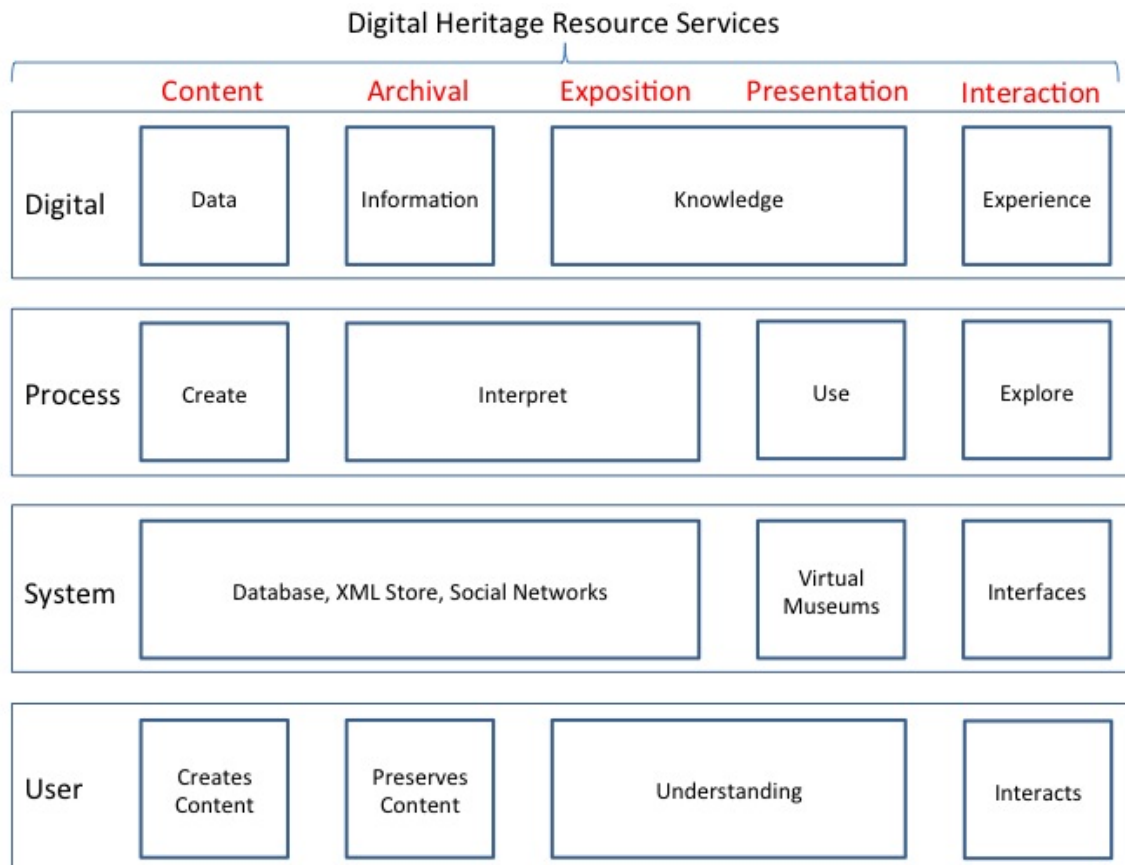


Figure 4-7 Abstraction of a RCH Type System

In Figure 4-7, we can see that we could create 'system' (to manage the overall RCH based virtual museum), 'content', 'archival', 'exposition', 'presentation' and 'interaction' workflow managers that match the service based components. On the other hand, we could build the 'create', 'interpretation', 'use' and 'explore' workflow

managers implied at the ‘process’ operational level. Whichever approach is used, each of these workflow managers is composed of two parts: 1) a host, which is the underlying application environment, and 2) the actual workflow, which is a runtime object that is populated with the service components. For example, archival and exposition services would be runtime objects used during the execution of the ‘interpretation workflow’, which is hosted in an application environment. The full details of RCH’s integrated workflow system implementation are comprehensively covered in Chapter 6 of this thesis.

4.9 Operational Scenario

The set of scenarios below highlight some of the functional areas that are related to the use of RCH by the participating museums.

4.9.1 Content Sharing and Distribution

The Brighton Museum wishes to add its data to the RCH digital resource. It utilizes RCH’s archival services to convert the data of its latest objects to a format displayable in RCH’s interface. The same applies to the two other museums, which constantly enhance the created shared heritage resource by adding more objects from their collections. This is a two-way operation as museums can also import data from RCH to enhance their own collections. Each museum uses the RCH data mapping and conversion tools to import/export the required data.

4.9.2 Object Retrieval

A user is looking for traditional tribal Sierra Leonean masks. He opens the RCH web interface (see Figure 4-4) and uses the keyword-based retrieval services to locate the objects that he is looking for. He views a list of search results and clicks on a mask that he is interested in to obtain its details. The RCH presentation services display the mask’s image and textual details to be viewed by the user. The user gets to see more results via the related objects gallery in the object details page.

4.9.3 Integration of Web 2.0 Mashups

A user browses the RCH frontend and opens the Scrapbook page (see Figure 4-5). He searches for the item “Mask” and chooses to find results in Flickr and YouTube. He then drags and drops some of the results in the 3D virtual museum space to use them to

illustrate some of the concepts that he intends to illustrate to a study group. The user can then save and publish the scrapbook to virtual space, thus adding knowledge to the collections.

4.9.4 Integration of Social Networking Functionality

A researcher is interested in finding African shields: he uses the RCH interface to locate some Sierra Leonean shields. Using his own knowledge, he notices that one of the shields is similar to one found in Ghana. He uses the Facebook commenting box to add that comment, which gets published under the object (see Figure 4-6). Other users start adding comments trying to explain the reasons behind such similarities.

4.10 Summary

The RCH system represents a good implementation or validation model for the DISPLAYS conceptual framework. RCH's utilization by the three museums that use it proves that it can be effectively used to serve the purpose of reanimating Sierra Leone's cultural objects. This reanimation is complemented by allowing RCH users to share and exchange their data. This model led to the creation of a highly available and accessible online digital resource that can be shared and accessed by a number of users in a distributed environment.

RCH is an SO application where modular services provide its overall functionality. In this regard, RCH has four main services, which are the archival, exposition, presentation and interaction services. It also employs some emerging web technologies, such as Web 2.0 mashups and social networking platforms, to enhance the contents displayed in its web presentation layer.

Web 2.0 mashups form an integral part of RCH's web interface where users are allowed to aggregate contents from different online sources such as YouTube and Flickr. Objects are located by means of using a set of advanced search and retrieval tools to locate the data that is related to the cultural objects of interest. The aggregated data can then be further animated by representing it in a virtual museum or a collection of web based exhibitions that are highly customizable and can be preserved for later online and offline use.

Social networking is also integrated within the system for the vital goal of adding the community's perspective into the objects on display. This integration is achieved via the

integration of Facebook, where, for example, users can comment on the displayed objects via a Facebook comment box. Facebook integration led to enabling the RCH users to have more interaction modes with the displayed contents, which consequently led to enriching the data on display with community-oriented contents.

Finally, the complex and intersecting system workflows make it necessary to devise an appropriate workflow management engine that is hosted within the system's enterprise. This workflow system aims to provide the necessary workflow management, tracking, coordination and monitoring capabilities for the RCH services and components. The RCH's data and control workflows are classified into six interrelated categories which are system, content, archival, expositions, presentation and interaction workflows. These workflows are meant to be managed by the devised workflow system. Based on that, workflow management formed an integral part of the RCH system as it managed its different functional operations, such as data mapping and conversion, content creation, object retrieval, etc., as will be detailed in Chapter 6.

CHAPTER V

5 RCH Archival and Presentation

This chapter discusses in more detail two of the RCH system services or components that will be modelled in Chapter 6 as a hosted workflow environment to test the validity of workflows in digital library systems (DLS). The architectural details presented in this chapter are based on the SO implementation model discussed in Chapter 4. This particular prototype implementation of the RCH archival and presentation components is described in terms of the Model-View-Controller (MVC) design pattern. The functionality of the archival create, read, update and delete (CRUD), and presentation (retrieve and display) components are illustrated in terms of creating digital objects for archiving in an XML store using the prototype archival processes. Retrieval of the same objects is achieved using a search and browse functionality to present these objects on the virtual museum interface. Additionally, this chapter also discusses the underlying implementation of the tools that the system provides, while focusing on their management and data flow aspects. These will be discussed in more depth in Chapter 6.

The design of each of the highlighted RCH components is outlined in conjunction with the technical solutions used to build and manage it. The interaction between the illustrated components is also highlighted. This interaction and message passing between the system components is also detailed in Chapters 6 and 7. This chapter also discusses the involved user interaction elements wherever applicable, especially RCH's website frontend.

5.1 Introduction

A system that is as convoluted as RCH has to overcome a number of challenges to be able to meet its goals and objectives. Such challenges range from the varied data formats that need to be handled by the system, to the need to effectively manage its workflows. A typical RCH workflow starts with the process of content creation and ends with content presentation. This process involves a number of intersecting data flow and control workflows that are inherently complex. Furthermore, RCH has to deal with a variety of digitized museum collections that belong to the participating museums. It

also has to deal with other digital museum systems and databases, such as the case with the ARCO system as explained in Section 4.4.1. This challenge is compounded by the fact that different museums use different data formats and collection management systems, which causes a number of compatibility issues [127].

Another important challenge that should be looked at in this context is the object data mapping needs. What is meant by ‘mapping’ is the process of mapping the object-related data from the formats and naming conventions used in the participating museums against the ones used in RCH, and vice versa. This process is an XML-based mapping process and required building custom-made mapping tools to facilitate data transfer and exchange operations, as outlined in Section 5.3.2.

In order to effectively manage the stored Sierra Leone’s collection objects, RCH employs customized tools that support a fully-fledged digital heritage resource workflow. This workflow spans the process of archiving and presenting the stored objects (starting with retrieval using search and browse and then displaying the result). This is achieved through specialized components that interact with each other within the system’s architectural model. Each component does not operate in isolation from the others; rather all components work in coordination with each other via message passing and parameter exchange (either directly or as a web service), as will be further showcased in Section 5.2 and Chapter 6.

The RCH components and their related tools are illustrated in Figure 5-1. In the archival component’s block, it can be observed that the archival related activities revolve around CRUD operations performed on XML files. The presentation component retrieves and displays data through the interaction with the devised search/exposition tools within RCH. These components interact with each other via message passing, as further illustrated in Section 6.6.2.

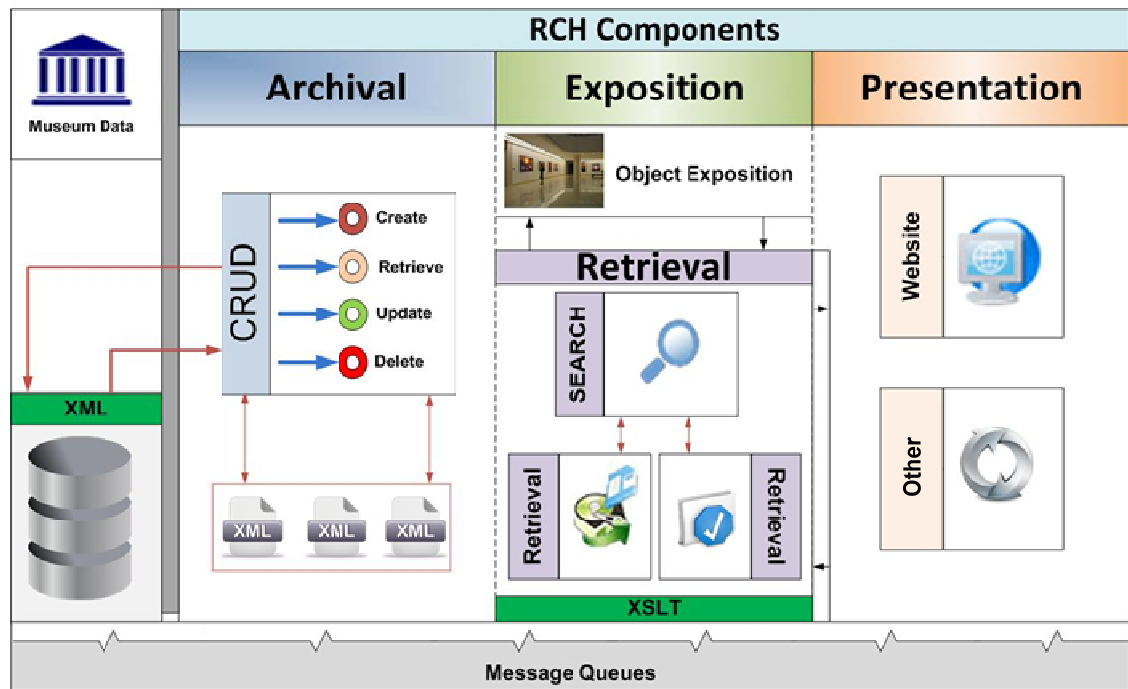


Figure 5-1 RCH Components

5.2 RCH MVC Model

As highlighted in Chapter 4, it was necessary to adopt a loosely coupled SO approach to be able to manage the complex RCH components. Hence, it was decided to adopt the MVC model to implement the RCH digital heritage resource. Using such a model was coupled with a number of operational and workflow management implications as further highlighted in this chapter as well as Chapter 6.

According to Marston [128], the MVC model is a software implementation paradigm in which an application is broken down into three parts, which are the **Model**, the **View** and the **Controller**. Furthermore, the MVC model aims to map the traditional software input, processing and output rules that are usually associated with an application's Graphical User Interface (GUI) [128]. The main concept behind the MVC model is based on the idea of totally isolating an application's business logic layer (the controller) from its data (model) and presentation (view) components. This paradigm aims to pave the way for separate and independent development, testing and maintenance for each of the developed components [129]. Moreover, the MVC model is the recommended architectural design paradigm for interactive web applications [130] as it provides a range of practical benefits. These benefits include centralized application control and flexibility in building multiple UI elements (views).

The MVC model provided RCH with a flexible model where it is possible to easily manage and maintain the different system components. The underlying RCH MVC implementation is highlighted in Figure 5-2 where the relationship between the Model, View and Controller parts of the RCH system is illustrated. Figure 5-2 shows that the RCH MVC implementation has three core components: the archival XML files, which represent the **Model**, the business logic XSLT files (and associated PHP server side scripting), which represent the **Controller**, and a number of dynamically rendered XHTML files that represent the **View** part of the model. The particular RCH functionality discussed in this chapter focuses on three distinct specialized RCH components that are encapsulated within the MVC model, which are the archival, presentation and retrieval components. The details of these components are discussed in Sections 5.3, 5.4 and 5.5 of this chapter.

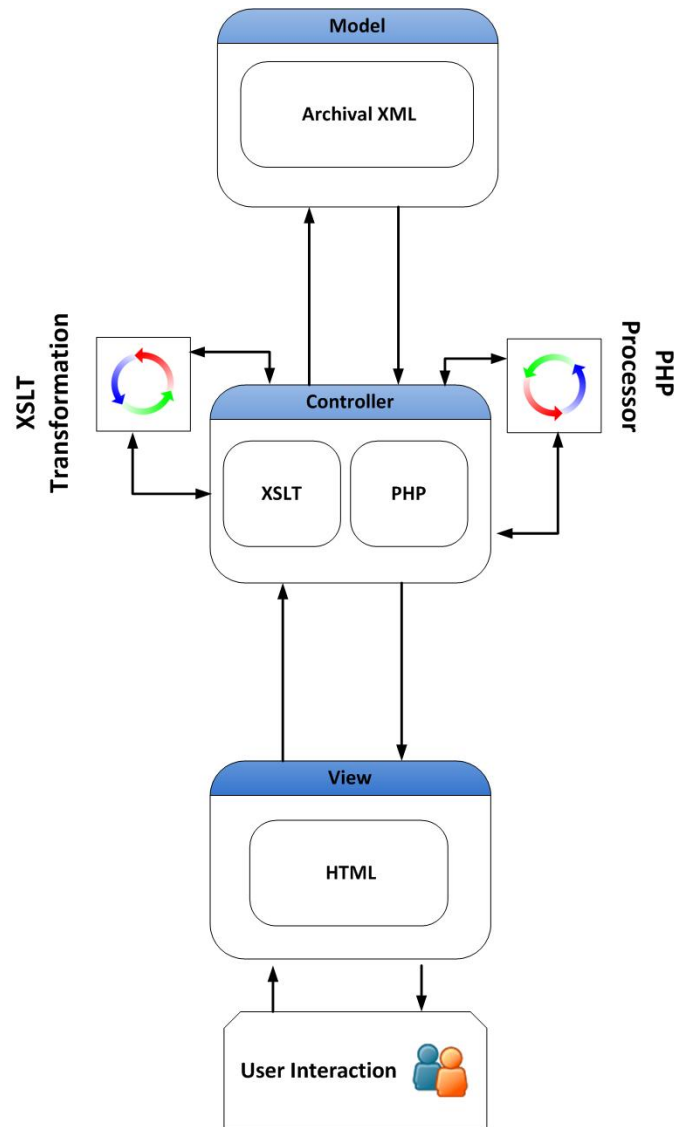


Figure 5-2 RCH MVC Implementation

5.3 RCH Archival Components

As was highlighted in Chapter 4, RCH comprised a number of interrelated components. From now on the term component rather than service is used, because the goal is to implement a prototype system to test the validity of workflows, rather than focus on whether the system uses web services or components. These components work in coordination with each other to achieve the required functionality. The RCH implementation as an MVC model focuses on components that are related to the archival, retrieval and presentation services, which are of significant importance to the RCH users. These components were built from scratch by the author of this thesis to validate the DISPLAYS concept and, more importantly, the use of WfMS within a DLS, as illustrated in Chapter 6.

Digital archiving systems, especially those that are used in cultural institutions, play a pivotal role in building and maintaining rich and accessible online digital resources. Furthermore, Kawano [131] indicates that such systems are usually associated with various types of digital contents such as textual information, images, videos and 3D object models. Moreover, Kawano [131] also argues that innovative technologies contribute to the long-term preservation of such objects; a worthy goal that RCH is pursuing in relation to Sierra Leone's cultural heritage objects in a practical way, as was explained in Chapter 4.

The implementation hurdles that the RCH archival components had to overcome can be compared to the ones faced in a number of similar implementations. For example, the team that developed the project of 'Digitizing the National Commission for the Protection of Human Subjects of Biomedical and Behavioral Research Collection' [132] moved through the whole archiving workflow while encountering a number of challenges. According to Kelley *et al* [132], these challenges included determining the appropriate settings, standards, and efficient workflow steps. The RCH project faced the same set of challenges, especially in relation to the different data formats and conventions used in the participating museums. Such problems were further complicated by the need to map each museum's data to the standard RCH format to be displayed in the designated RCH website frontend. The techniques that RCH adopted in its prototype archival components were aimed at overcoming such challenges, as highlighted in Section 5.3.2.

The goal of allowing the participating museums to share and distribute their data was realized by means of creating a number of specialized distributed archival tools. These tools perform a number of tasks related to the underlying needs of the archiving process. These tools include custom-made data mapping and migration utilities; they were built from scratch to validate the archival workflows as illustrated in Section 5.3.2. Moreover, the provision of the archiving tools was instrumental in the operation of the whole RCH system as it allowed the participating museums to achieve the key goals listed in Table 5-1.

Table 5-1 The RCH Archival Services Goals

Goal	Purpose
Collection Unification	To unify the collections of the ‘communities of practice’ (i.e. the participating museums) into a single shared and highly accessible digital resource.
Data Archiving	To allow the participating museums to archive their own collections within the RCH repository.
Data Mapping	To allow the participating museums to export/import their objects to/from the RCH repository.
Data Exchange	To allow the participating museums to exchange digital object collections among themselves.

5.3.1 The Archival Tools

The current RCH implementation is a ‘lite’ implementation, where the data is stored in XML files rather than using a traditional relational database backend (see Section 4.4.1). Using XML files is justified by the need for a flexible medium to allow for quick and accurate data mapping and exchange operations between the participating museums. The data belonging to each museum is stored in a separate XML file that is

handled by RCH's code modules for the data mapping, retrieval and presentation purposes. Furthermore, the RCH archival component comprises a number of tools that enable the participating museums to submit their collection-related data to be added to the RCH repository; hence, the archival mapping process can be a two-way operation.

The archival tools have two operational modes that are managed by the devised workflow management infrastructure as illustrated in Chapter 6. The first operational mode is an **online** mode where the data mapping and transfer tools are provided as a set of application tools that are published in a shared network. This network can be either the Internet or an LAN.

The other mode of operation is the **offline** mode that serves the needs of the museums that are unable to directly connect to the published archival tools. In this context, a set of independent application tools that can work in isolation from the system's enterprise are provided. These tools have the ability to synchronise their data with the main system when connected to the appropriate network resources. The RCH archival components comprise a number of specialized tools outlined in Section 5.3.2.

5.3.2 Data Mapping Tools

Data transfer from the participating museums to RCH and vice versa is not a straightforward operation. This is because the participating museums use diverse naming conventions, standards and file formats for capturing, managing and archiving the cultural objects in their collections. Additionally, the data migration process is further complicated by the fact that some museums use proprietary non-standard metadata schemas that include complex nested data records that need accurate mapping procedures.

Such a complexity made it necessary for the RCH mapping tools to be flexible enough to handle the different formats and conventions used in the participating museums. The author of this thesis transformed the manual mapping operation into an autonomous one by devising a data mapping tool within RCH's archival component. This tool was then hosted and managed by the RCH WfMS as illustrated in Chapter 6.

The devised RCH data mapping tool allowed the participating museums to effectively export their data to RCH to be displayed in its frontend. A snapshot of the data to be mapped between the three participating museums is illustrated in Figure 5-3. Each museum uses its own metadata naming conventions for its Sierra Leonean cultural

heritage objects. Therefore, to unify the managed collections RCH adopted the ARCO Metadata Element Set (AMS) [133] to enable a more visitor friendly mapping for the objects that it stores and distributes. Hence, transferring data from a museum to the RCH repository involves mapping the migrated data to the RCH format.

The mapping tool itself is flexible enough to be updated with new schema mapping information when the need arises. Such an update is needed when a museum changes the naming conventions in use, or adds new data fields to be mapped to the corresponding RCH fields for instance.

The data mapping process is a two-way operation as it is possible to map the museums' data to RCH format and vice versa. The need to map from RCH's format to the museums' format arises when museums need to import data from RCH to enhance and complement their own collections. The different mapping scenarios are detailed in Section 5.3.4.

AMS Elements	Object Category	Name	Description	Material	Culture Group	Dimension
RCH Elements	Object Category	Object	Description	Materials	Culture Group	Dimensions
Glasgow Elements	Classifications	Object Name	Description	Materials	Culture/School	Measurements
Brighton Museum Elements	Sub Collection	Object Name	Description	Materials	Culture Group	Measurements
British Museum Elements	Object Category	Object Name	Description	Material	Ethnic Name	Dimensions
	musical instruments	gunibri; lute	Gunubri or lute, lute, rectangular body, with wooden sides, decorated with metal studs and shells, with leather stretched over, and short cylindrical wooden handle	wood, leather, metal, shell, sinew	West African	overall: 840 mm x 180 mm x 125 mm 1873.0 g
	magic and religion	figure	Black wooden figure of Mende girl, prior to initiation of cicatryation on arms, face and stomach. From Sierra Leone.	wood	West African	overall: 463 mm x 115 mm x 116 mm 702 g
	magic and religion	figure	Black wooden figure of Ghinzi woman. Initiation of cicatryation on face, arms and back. From Sierra Leone.	wood	West African	overall: 465 mm x 135 mm x 110 mm 646 g
		iron money	Example of Ghizi money, consisting of twisted iron rod with t-shaped and spade shaped ends. From Sierra Leone.	iron		l: 16 1/2 in
	currency	iron money	Example of Ghinzi money, consisting of twisted iron rod with t-shaped and spade shaped ends. From Sierra Leone.	iron	West African	overall: 425 mm x 55 mm x 2 mm 23 g
	currency	iron money	Example of Ghizi money, consisting of twisted iron rod with t-shaped and	iron	West African	overall: 376 mm x 60 mm x 3 mm 20.5 g

Figure 5-3 A Snapshot of the Managed Digital Heritage Object Data

5.3.3 Design and Analysis of the Archival Mapping Tool

An online mapping tool was created to fulfil the required data management and distribution producers between RCH and the participating museums. The term ‘online’ is used here as the tool was developed as a web application that interfaced with RCH and the participating museum repositories. The devised mapping tool had a number of key requirements as follows:

- the mapping tool needed to be flexible enough to be able to accommodate new schema mapping rules, or modifications to the current defined rules and conventions.
- it needed to be effective enough to handle complex mapping operations within reasonable conversion time and resource consumption rates.
- it should be able to handle any exceptions such as missing tags, blank records, runtime errors, etc. without crashing (effective error handling).
- it must have a user-friendly interface for fast and straightforward mapping operations.
- it should have a mechanism whereby new schemas can be introduced and saved easily.
- It should be able to effectively handle a variety of file types (XML, Excel, PDF, TXT, etc) for the data import and export operations.

The functionality of the mapping tool revolves around manipulating the ‘node’ attribute within the handled XML files, which are used to hold the data associated with the managed cultural heritage objects. The node names represent the names of the cultural object attributes stored in the participating museums’ data repositories. So, at a higher level, the mapping process involves mapping the node names of the museum from which the mapping process is initiated to the names being used in the target museum’s data schema.

It can be seen in the data sample in Figure 5-3 that ‘<ObjectCategory>Musical Instruments</ObjectCategory>’ represents a typical node that gets mapped when conducting the mapping process. So, when involved in the mapping process, the attribute <ObjectCategory> (cultural heritage object property name) will be mapped to the corresponding name in the target museum’s XML schema. The actual object data is not altered in any way in the mapping process as the entire process aims only at

mapping the node name (object attributes). The goal of this process is to facilitate the cultural heritage object data import and export operations between RCH and the local museums repositories as further highlighted in Section 5.3.4.

One of the key functional aspects of the created mapping tool is the data import and export mechanism. As the devised mapping tool is data-intensive in nature, it needed to have powerful data import and export capabilities to enable the participating museums (or communities of practice) to perform the needed data mapping procedures. The mapped data is then used in the data import/export activities between RCH and the museum repositories. Hence, it was taken into consideration to firstly allow the mapping tool to accept a variety of file types such as the standard XML files as well as other formats that hold structured data sets such as Excel Sheets and delimited text files. Furthermore, for data output, XML and other output file formats are also supported according to the user preferences.

The mapping tool was created as a web application as the aim was to make it available online to the participating museums while overcoming any hardware or software limitations. This implementation also suited the distributed nature of DHRs and provided the infrastructure for potentially publishing the same application as a set of public web services. Such web services have the flexibility to be utilized in an online environment in many ways according to the specific needs of the DHR using them.

The UI elements of the created mapping tool comprised a number of basic controls used for the file manipulation operations (import/export) and the actual mapping process. Basic indicators are also being displayed such as the total number of converted records (see Figure 5-12).

The main approach in designing the mapping tool was an Object Oriented one. The mapping tool was built from a number of specialized classes and components. These components performed the required mapping and data import/export functionality. The functionality of the created classes ranged from as basic tasks as a file upload to complicated tasks such as schema mapping and runtime exception handling.

For the sake of flexibility, the mapping tool's users were provided with the ability of saving their preferred schemas. This facility meant that mapping rules can be defined and stored permanently within the mapping tool itself. Moreover, appropriate error and

exception handling is applied. Runtime errors are handled via special exception handlers so that the tool does not crash in the middle of the mapping process.

5.3.3.1 Mapping Tool Components

From a structural point of view, the mapping tool comprises three main components, which are the **Mapping Classes**, **Schema Manager**, and **User Interface (UI)** as shown in Figure 5-4. The Mapping Classes are responsible for carrying out all the mapping-related functionality and handling any errors and exceptions during the mapping process. The Schema Manager is the component responsible for storing the schemas used in the mapping process and their associated mapping rules. The above two components interact with system users through its online UI that can be viewed in standard internet browsers. The whole mapping tool primarily interacts with two types of cultural heritage object data repositories. The first type is the RCH repository itself that acts as the prime storage medium within the mapping tool as it holds the managed cultural heritage objects. The second type of repositories is represented in the actual museum repositories. The mapping tool interacts indirectly with these repositories via their users (data administrators) as will be further highlighted in Section 5.3.3.4.

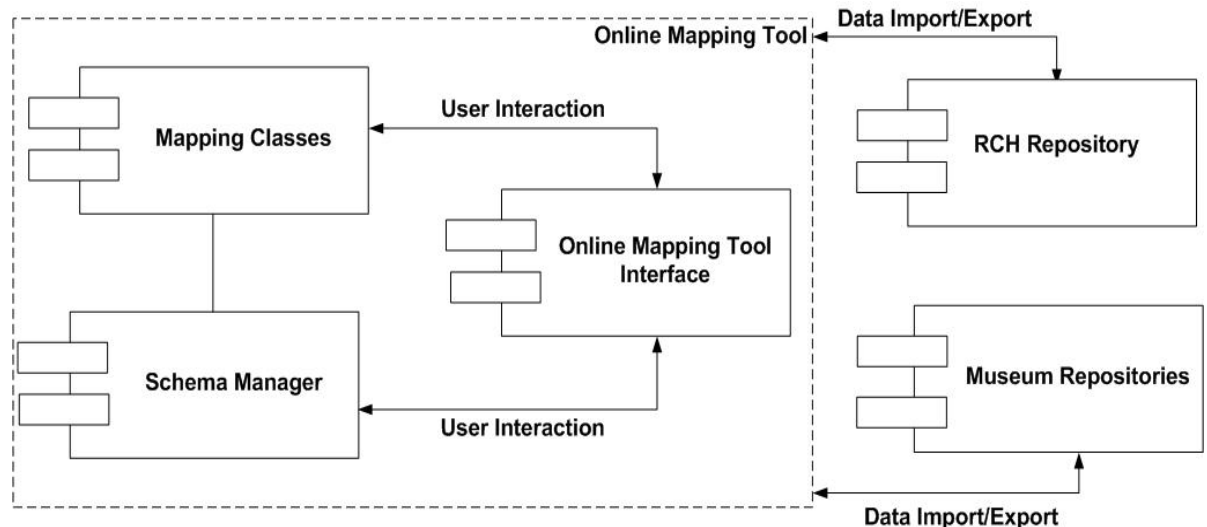


Figure 5-4 Mapping Tool Components

5.3.3.2 Mapping Tool Use-Cases

Figure 5-5 below represents a high level view at the mapping tool's use-cases from the perspective of the museum data administrators. The data administrators represent the main system users who use the mapping tool to perform data import and export activities as well as schema management tasks. It can be seen from the use-case diagram

that there are 4 main system operations. These operations include adding a new schema, managing the existing schemas, exporting museum data and importing museum data (we mean by museum data the cultural heritage objects' data).

The process of adding a new XML schema to the mapping tool is one of the key use-cases involved within the operation of the mapping tool. This process has the goal of defining a new set of schema mapping rules to be used when performing the cultural heritage object data mapping process. It can be seen in the use-case diagram below that the whole workflow of the schema introduction process starts with adding the attributes of the schema file such as its version, source, etc. This procedure is followed by the process of defining the mapping rules for the concerned schema i.e identifying the corresponding nodes in the RCH schema. Then, the new schema gets stored within the mapping tool for later data import and export activities. The stored schemas can also be managed by the mapping tool's users via the provided management interface. Typical schema management operations involve editing an existing schema, deleting a schema, replacing an existing schema and changing an existing schema's mapping rules.

Another important use-case is the one involved with the process of exporting data to the RCH repository via the mapping tool. This process starts with uploading the actual file that contains the data whether it is an XML file, an Excel Sheet or any other supported file types. The mapping tool then converts the XML nodes of the uploaded file into the format of RCH based on the stored schema mapping rules. The mapping (conversion) process starts with firstly identifying the source and target formats and then by performing the actual mapping process. This process ends with storing the converted data objects into the RCH repository resulting in expanding it overtime.

The process of importing data from the RCH repository is exactly the opposite of the cultural heritage data import process. In a typical data import use-case, a museum's data administrator may want to add more cultural heritage objects' data to the database (repository) the museum he is managing. In this case, the whole process starts with specifying the data records that are required to be imported, converting them to the target museum's XML format and finally outputting the resulted file in the preferred format that the user has chosen.

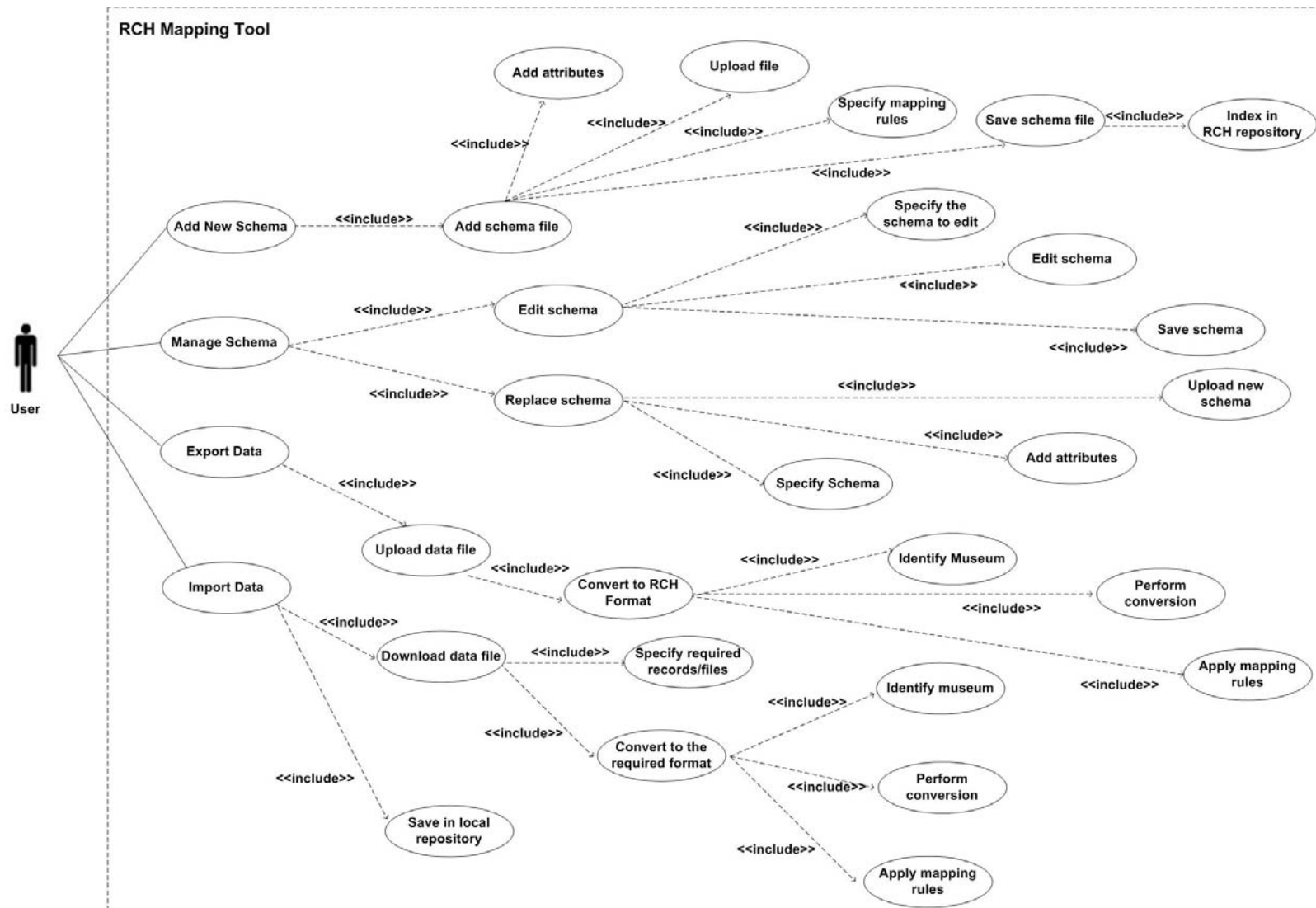


Figure 5-5 RCH Mapping Tool Use-Case Diagram

5.3.3.3 *Mapping Tool Classes*

The RCH mapping tool consists of a number of classes that cover the functionality it needed to perform. What follows is a brief listing and description of the main classes and objects used within the mapping tool, which are illustrated in the class diagram in figure 5-6. These classes collectively represent the overall functionality of the mapping tool.

- **The File Class**

The File class represents the file objects that are being handled through the mapping tool. For example, schemas are being exchanged in standard XML schema files, museum data is being transferred in the form of XML or Excel files and so on. Therefore, the File class has the necessary attributes and methods (functions) to handle the file-related functionality within the mapping tool. Such functionality includes create, open, save, delete and move file. Basic attributes include fileName, Type, Size, Path, etc. The File class is the parent class of the two subclasses which are the inputDataFile and the outputDataFile classes which are used in the data input and out operations respectively. These classes inherit the attributes and methods of the base File class as explained below.

- **The inputDataFile Class**

The inputDataFile class is a class that inherits the characteristics of the File class that is a generalization of this class. This specialised class is used in the data input operations that are based on file exchange. The file exchange operations are usually conducted between the RCH repository and the local repositories of the participating museums. Within the context of the RCH mapping tool, instances of the inputDataFile class are created and used when performing either the data import or the schema addition operations.

- **The outputDataFile Class**

Similar to the inputDataFile class, this class is a specialization of the generic File class. It is mainly used in the data output operations involved with the mapping process. For example, after concluding any of the performed mapping processes within the data export operations, the result is represented in a file that holds the mapped data. This file is passed to the user who initiated the export process to be added to the target museum's repository.

- **The Schema Class**

As illustrated in the use-case diagram in Figure 5-6, one of the main use-cases within the mapping tool is the one of adding and managing the cultural object data schemas. This class performs the actual schema management tasks within the mapping tool. The Schema class is the one responsible for storing and managing the data schemas being used during the mapping process. The defined methods include Add, Delete and Edit schema. The used attributes include Name, ID, Version and Source of the schema.

- **The mappingRules Class**

The mappingRules class is concerned with storing and managing the mapping rules within the mapping tool. These rules are used when performing the actual cultural heritage data import and export activities. As can be seen from figure 5-6, the mappingRule class has a number of key attributes including Source (the museum from which the schema came), Destination (which can be RCH's or any of the participating museums' repositories), nodeArray (the XML node names within the schema) and Version (used to manage and track down the different versions of the same schema mapping rules). There is an association link between the mappingRules and the Schema classes as mapping rules are based on the schemas supplied by the system users. This class has 4 main operations that are: Define, Store, Update and Delete schema mapping rules.

- **The Mapper Class**

This class is responsible for performing the actual data mapping operations according to the rules defined within the mappingRules class. This is indicated in the class association between the mappingRules and the Mapper Classes. This class bases the mapping process on the predefined mapping rules and the preferred user criteria. Such criteria include the source and destination museum formats and the location where the user prefers the final XML file to be stored in. This class comprises a number of key mapping functions including: loadSchemaRules(to load the schema rules needed in the mapping operation), returnNodeAlerts(to alert the user about any exceptions that he needs to handle), returnErrorMessages(to return any runtime error messages), returnIndicators(to return the final results of the mapping process) and returnMappedFiles(to return the mapped files in the user's desired format).

- **The Repository Class**

A repository can be either the RCH repository that stores the cultural heritage objects' data managed through RCH, or the local museum repositories that vary in their technologies and structure. As indicated in the class diagram in Figure 5-6, an inheritance relationship exists between the Repository class and two sub-classes, which are the relationalDatabase and the fileDatabase classes. These classes represent the actual data storage mediums that might be used in the participating museums and inherit the actual repository management functionality that exists in the Repository class. Such functionality constitutes the following core methods: checkStatus, Connect, terminateConnection and flushData.

- **The relationalDatabase Class**

While inheriting the basic repository management functions from the Repository class, the relationalDatabase class has its own attributes and functionality that conforms to its nature and structure. For example, it has attributes such as connectionString, databaseType and functions such as queryDatabase.

- **The fileDatabase Class**

Similar to the relationalDatabase class, the fileDatabase class has its own attributes and methods. The unique attributes of this class include: Path, fileType, fileName, dateCreated, dateLastModified and methods such as searchFile.

- **The Museum Class**

This class represents the attributes of actual museums that deal with the RCH repository. Each museum has one or more data administrators who directly interact with the interface of the mapping tool and perform tasks such as schema definition, data import and export, etc. (see Section 5.3.3.2 for more details).

- **The museumAdminsitartor Class**

The museum administrators are managed through the **museumAdminsitartor** class that stores the details of the concerned administrators. Each museum may have one or more data administrators who represent the main users of the mapping tool.

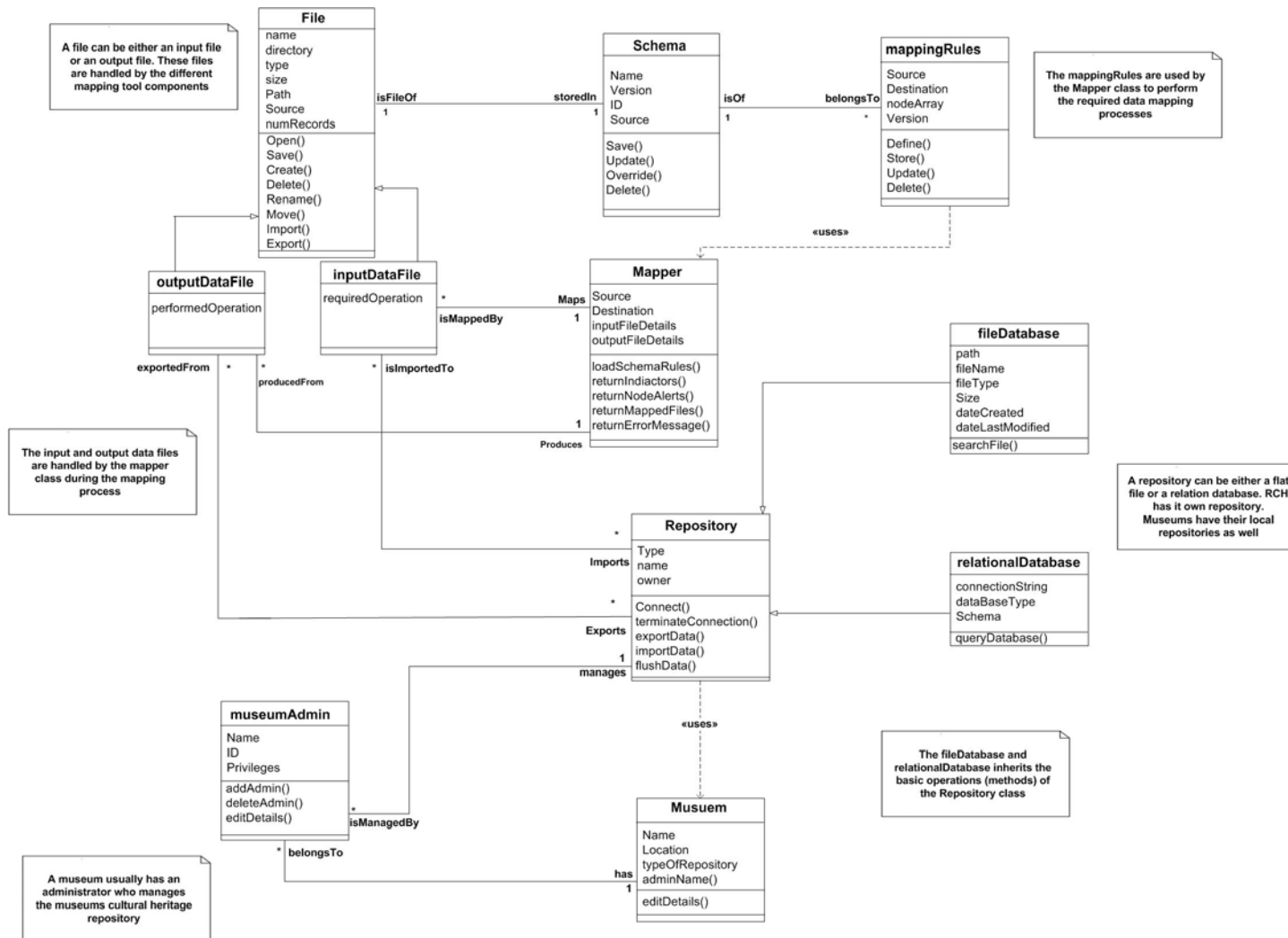


Figure 5-6 RCH Mapping Tool Class Diagram

5.3.3.4 *Sequence of Events*

The operation of the mapping tool is governed by a certain sequence of events depending on the user actions as highlighted in the following set of sequence diagrams.

- **Export Data**

The process of data export has the purpose of adding new cultural heritage object data to the RCH repository. The goal of this operation is to expand the RCH repository and to share the added data among the participating communities of practice. The sequence of events during the data export operation is highlighted in the sequence diagram in Figure 5-7. The sequence of events in a typical data export scenario starts with the user opening the mapping tool in his preferred online browser. The interface of the mapping tool gets loaded into the browser and the user chooses the desired operation, in this case, the operation is the action of exporting new cultural heritage objects' data to the RCH repository. Then, the user starts adding the attributes associated with this operation, such as adding a description and remarks to be added to the data if required.

The actual export process starts with passing the data from the museum's repository to the RCH interface. In this case, the user either uploads a data file (XML or Excel for example) or connects the mapping tool to the database of his museums through the provided connection interface. The mapping tool then checks for any errors or data corruption in the entered parameters and passes them to the specialised mapping classes. These classes start mapping the exported data into the RCH format. The mapping classes start off by identifying the source and destination formats to determine the required mapping rules to be loaded from the Schema Manager, which represents the code infrastructure used to manage the mapping rules as per the defined museum schemas. Then, the data gets mapped into the RCH format based on the loaded predefined mapping rules. The cultural heritage object data export process is concluded by passing the mapped data to the RCH Repository component. This component represents the physical storage medium within RCH. The new data gets stored within the repository and the confirmation of the successful export process gets passed to the user through the mapping tool's interface.

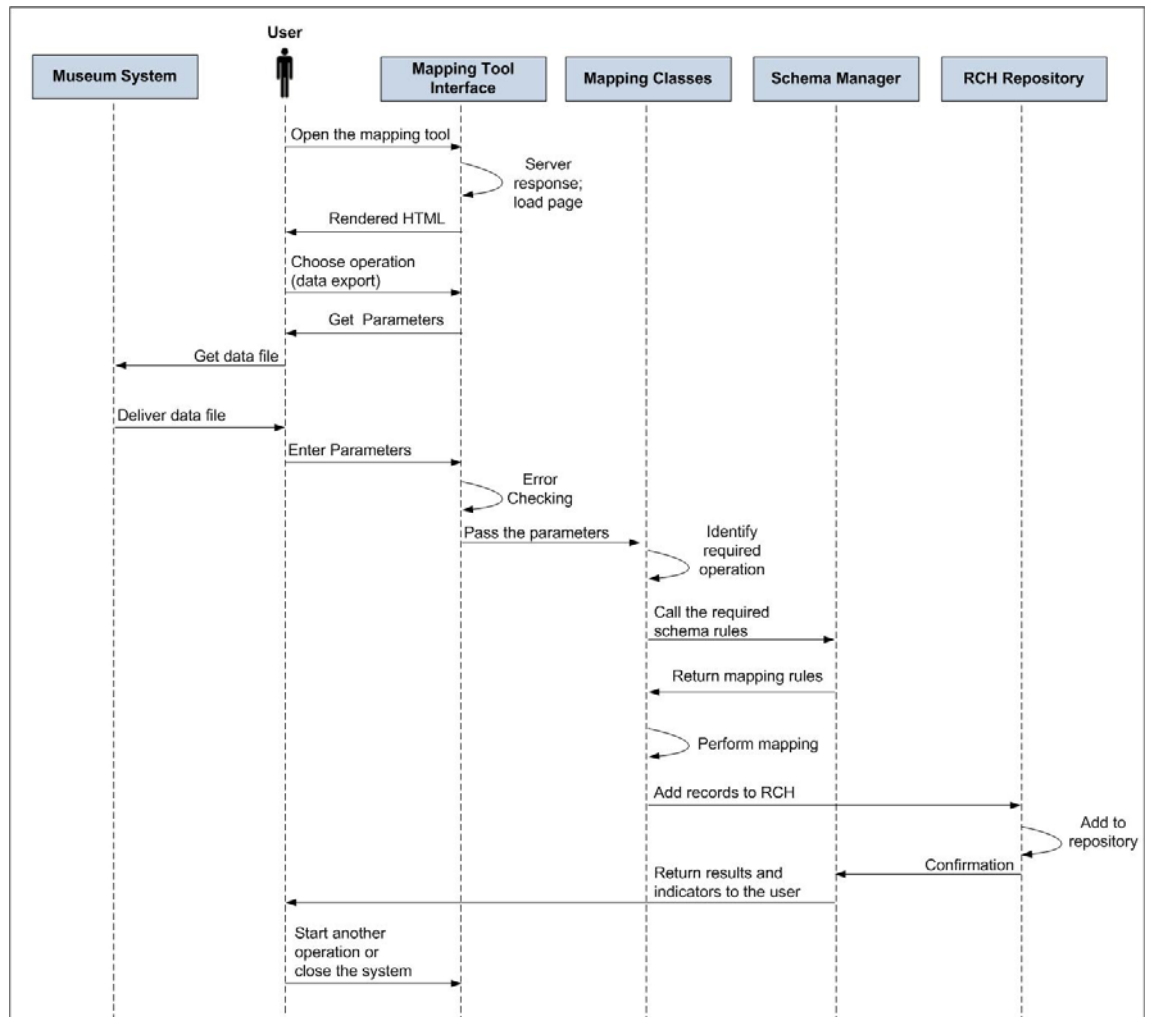


Figure 5-7 Data Export Sequence Diagram

▪ Import Data

Another frequent operational scenario within the mapping tool is the process of importing cultural heritage objects' data from RCH to the local repositories of the participating museums. This process is exactly the opposite of the data export sequence and starts off by the launching the mapping tool's interface through the user's internet browser, as shown in Figure 5-8. Then, the required parameters are entered by the user. These parameters include the range of the data objects to be imported to the target museum's repository. After error checking, these parameters are passed to the RCH repository itself to extract the required data items. The extracted data undergoes the intermediate process of mapping before being passed to the user. The mapping process is a necessary step in this instance; this is because the data needs to be converted to the format being used in the target museum's repository. After performing the mapping

process the resulted data (either in the form of XML files, Excel files, database records, etc.) gets passed to target repository to be added to its existing records.

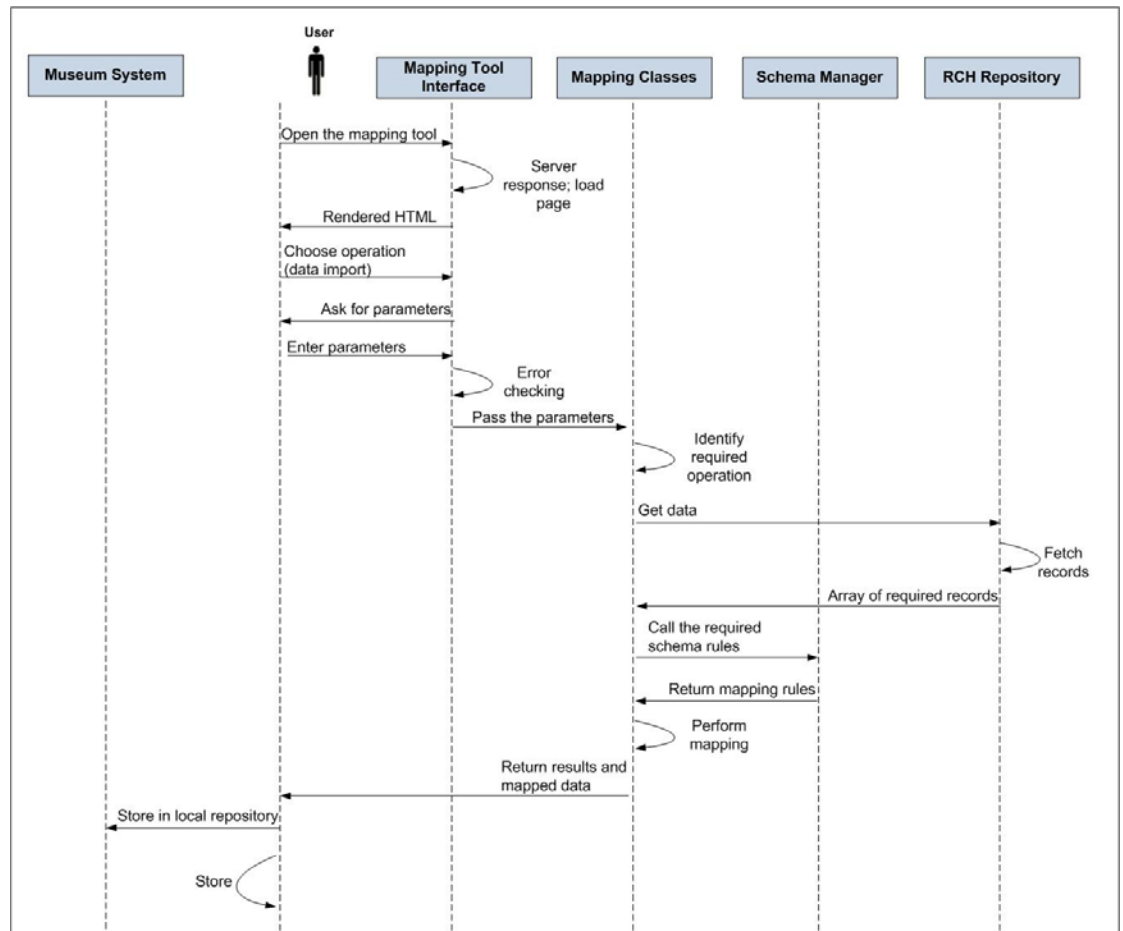


Figure 5-8 Data Import Sequence Diagram

- **Schema Definition**

One of the main use-cases in the mapping tool is involved with the process of adding new mapping schemas as highlighted in Section 5.3.3.2. This is an important operation as the outcome of the mapping process depends on the stored schemas and their associated mapping rules (a practical implementation of the mapping process is illustrated in Section 6.6.1).

The process of adding a new schema starts by opening the interface of the mapping tool as shown in sequence diagram in Figure 5-9. The mapping tool then confirms the mapping rules by prompting the user to review the mapped nodes and make any corrections and modifications if required. These mapping rules are then stored permanently within the RCH mapping tool for later use. The specialised mapping classes use these rules to perform the data export and import operations as highlighted above.

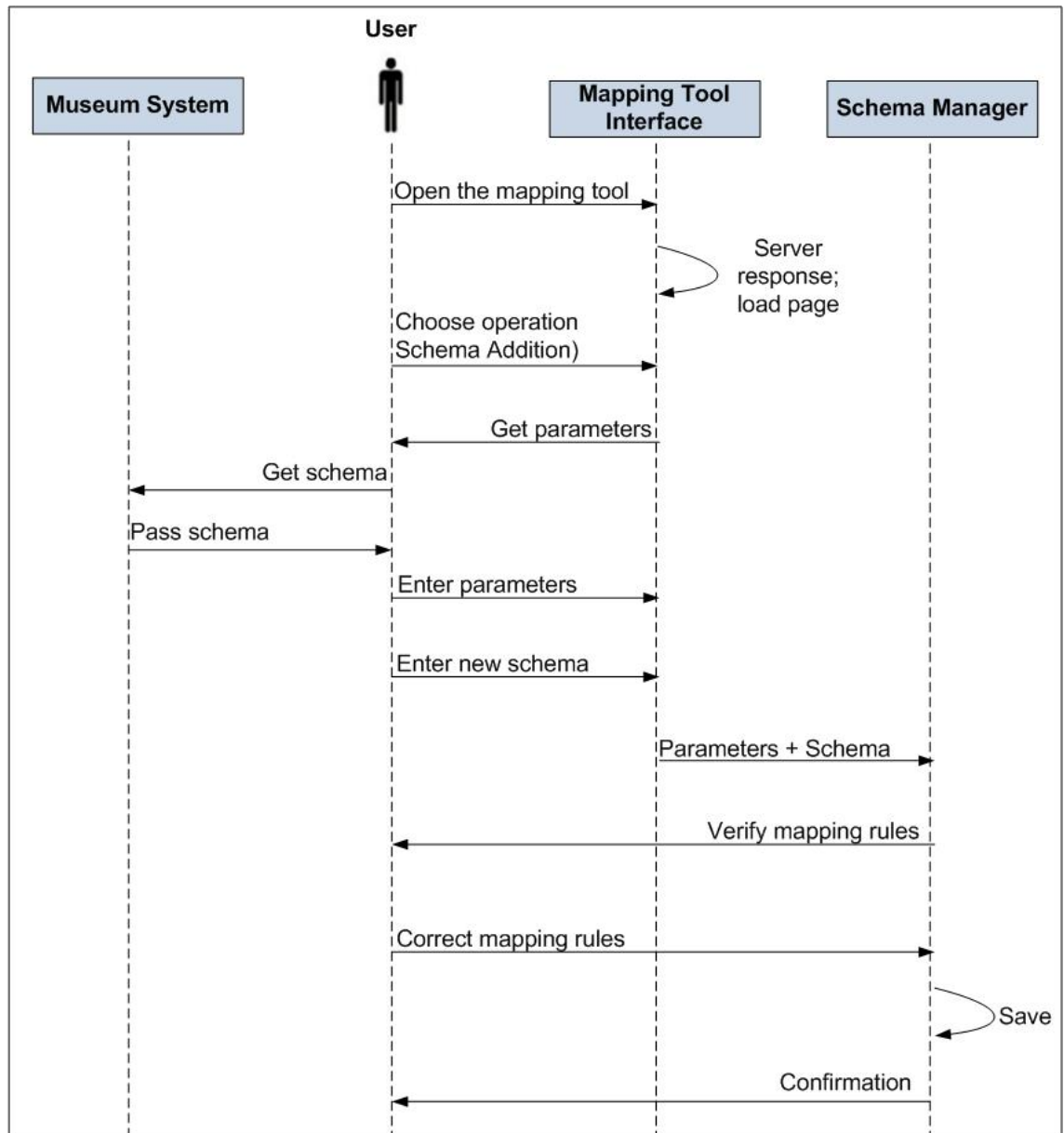


Figure 5-9 Adding a New Schema Sequence Diagram

5.3.4 The Data Mapping Scenario

To meet the requirements of the participating museums, the mapping tool is capable of handling a variety of file formats including XML files. Each file contains the objects' data and their associated attributes and metadata. Each of the participating museums (the British Museum, Brighton Museum and Art Gallery, and the Glasgow Museum) has its own format (naming conventions). During the mapping process, all these formats are transformed to the standard RCH format so that the data can be unified and displayed in a single unified interface (RCH website or virtual museum) while adhering to a single unified format as mentioned above. Once the data is fully mapped by using the provided tools it is then listed in the RCH's frontend in the designated data sections.

The end result of the mapping process is a single valid XML file for each museum that is manageable by the RCH web application, allowing for control of data presentation and retrieval operations.

The mapping tool handles the mapping procedure through a set of predefined mapping conventions. For example, the Glasgow Museum uses the term ‘Classifications’ for describing the categories of its objects, whereas RCH uses the term ‘Object Category’ for the same purpose, as shown in Figure 5-3. Therefore, when transferring object data from the Glasgow Museum to RCH, the Glasgow schema is mapped into RCH’s to enable such a migration.

Another important role that the mapping process plays is enabling the participating museums to expand their own collections data by importing the data used in other museums. Therefore, a museum can import the details of the objects that do not exist in its collections directly from the RCH repository. The steps of the mapping process are illustrated in Figure 5-10.

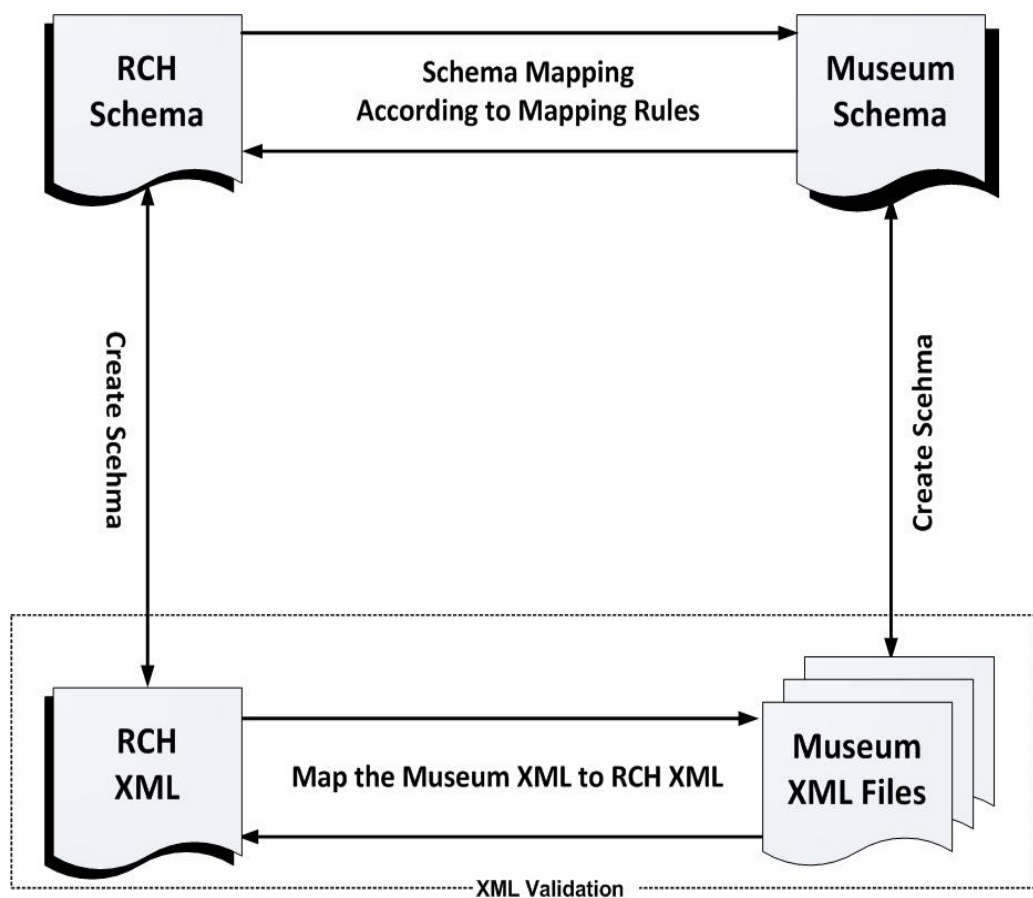


Figure 5-10 The Mapping Process

According to Figure 5-10, the steps involved in the mapping process are as follows:

- **Define schema:** the RCH data schema is defined within the mapping tool.
- **Initialize the mapping tool:** launch either the online or offline mapping tool.
- **Upload the data:** upload the data to be mapped. Data can be in the form of an XML file.
- **Generate Schema:** generate a schema that depicts the format of the entered data.
- **Map the data:** map the schema of the entered data to the RCH schema according to the predefined mapping rules.
- **Validate XML:** validate the resulting XML by checking that the resulting files contain valid XML tags. This process is conducted to make sure that the imported objects' data can be displayed in the RCH website frontend correctly.

5.3.5 The Mapping Workflow

RCH's mapping logic comprises a number of modular code units that facilitate the mapping process. The system also has the ability to deal with any change in the structure of the entered data. This is achieved through the system's administrative end that enables its users to dynamically redefine the mapping rules whenever necessary. Furthermore, the mapping tool can also handle any level of complexity in regard to the entered XML files, including the successful mapping of nested XML nodes. The workflow of the whole mapping process is further highlighted in Figure 5-11.

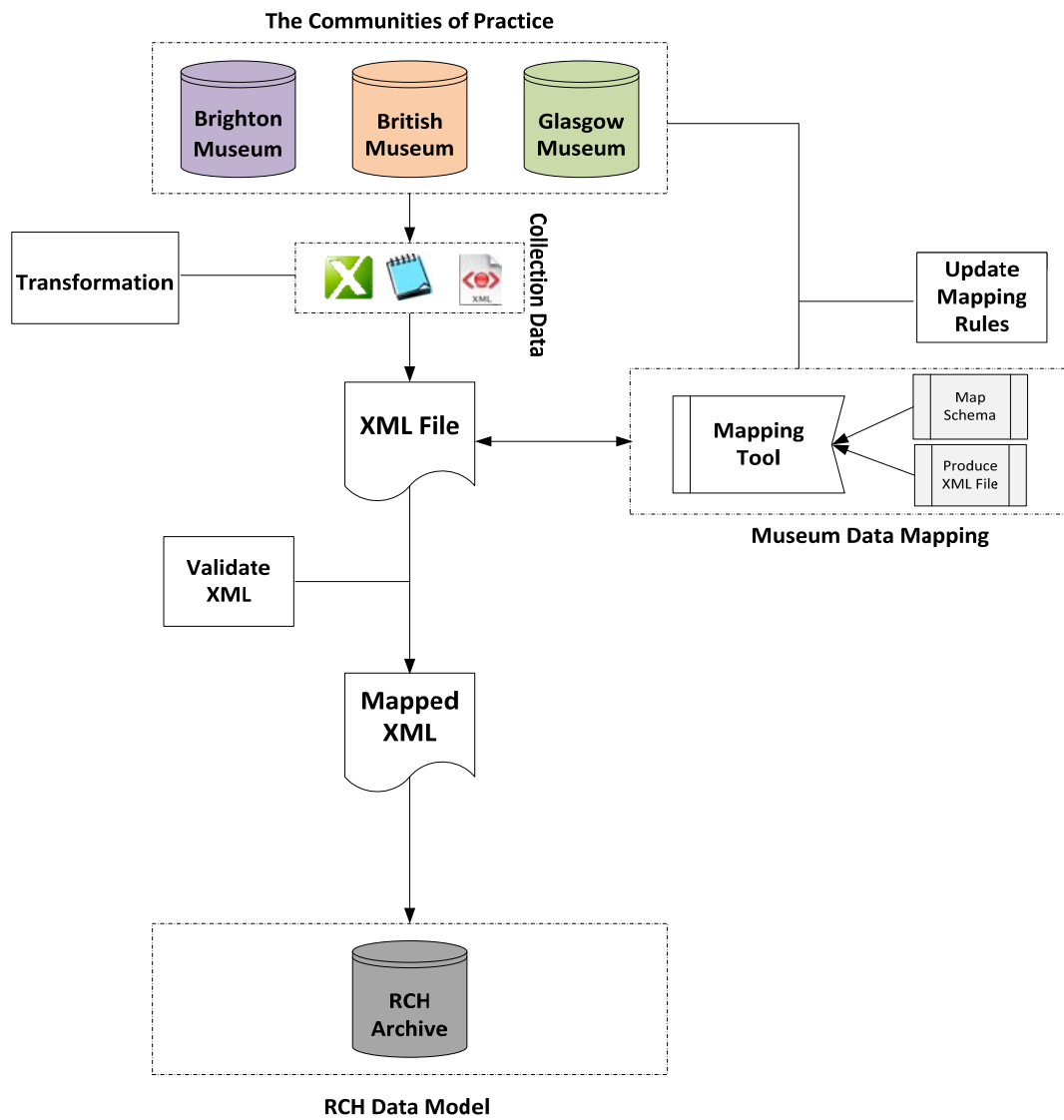


Figure 5-11 The Mapping Workflow

The interface of the RCH online data mapping tool is highlighted in Figure 5-12. The illustrated interface provides a number of controls as a user friendly medium to perform the mapping operations.

XML Convertor	
From Excel	From HTML
<p>Mapping Type <input type="text" value="MIMSY"/></p> <p>Browse for Excel File <input type="text"/> <input type="button" value="Browse..."/> <input type="button" value="Upload"/></p> <p>365 records inserted.</p> <p>XML</p> <p>A new XML file with the name xml28975.xml has been created Open the file!</p>	<p>Browse for HTML File <input type="text"/> <input type="button" value="Browse..."/> <input type="button" value="Convert"/></p>

Figure 5-12 The Online Mapping Tool

As mentioned above, the mapping process transforms the metadata attributes of one museum to the ones used by the other participating museums or to the RCH format. The XML code snippet below shows an excerpt from an XML file that holds the metadata of one of the Glasgow Museum's cultural objects.

```

<CulturalObject>
  <ObjectName>dagger and sheath</ObjectName>
  <Description>Dagger, back of blade inlaid with brass, in
  carved wooden sheath. Used by followers of Mahomet at
  Sierra Leone. From collection of African ethnological
  specimens.</Description>
  <Materials>metal, brass, wood</Materials>
  <Culture/School>No Data</Culture/School>
  <Measurements>overall: 437 mm x 32 mm x 15 mm 263.5
  g</Measurements>
  <PlaceMade>Africa, Equatorial Africa (place of
  manufacture)</PlaceMade>
  <Source>Neil, Thomas and John</Source>
  <Museum>Glasgow Museum</Museum>
  <IDNumber>GLAMG:1877.18.x</IDNumber>
</CulturalObject>

```

For illustration purposes, the devised mapping tool was used to map the Glasgow object's data shown above to the formats of RCH and the British Museum. The XML code snippet below shows the results of converting to the British Museum's format.

```
<CulturalObject>
<ObjectCategory>No Data</ObjectCategory>
<ObjectName>dagger and sheath</ObjectName>
<Description>Dagger, back of blade inlaid with brass, in
carved wooden sheath. Used by followers of Mahomet at
Sierra Leone. From collection of African ethnological
specimens.</Description>
<Material>metal, brass, wood</Material>
<Dimensions>overall: 437 mm x 32 mm x 15 mm 263.5
g</Dimensions>
<ProductionPlace>Africa, Equatorial Africa (place of
manufacture)</ProductionPlace>
<AcquisitionDetails>Neil, Thomas and
John</AcquisitionDetails>
<Museum>Glasgow Museum</Museum>
<RegistrationNumber>GLAMG:1877.18.x</RegistrationNumber>
</CulturalObject>
```

The XML code snippet below shows the results of converting the Glasgow object data to the RCH format.

```
<CulturalObject>
<Name>dagger and sheath</Name>
<Description>Dagger, back of blade inlaid with brass, in
carved wooden sheath. Used by followers of Mahomet at
Sierra Leone. From collection of African ethnological
specimens.</Description>
<Material>metal, brass, wood</Material>
<Dimension>overall: 437 mm x 32 mm x 15 mm 263.5
g</Dimension>
<ObjectProductionPlace>Africa, Equatorial Africa (place
of manufacture)</ObjectProductionPlace>
<AcquisitionSource>Neil, Thomas and
John</AcquisitionSource>
<CurrentLocation>Glasgow Museum</CurrentLocation>
<Source>GLAMG:1877.18.x</Source>
</CulturalObject>
```

5.4 RCH Object Presentation

The object presentation components served the purpose of visualizing the RCH objects and collections with their various data formats. These formats include 3D models and complex multimedia formats. The main medium for object presentation is a website frontend that displays a number of HTML pages to showcase the cultural objects stored within the RCH repository. It is also used in conjunction with the retrieval operations to display the search results.

The actual implementation of the object presentation components involved the utilization of a number of web and data sharing and distribution technologies, including Hypertext Preprocessor (PHP), XML and XSLT. The presentation of the stored objects is achieved through the utilization of XSLT files, which are designed to transform the XML files of each museum into well-formatted valid XHTML that is further styled through Cascading Style Sheets (CSS). This XHTML code is in turn displayed in the RCH website frontend.

The utilization of XSLT files meant that any changes in managed objects are immediately reflected in RCH's frontend: this approach suits the needs of dynamic object data display. This dynamicity in data display is achieved through dynamic XSLT transformation that maintains a constant link with the backend XML files. From a lightweight architectural approach, the key thing to note here is that RCH is effectively using an XML software stack.

An example of the dynamic RCH pages is shown in Figure 5-13 that illustrates the object browsing page. This page displays the latest list of the Sierra Leone's cultural heritage objects augmented from the collections of the participating museums. Each museum's collection is presented in a separate content slider that represents thumbnails of the objects on display. The system users are provided with a number of options in relation to the general object listing, including the use of a number of filters for viewing a subset of the displayed objects. Object views can be customized further by making appropriate changes in the corresponding XSLT files. Therefore, the dynamically produced XHTML code can meet the evolving needs of the RCH users.

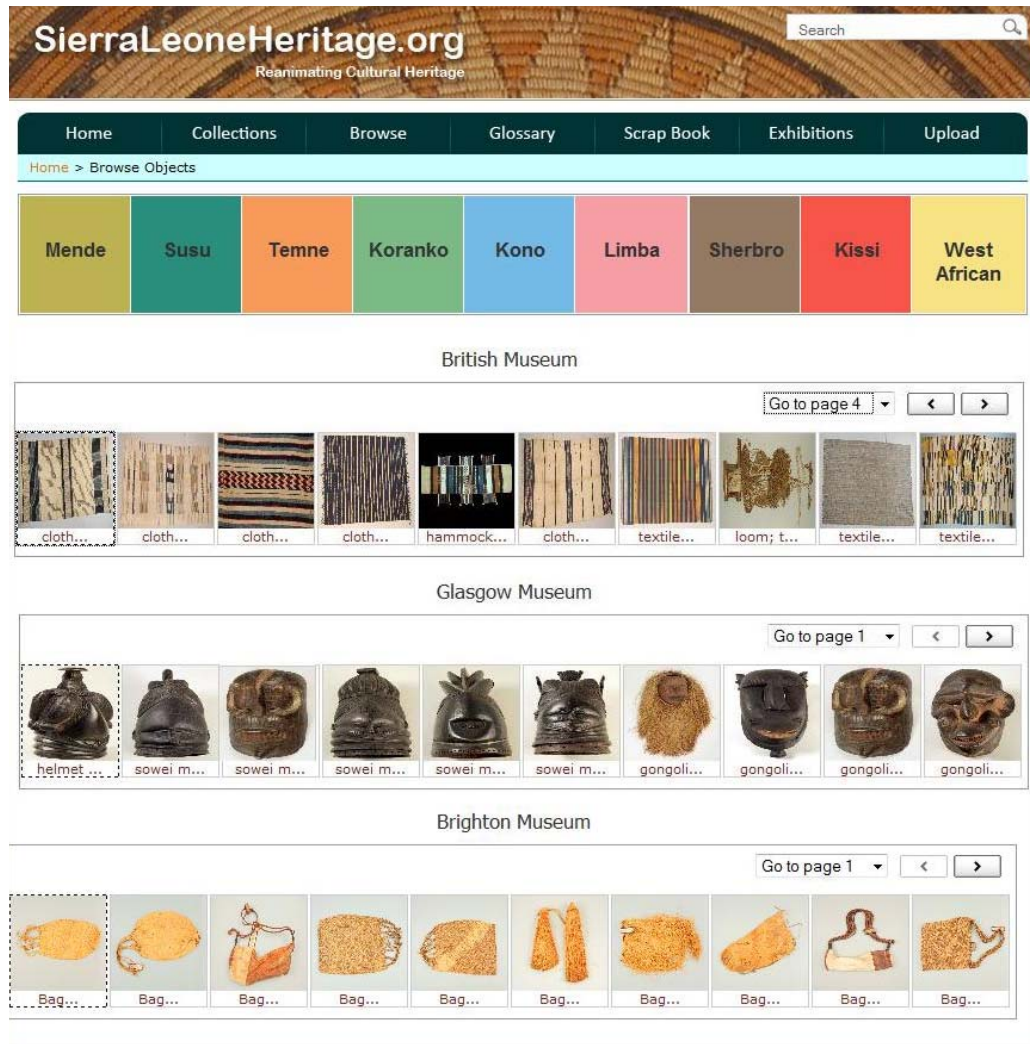


Figure 5-13 RCH Frontend

5.4.1 Design and Analysis of the Presentation Components

The RCH object presentation components were built as independent encapsulated functional modules that include a number of specialised functions. These components involve a number of use-cases that vary in their complexity according to the user actions and the performed tasks. They also involve a set of sequential events to achieve different object presentation objectives as illustrated below.

5.4.1.1 RCH Presentation Components Use-Cases

The presentation components involve a number of use-cases that are illustrated in Figure 5-14. The main presentation activities in RCH are oriented around the process of viewing the details of the stored cultural heritage objects. The process of viewing the details of the stored cultural heritage objects is associated with a number of operational scenarios within the RCH system.

For example, when carrying out search (object retrieval) operations, it is vital to have the necessary means by which the retrieved objects' can be presented to the end-users. Furthermore, basic object browsing operations also require adequate presentation services to enable users to go through the stored archive of cultural heritage objects. Hence, RCH users are likely to need two core object presentation services; a one is involved with object search and retrieval and the other is involved with basic cultural heritage object browsing. The details of the retrieved objects include detailed textual descriptions in addition to their associated images. The use-case diagram below highlights 4 main user actions in relation to object presentation. These actions include: Browse Objects, View Objects and Browse Search Results.

The 'Browse Objects' use-case is involved with browsing the RCH objects through RCH's web interface. This use-case consists of a number of sequential actions starting with opening the RCH website through the user's preferred browser. Then objects are viewed in the objects listing page (the RCH browse page, see Figure 5-13) where thumbnails and brief information about each object are presented to the end-user (for the actual technical details of object presentation, please refer to section 5.4.2).

The browsing operations are complemented with the actions of viewing the underlying details of the objects browsed through the RCH website. The 'View Object' use-case involves firstly locating the wanted object and then viewing its full details via the provided links. Clicking on an object's name or image leads to viewing its full details in a designated dynamically-generated object details page.

Another user operation within this context is the process of viewing the results of the search process. Any search process within RCH is associated with presentation of a basic listing that presents brief details of the retrieved objects. Similar to the 'View Object' use-case, the 'View Search Results' use-case involves viewing the presented list of search results in the form of thumbnails of objects associated with brief textual descriptions. Users can then view the actual object details via the provided navigational links. These links provide an access to the object's details page where the full range of its images and attributes can be viewed. A list of related objects is also provided in the details page for further browsing.

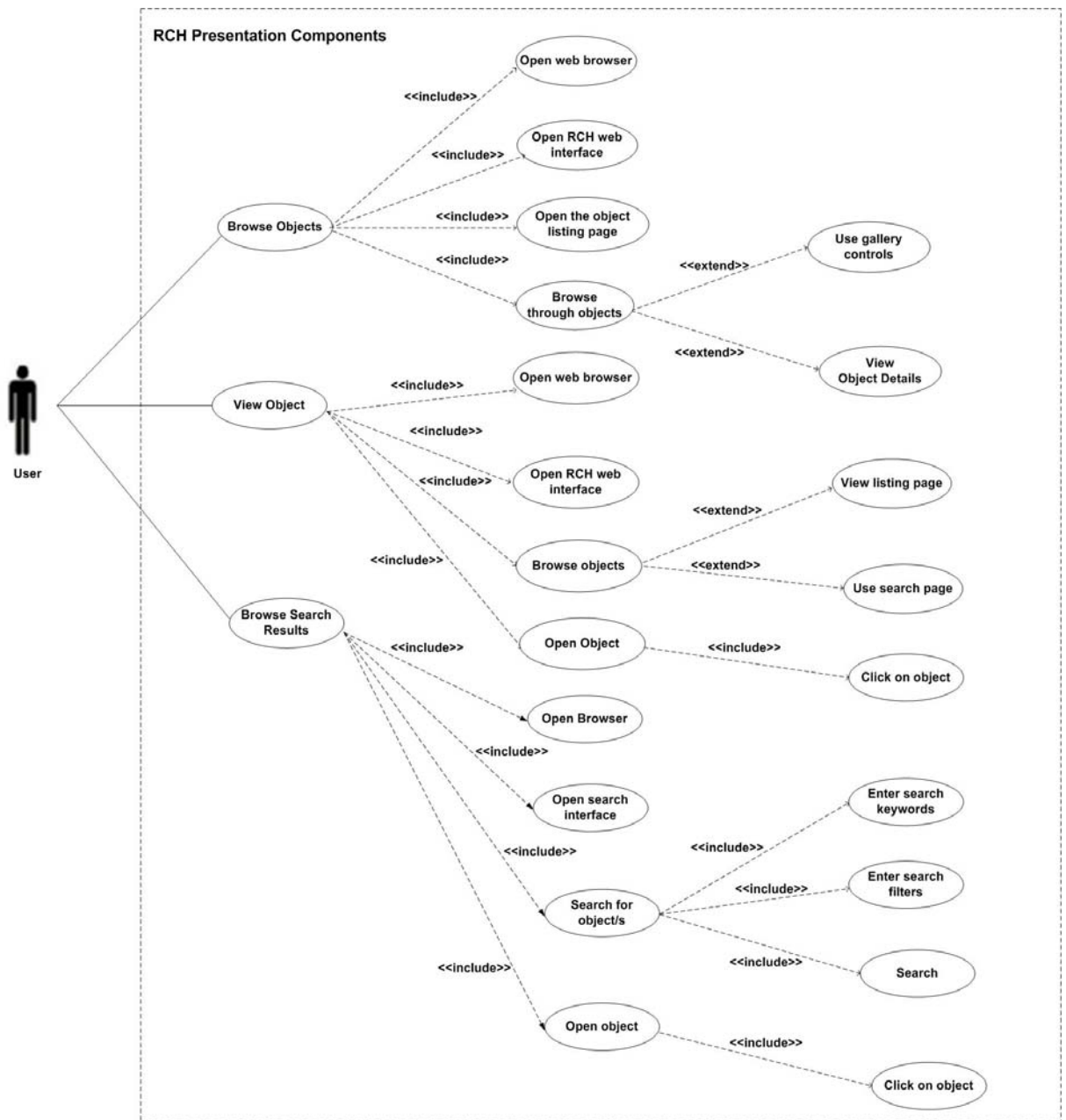


Figure 5-14 Presentation Components Use-Case Diagram

5.4.1.2 RCH Object Presentation Components Structure

If we look at the presentation components from a structural point of view, we will be able to see that they comprise a number of basic sub components that interact with each other to provide the required object presentation functionality. Figure 5-15 represents the presentation components adopted in RCH and their main constituent parts. This model comprises three main entities, which are the Data Repositories, the actual Presentation Model and the Presentation Technology Stack.

The left side of the diagram represents the range of repositories that the presentation model has to deal with to retrieve the required data for accurate cultural heritage object presentation operations. The main repository in the current scenario is the actual RCH repository which interacts with the RCH code components to perform the object presentation operations.

The middle part of Figure 5-15 represents the core presentation sub-components within RCH's presentation model. These sub-components are further categorized into data-related and layout-related modules. The data-related modules include the functionality necessary to connect with and retrieve the required presentation information (object details and images) from the RCH repository. On the other hand, the layout-related modules represent the actual tools used for object presentation such as page themes, dynamically-generated web pages, CSS styling, etc.

Object presentation is not possible without the technologies that would facilitate such presentation operations. Therefore, what is called the 'Presentation Technology Stack' includes a number of web technologies that collectively contribute to the accurate presentation of RCH's cultural heritage objects. These technologies include XHTML for final object presentation within the RCH website, CSS for unified presentation styling throughout the RCH website, XSLT for rendering the information extracted from the archival XML files into valid XHTML, JavaScript for complementary functionality especially image and gallery related layouts and PHP for dynamic content and page generation when needed.

In a typical illustrative scenario, a user would carry out a search process, which will invoke the RCH code classes to retrieve the details of the required object/s (text and images) in coordination with the data-related modules of the presentation model. The layout-related modules then render the required XHTML page to display the retrieved object/s details. This process involves calling the correct XSLT file to retrieve the object. This is followed by applying CSS styling to unify the layout of the generated page with the rest of the RCH website (we are talking here about dynamically-generated content pages). Object presentation is complemented with extra functionality that aims to make the retrieval process more efficient. For example, when viewing the details of any of the retrieved objects, users are presented with links to related objects (if available) for further browsing.

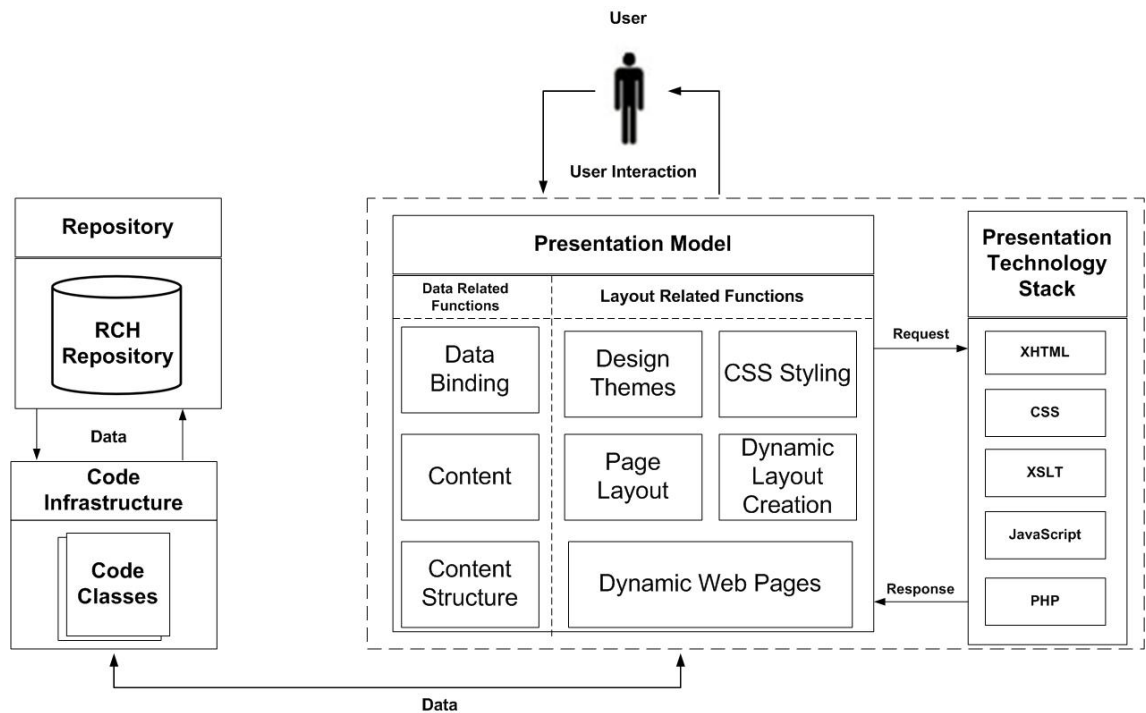


Figure 5-15 The Adopted Presentation Model

5.4.1.3 RCH Object Presentation Sequence of Events

The presentation components perform a number of sequential steps to achieve the desired object presentation operations. The sequence diagram in Figure 5-16 highlights a typical object presentation sequence when a user carries out a search operation through the RCH website interface. This sequence involves the interaction between the user from one side and 3 main components from the other. These components are the RCH Frontend, RCH Logic, UI Generator and the RCH Repository.

The 'RCH Frontend' represents the actual RCH website that users interact with. The 'RCH Logic' encapsulates the functional code modules of RCH which perform the overall RCH functionality. The 'UI Generator' is encapsulating the presentation elements of the RCH website including XSLT files, CSS, JavaScript and PHP which collectively produce rendered XHTML that can be displayed in RCH Frontend.

The illustrated sequence of events in Figure 5-16 starts with the action of opening the RCH website; this will result in calling the UI Generator to display the home page elements to the user to start navigating through the website. As an example in this scenario, the user requests to view a specific cultural heritage object. This request is passed to the RCH Logic which interacts with the RCH Repository to retrieve the details of the requested object. The details of the retrieved object will typically contain

images and XML tags that hold the object's details. These details are passed to the UI Generator, which performs a 3-step operation to render the retrieved data to a displayable form. The first step is to apply XSLT styling to the retrieved XML elements so that valid XHTML is produced out of this process. Then CSS styling is applied to unify the presentation of retrieved object with the rest of the RCH website pages. Finally, the produced XHTML is rendered for final display within the main RCH page template to be viewed by the end-user.

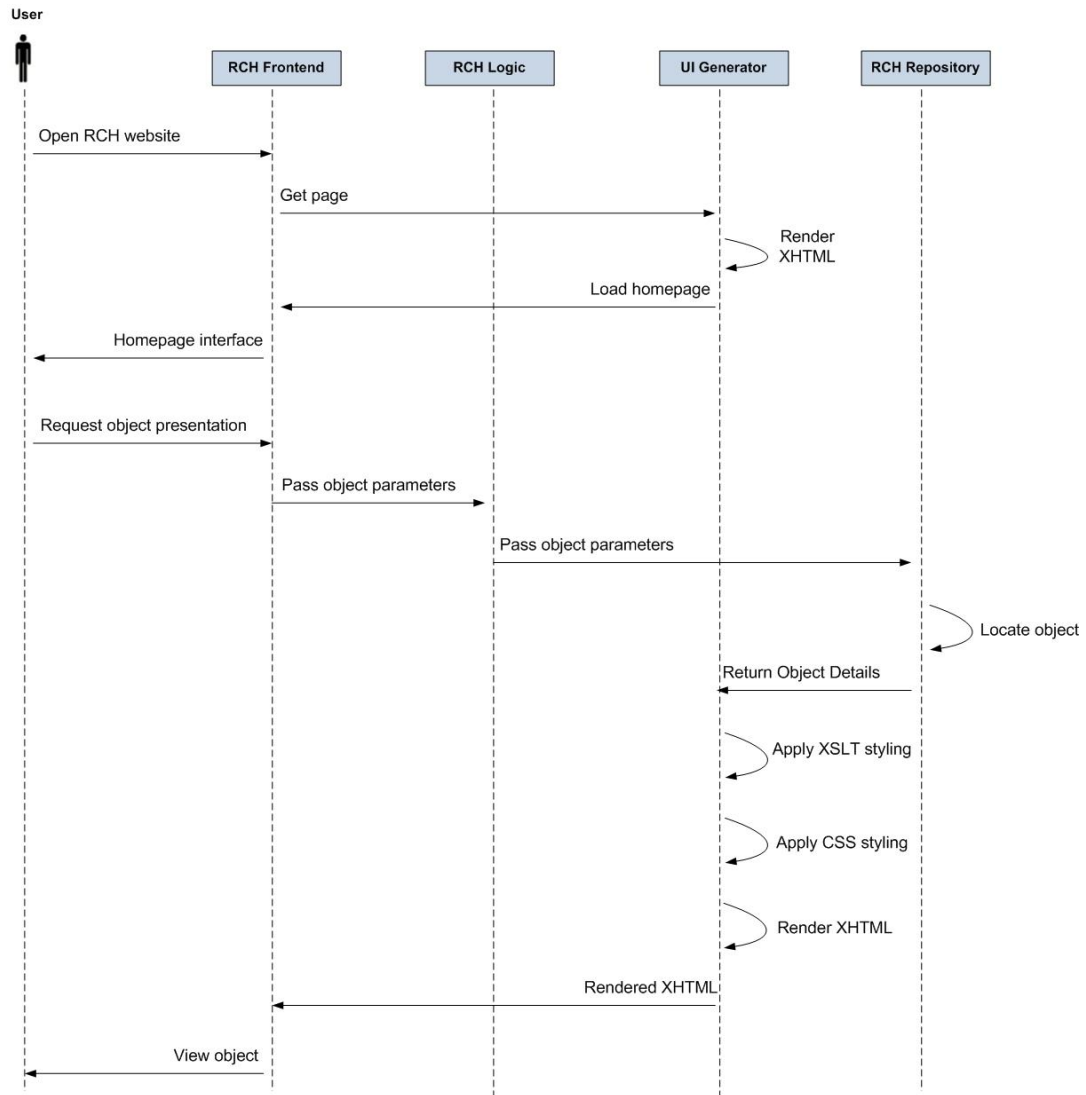


Figure 5-16 RCH Presentation Components Sequence Diagram

5.4.2 Dynamic Content Generation

XSLT is an effective template-based language that has the sole purpose of transforming XML documents to other types of manageable knowledge representations [134].

The use of a combination of XML and XSLT files is a common approach in creating dynamic HTML or XHTML contents. This approach allows for the seamless integration

between the business logic and the presentation layers of any given web application, while providing a model that is easy to build and maintain [136]. Another benefit of this approach is the ability to build multiple dynamic frontends (views) that can serve the different evolving needs of the end-users. So, the same contents can be displayed in different ways to suit the different operational contexts of the system.

The RCH website (the View) receives the user input in the form of HTTP requests, for example as a part of a search query. These requests are then passed to the Model and Controller parts of the adopted MVC model. This process results in the provision of rendered XHTML code that represents a subset of the data stored in the archival XML files (i.e. the data store), note that this could be a database that generates the XML files as in the ARCO system, which can do this through its XML Data Exchange Interface [65] in response to the user actions. Moreover, RCH's frontend handles the necessary parameters for XSLT operation; for example, the search keywords that are used to query the underlying XML files. These parameters are passed to the XSLT files that produce correct XHTML code representations.

An example of parameter passing is the process of passing the search keywords to the RCH's backend through its website frontend. The PHP code snippet below shows that the variable 'Keyword' gets passed to the search file "search.xml" where the search query is handled. The search results are then returned as rendered XHTML pages that are displayed to the user who performed the search operation. Object presentation is the end result of querying the 'britishMuseum.xml' file that holds the data related to the British Museum's collection for instance.

```

$xml->load('search.xml');
$xml_doc->load('britishMuseum.xml');
$xml->setParameter('', 'keyword', $keyword);
$xml->setParameter('', 'numrows', '3');
if ($html = $xml->transformToXML($xml_doc)) {
    echo $html;
} else {
    trigger_error('XSL transformation
failed.', E_USER_ERROR);
}

```

Parameter passing and message exchange is instrumental in achieving some of the system's functionality. This significance is further highlighted in Chapter 6 of this thesis.

5.5 RCH Object Retrieval Components

The data-intensive nature of RCH made it necessary to have powerful retrieval (search and browse) tools to complement its archival and presentation components. The RCH object retrieval operations work in coordination with the other system components within the adopted MVC model. RCH provides its users with a sophisticated search logic that enables them to carry out advanced keyword search operations to locate the objects that they are looking for. The search module that is built-in within the system's XSLT files responds to the users' queries and retrieves the required results from the backend XML files. Moreover, the utilization of XSLT for the search operations was an effective option as it suited the system's XML data infrastructure (XSLT is considered to be among the best mediums for querying XML files [135]). The underplaying details of the object retrieval process and are discussed in Sections 5.5.1.3, 5.5.2 and 5.5.3.

5.5.1 RCH Object Retrieval Design and Modelling

The object retrieval operations within RCH involve a number of key use-cases and events. These events and use-cases are derived from the interaction of RCH users with its user interface as further explained below.

5.5.1.1 RCH Retrieval Components Use-Cases

The actual object retrieval processes within RCH can be categorised into two main use-cases: 'Basic Search' and 'Advanced Search'. As shown in Figure 5-17, a typical retrieval operation starts with opening the RCH website and then utilizing the provided search interface to locate the required objects. A basic search operation involves the sole use of textual keywords to locate the required cultural heritage objects. This is a basic traditional text-based search operation where the entered keywords are compared with the descriptions of the stored cultural heritage objects. Object descriptions are stored within the archival XML files in the RCH repository. Once an object is found, its details are returned to the user after undergoing the necessary PHP, XSLT, CSS and XHTML processing (see Section 5.4 for more details about RCH object presentation).

The advanced search operation differs from the basic search operation in the fact that keywords are combined with a number of search filters (museum collection, cultural group, category and theme) to arrive at a narrower and more accurate set of results. These filters are compared with the actual object details in the archival XML files. The details of the found objects are returned to the end-user via the RCH website frontend for further browsing.

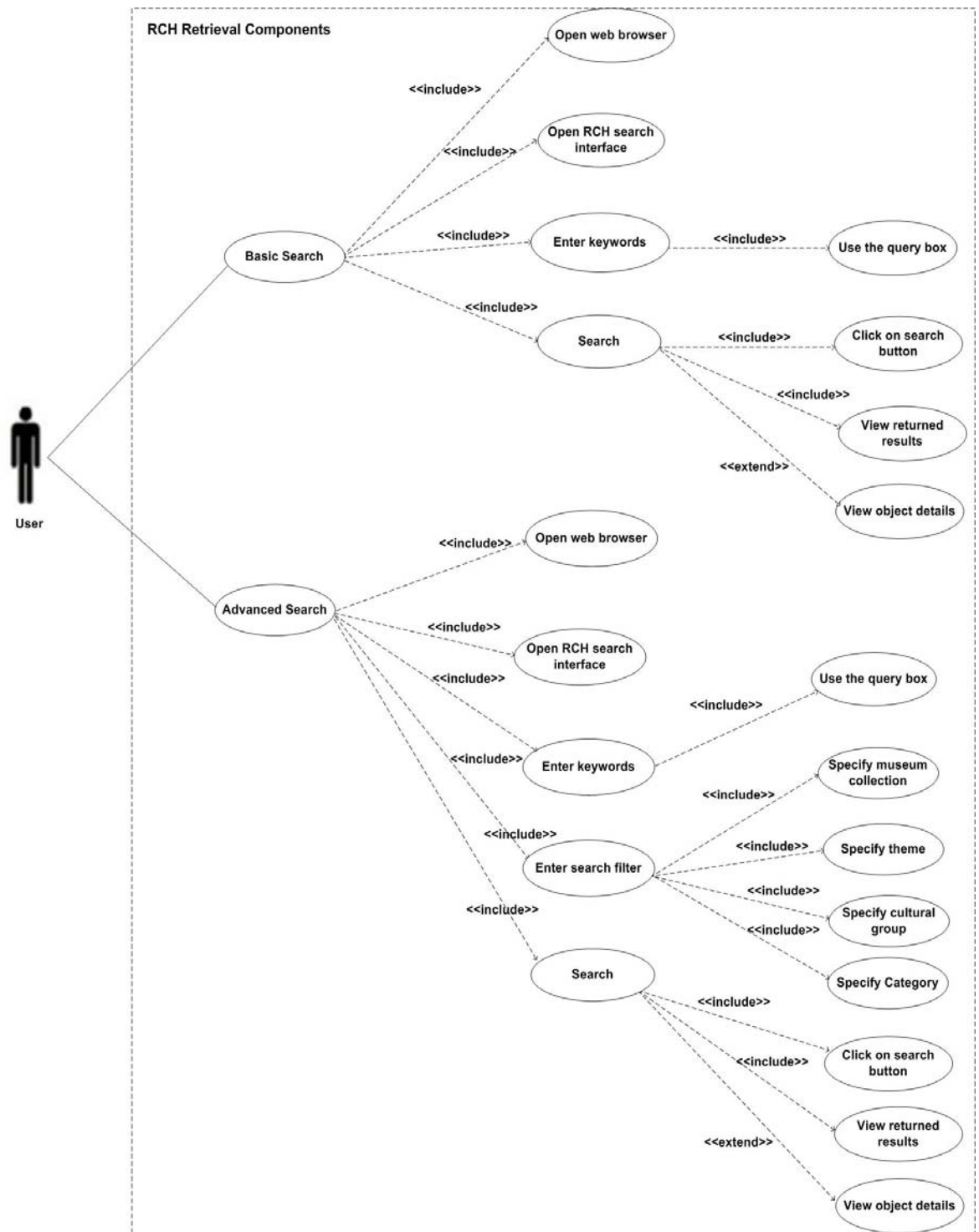


Figure 5-17 Object Retrieval Use-Case Diagram

5.5.1.2 *RCH Retrieval Components Classes*

The object retrieval components consist of a number of specialized classes and code modules that are capable of performing complex object retrieval operations. Figure 5-18 highlights a snapshot of the main classes used in the retrieval process and the relationships between them. This diagram intends to highlight the different components involved in the retrieval process and their internal processes. The involved classes include: Query, objectQuery, Keywords, Attributes, culturalGroup, museumCollection, objectCategory, Theme, SearchPackage, searchFilters, queryBox, objectLocator, imageFinder, Repistory, and resultsInterface.

- **The Query Class**

A typical retrieval process starts with a search query entered by the user via the RCH online search interface. The search interface is encapsulated within the searchIntreface class that receives the query, packages it and passes it to the retrieval classes. The search query itself is initially handled through the Query class that is a generalization of a sub-class called objectQuery.

- **The objectQuery Class**

This class specializes solely in the operation of retrieving the details of cultural heritage objects from the RCH repository. An objectQuery is constructed of two main attributes, which are the 'keywords' used by the user and the actual 'search attributes' used to narrow down the search results. According to the RCH structure, 4 types of search filters (attributes) exist which are the: Theme, Cultural Group, Museum Collection and Category. Some or all of these filters can be used in the retrieval process. The combination of the keywords and the search filters represent the final search query. Search queries are rendered into search packages as explained below.

- **The searchInterface Class**

The search queries are passed through the searchInterface class that creates instances of the searchPackage class. The searchInterface class is composed of two main sub-classes, which are queryBox and searchFilters. An instance of the queryBox class represents the actual field used to collect the search keywords entered by the user. The searchFilters class collects and handles the combination of search filters entered by the user.

- **The queryBox and searchFilters Classes**

These classes represent the visual elements of the searchInterface class and are used to collect the search attributes entered by the user.

- **The searchPacakge Class**

A searchPacakge is basically the combination of the entered keywords and search filters alongside any other attributes entered by the user during the search process. The generated search packages are passed to the Searcher class that is the central functional entity within the retrieval components of RCH.

- **The Searcher Class**

The Searcher class simultaneously processes the searchPacakge instances and returns the retrieved results to the end-user. The Searcher class uses two main classes to perform the retrieval operations. These classes are the objectLocator and the imageFinder classes. These classes retrieve the details (text and images) of the required objects according to the attributes of the searchPacakge class.

- **The objectLocator Class**

The objectLocator class specializes in retrieving the textual information related to the object/s being searched. This class basically queries the XML files that hold the details of the available cultural heritage objects. Once an object is found, its details are retrieved such as name, description, cultural group, owner, etc.

- **The imageFinder Class**

The imageFinder class locates the images related to the objects being searched. Images are handled separately due to their special nature and the fact they are stored in special directory structure within the RCH repository. This class locates the group of images associated with each object to be displayed along its other details to the end-user.

- **The resultsInterface Class**

Once the results are retrieved by the Searcher class, they are passed to the resultsInterface class which renders them for final display. This class is a part of RCH's presentation components mentioned in Section 5.4. This class applies the final XSLT, CSS and PHP processing to present the search results as fully optimized XHTML.

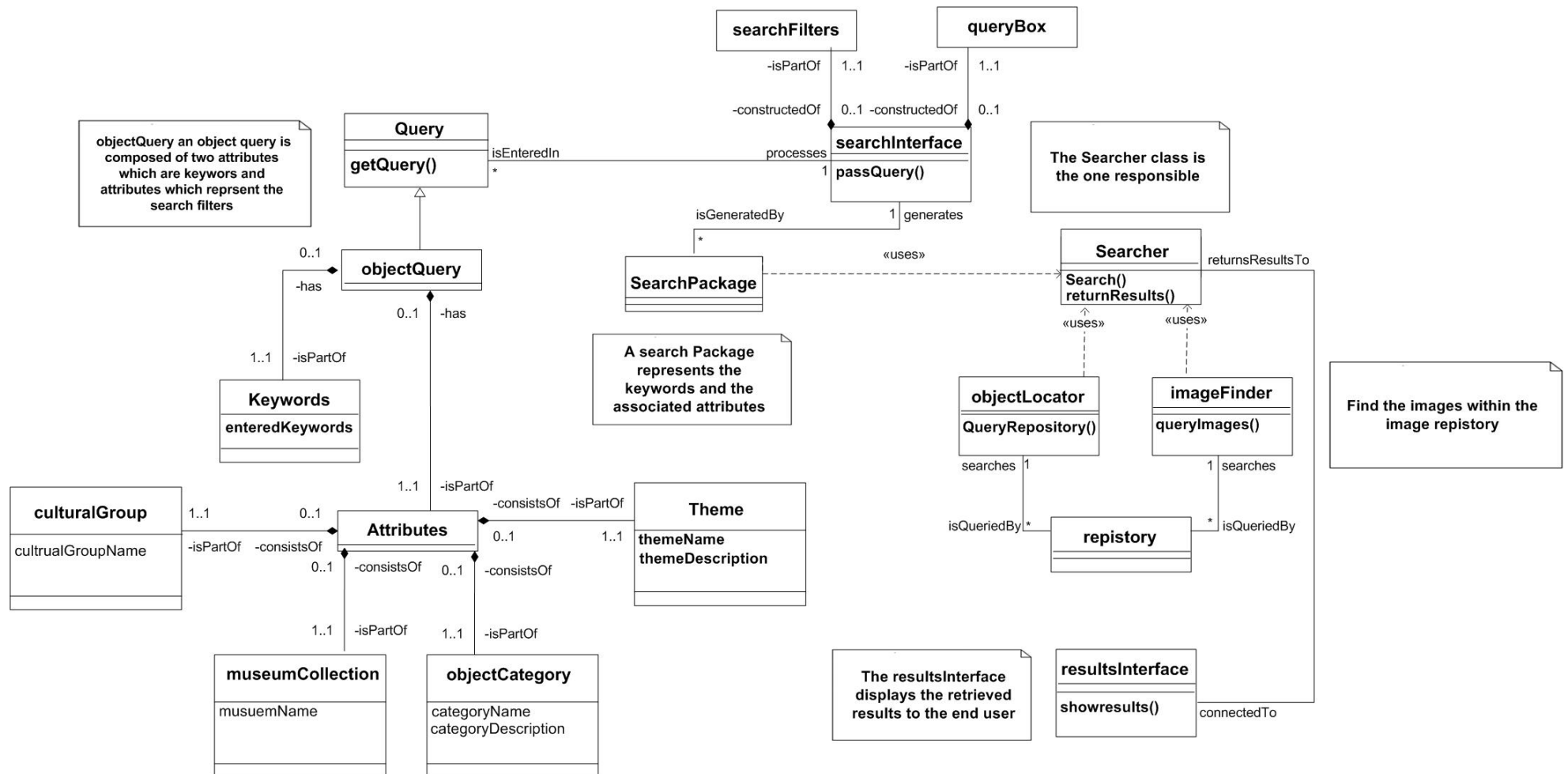


Figure 5-18 Object Retrieval Class Diagram

5.5.1.3 RCH Object Retrieval Sequence of Events

As RCH applies the MVC design pattern in its architecture, its object retrieval components fall under the Controller part of the adopted model where all the search and retrieval oriented operations are conducted. Therefore, the Controller component allows the search logic to interact closely with the View component where the Controller reads the user queries that come through the system's website (the View).

Search queries are consequently passed to the data retrieval components which reside within the Controller part of the system. The Controller queries the Model and this process results in returning the search results to be displayed in the View. This sequence is further highlighted in Figure 5-19 that shows how the different system components communicate with each other within the context of the data retrieval process.

The retrieval process sequence diagram illustrated in Figure 5-19 involves the interaction between three main components which are the RCH Frontend (the view), the XSLT files (the controller) and the XML archival files (the model). The sequence diagram shows that the whole retrieval scenario is initiated with a user's request to retrieve specific object/s by means of submitting a query through the RCH's frontend (the View). A search process may be a basic keyword-based search operation or an advanced search operation. An advanced search operation comprises keywords in addition to the chosen search filters for a more accurate set of results.

Search queries are passed to the XSLT search module (the Controller) for further processing. The entered search terms are compared with the actual contents of the archival XML files resulting in extracting the details of the matching cultural heritage objects. The actual retrieval process is based on the devised search logic within the XSLT files. Designated retrieval XSLT files loop through the actual contents of the XML files and retrieve the required objects' details.

The retrieval process is combined with the process of wrapping the retrieved information (in the form of raw text) with the appropriate XHTML tags. This process aims to make the retrieved objects suitable for web display. The rendered XHTML is returned to RCH frontend to be displayed to user who initiated the retrieval process. CSS styling contributes to making the retrieved objects conform to the overall look and feel of the RCH website.

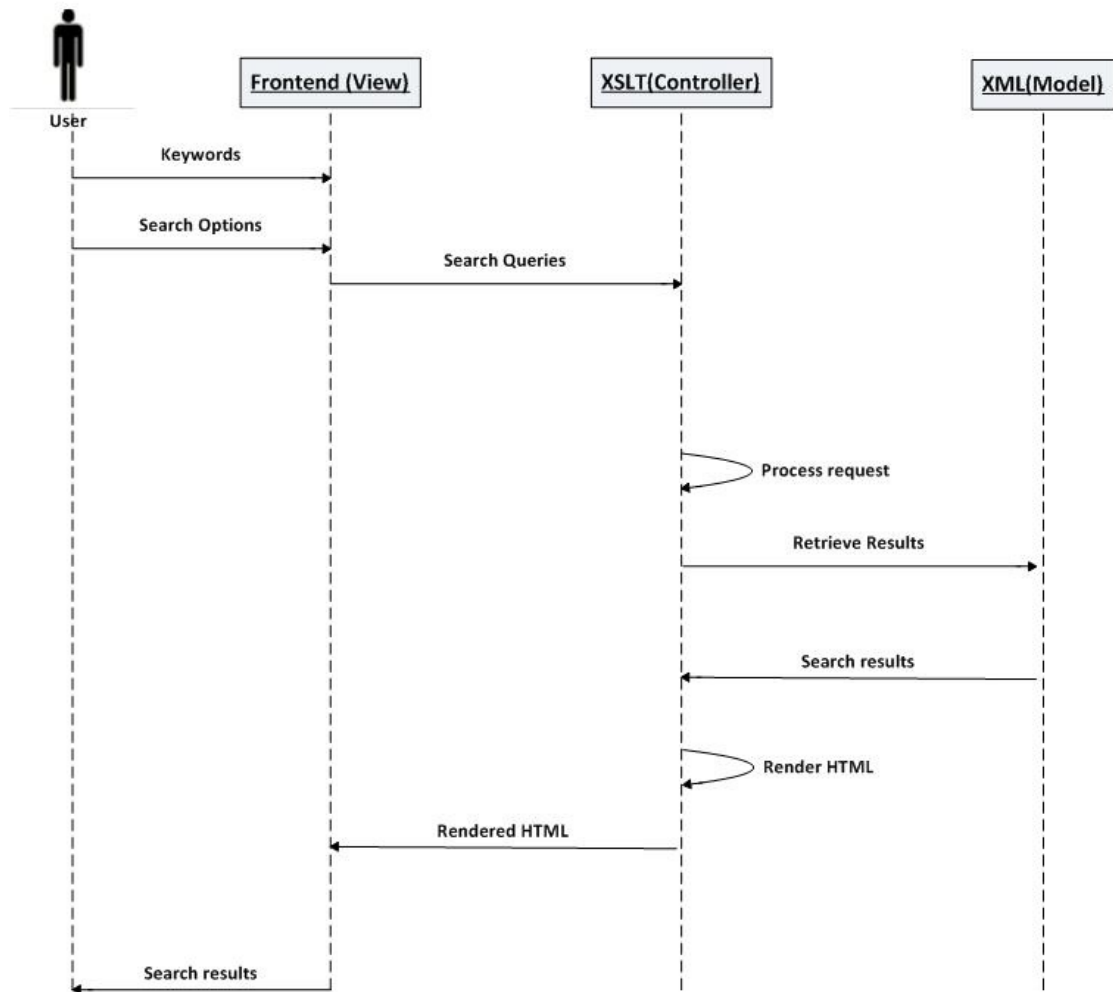


Figure 5-19 The Object Retrieval Sequence Diagram

5.5.2 RCH XSLT Dynamics

When a search operation is conducted, the specialized search XSLT files process the corresponding XML files to retrieve the search results to be displayed in RCH's frontend. The built-in search modules loop through the XML nodes while returning the ones that match the search criteria used. The search process itself can be a plain keyword-based search, or a more advanced process whereby the keywords are associated with a number of attributes and filters to narrow down the returned results. XSLT files are used optimally by applying a combination of template matching 'match' functions and XPath expressions. These expressions are used to extract data from the target subtrees within the designated data XML files. For example, the "contains" expressions are used to perform the straightforward keyword search operations as highlighted in the code snippet below:

```

<xsl:when test="contains($selector,'GLA')">
  <div class="object-details">
    <p>
      <span class="secondary-name">
        <a href="{ $olink}" target="_self" title="{ $n1}">
          <xsl:value-of select="Object"/></a></span></p>
      <p><span class="label">
        Culture:</span>
        <xsl:value-of select="cultureGroup"/></p>
      <p><span class="label">Category:</span>
        <xsl:value-of select="objectCategory"/></p>
      <p><span class="label">Museum:</span>Glasgow Museum
    </p></div>
  </xsl:when>

```

The code snippet above illustrates that the “contains” expressions are used to locate objects within the stored collection of the Glasgow Museum. The same is applied to the other museums when there is a need to search their collections. Moreover, content filtering (Museum, Tribe, Category, Theme, etc.) can be programmatically achieved through the introduction of a series of XSLT ‘if’ and ‘contains’ expressions that allow for the accurate incorporation of search filters. The above code snippet also illustrates the actual process of extracting the textual information of the found objects while wrapping them with the appropriate XHTML tags. Such tags include <p>, <div>, etc., which aim at displaying the retrieved objects correctly within the RCH website as can be seen in the screenshot in Figure 5-21.

The code snippet below shows another example that illustrates the process of outputting an image as well as a link to view the full details of a cultural object, where the (to display the image) and <href> (for a dynamic link) tags are used in conjunction with variables extracted from the queried XML file/s.

```

<xsl:when test="contains($selector,'GLA')">
  <div style="width:85px; height:105px; float:right;" >
    <a href="{ $object}" >
      </img>
    </a>
  </div>
</xsl:when>
<p>
  <span class="label">Category:</span>
  <xsl:value-of select="ObjectCategory"/>
</p>
<p>
  <span class="label">Museum:</span>Glasgow Museum
</p>
</div>
</xsl:when>

```

It should be noted here that it is not the intention of this thesis to build an efficient and sophisticated search engine using XSLT. The intention was more to build a set of components (i.e. a test bed) based on the archival and presentation concepts discussed in the previous chapter that could be used to test the validity of workflows (see Chapter 6).

5.5.3 The RCH Search Interface

The RCH website provides the necessary tools to facilitate the data retrieval process. Keyword-based searches can be made more accurate by means of associating a number of attributes including object category, museum, theme, etc. The main prototype search interface is highlighted in Figure 5-20.

SierraLeoneHeritage.org
Reanimating Cultural Heritage

Search

Home Collections Browse Glossary Scrap Book Exhibitions Upload

Home > Collections Search

Word Search

Object Category ☒ SEARCH ALL

- ☒ Furniture
- ☒ Magic and Religion
- ☒ Narcotics
- ☒ Currency
- ☒ Musical Instrument
- ☒ Transport
- ☒ Personal Adornment
- ☒ Container
- ☒ Masques and Drama
- ☒ Tools
- ☒ Art
- ☒ Textiles

Museum Collections ☒ SEARCH ALL

- ☒ British Museum
- ☒ Brighton Museum
- ☒ Glasgow Museum

Theme ☒ SEARCH ALL

- ☒ Theme1
- ☒ Theme2
- ☒ Theme3

Sierra Leone Culture Groups ☒ SEARCH ALL

Map of Sierra Leone Culture Groups:

- Susu
- Limba
- Koranko
- Kono
- Mende
- Sherbro
- Creole
- Other tribes

☒ Mende ☒ Limba

☒ Temne ☒ Sherbro

☒ Koranko ☒ Kissi

☒ Kono ☒ West African

Facebook YouTube Flickr Wiki Blog Email

Copyright © 2010 Reanimating Cultural Heritage Project. All rights reserved. About SierraLeoneHeritage.org

Figure 5-20 The RCH Search Interface

The rendered XHTML results are then displayed in the search results page, as illustrated in Figure 5-21.

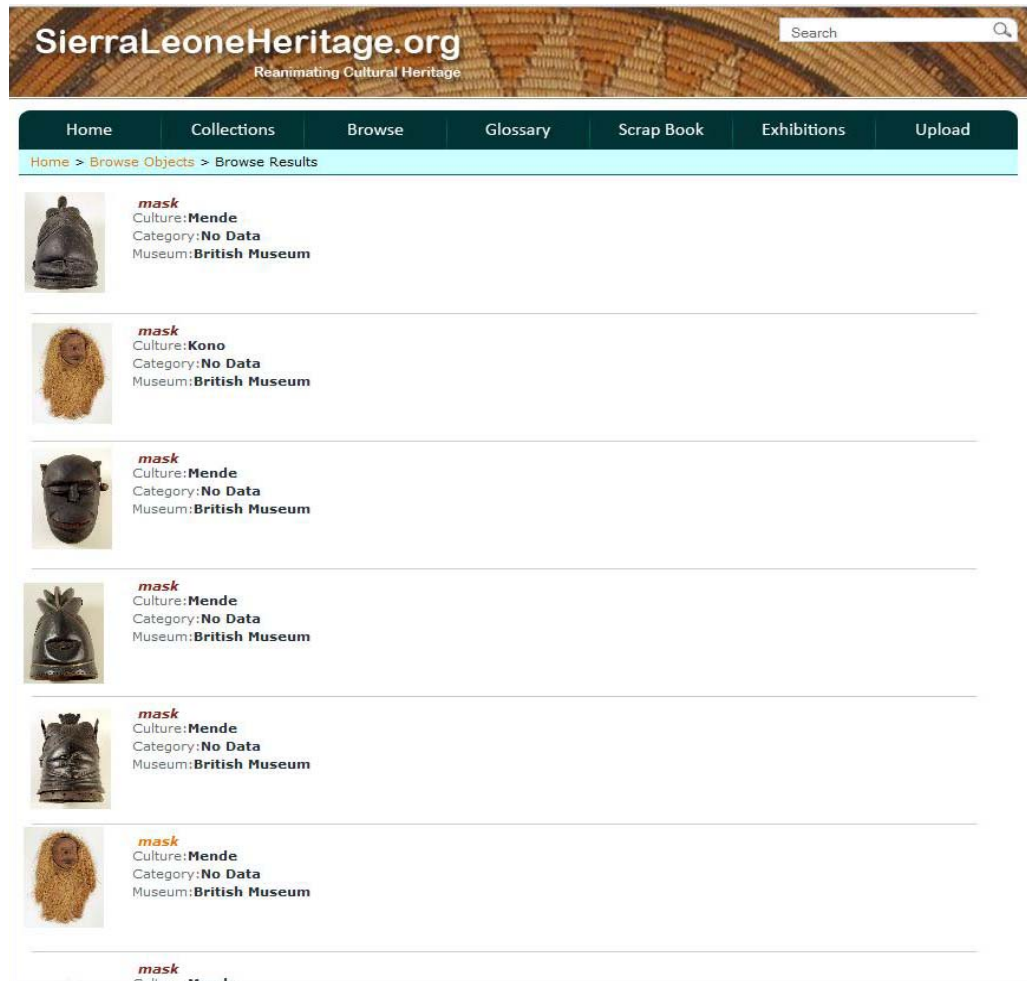


Figure 5-21 The Search Results

The interface of the results page shows a summary of the retrieved objects. Clicking on an object's image opens an enlarged version of it with a brief summary of the object details, as shown in Figure 5-22.



Object:
sowe mask

Description:

Sowe mask, used in Sande society, Mende, of blackened wood, 5 fan-ridge hair style with white metal strips nailed along each ridge edge. Carved in a simpler style which Ruth Phillips thought at first was a Gondei style, but the mask has too much tin plate ornament for this. It is made from a heavier wood than the other sowe masks, a hardwood rather than a cotton tree wood.

[View Record](#)




Figure 5-22 Enlarged Object View

Clicking on “View Record” opens a page containing the full details of the object, including a full textual description and a number of important attributes (material, production date, source, etc.), as highlighted in Figure 5-23

[Home](#)
[Collections](#)
[Browse](#)
[Glossary](#)
[Scrap Book](#)
[Exhibitions](#)
[Upload](#)

Home > [Browse Objects](#) > Browse Results









Glasgow Sierra Leone Objects: Images



sowei mask



Object:	sowei mask
Description:	Sowei mask, used in Sande society, Mende, of blackened wood, 5 fan-ridge hair style with white metal strips nailed along each ridge edge. Carved in a simpler style which Ruth Phillips thought at first was a Gondei style, but the mask has too much tin plate ornament for this. It is made from a heavier wood than the other sowei masks, a hardwood rather than a cotton tree wood.
Materials:	wood, metal
Culture Group:	Mende
Dimensions:	overall: 400 mm x 190 mm x 220 mm 1853 g
Production Date:	No Data
Associated Places:	Africa, West Africa, Sierra Leone, Mende (place associated)
Creator:	No Data
Source:	Julian and Judith Massie-Taylor and Vivien Scarth
Collector:	No Data
Museum:	Glasgow Museum
Accession Number:	GLAMG:A.1985.13.ac


Related Objects











sowei m... sowei m... sowei m... **sowei m...** sowei m... sowei m... sowei m... sowei m...

Comment:

 Like
  Be the first of your friends to like this.



 Logged in as Alan Smith

Add a comment...

☒ Post comment to my Facebook Profile


 Facebook social plugin

Figure 5-23 Detailed Object View

5.6 Summary

This chapter provided an insight into the technical aspects of a DLS implementation (RCH) constructed to validate both some of the DISPLAYS concepts, but more importantly in the context of this thesis, to act as a test bed for validating the use of workflows (see Chapter 6). It covered the adopted MVC design pattern and the ways by which it succeeded in separating the different system code and design components. This separation paved the way for achieving modularity in terms of the core system components (archival and presentation (with its associated retrieval components)). These components work in coherence with other components within a custom workflow management infrastructure, as will be detailed in Chapter 6.

All the RCH components are architected within a hierarchical MVC design pattern that separated the business logic, data and presentation aspects of RCH. The object presentation components formed the View part of the model and actively integrated with the Controller part. The Controller part is represented in the data retrieval components and their associated XSLT files. The Model part is constructed from the system's archival backend (XML files) which allowed for the creation and expansion of a shared digital heritage resource, one of the main goals of the RCH system.

The highlighted RCH components are carefully constructed and uniquely integrated to provide a fully functional RCH implementation model as a prototype system, which in the context of this thesis will be used in Chapter 6 to validate the workflow approach. This RCH system is now being further developed into a fully operational model that covers the DISPLAYS services. The RCH archival tools provided the necessary data storage capabilities for the participating museums, complemented with advanced data mapping and validation tools. These tools enabled the creation of a shared resource by allowing the participating museums (or other entities such as ARCO) to submit their data to RCH while converting it to a custom XML format (RCH format). This process is not straightforward due to the complexity and diversity of the data that each museum possesses which require highly customized mapping tools. The created RCH mapping tools allowed the communities of practice to export/import Sierra Leonean digital heritage object data to and from the RCH repository.

The RCH object retrieval tools provided the necessary data retrieval functionality by querying the XML archival files and returning the corresponding results to the system users. These tools allowed for sophisticated data retrieval operations by exploiting XSLT files that query the system's XML archival files. The object retrieval tools act to transform a subset of the stored data to be displayed in the system's website frontend based on user queries. Moreover, the search logic is capable of handling a combination of search keywords and filters paving the way for advanced retrieval options for the end-users.

The object presentation services represent UI of the system that displays the augmented data into a unified website frontend that is highly dynamic and interactive. The contents of the RCH website are driven from the RCH's XML files that acted as the main data storage medium of the system. The RCH website also provided the search interface to query the available data objects and display the returned rendered XHTML in a highly

structured way. Search queries are rendered as HTTP requests that are treated by the search logic (in XSLT files).

The prototype RCH system has proved to be a successful representation of a functional online digital heritage resource that can be accessed and utilized by a number of communities of practice with a different set of data sharing and distribution needs. The separation between the presentation, data and business logic parts of the system by adopting the MVC model proved to be a practical approach especially in terms of providing multiple user interfaces (views), while relying on a single backend data model (XML files). Also, the utilization of XSLT files as a part of the RCH's MVC model Controller has proved to be a natural choice as it smoothly integrated with the system's XML files. This integration allowed for advanced and fast data manipulation operations, including search and retrieval operations that responded to the parameterized search queries coming through RCH's frontend.

CHAPTER VI

6 Hosting RCH as a Workflow

This chapter discusses an innovative workflow management solution that was devised to manage the convoluted components of the RCH prototype discussed in Chapter 5. The RCH Workflow Management System (WfMS) was designed and implemented to validate the proposed concept of workflow management integration within DLSs and DHRs (as discussed in Chapter 3 within the context of the DISPLAYS services). The architectural approach used in the devised WfMS is discussed in relation to the managed system components (archival, retrieval and presentation), which were discussed in detail in Chapters 4 and 5. The way by which these loosely-coupled components were hosted within a WfMS hosting environment is also discussed while exploring the actual building blocks of the RCH WfMS.

Another important aspect of the RCH WfMS was the concept of workflow hosting (hosting of the workflow runtime services), which has materialized into a standalone host application. This application that incorporates workflow technology is called the RCH Content Management System (RCMS), which provides the actual WfMS UI (itself a prototype to prove the concept of workflow management, see Figure 6-10). The discussed technical details are complemented with a number of test scenarios that aimed at assessing the efficiency of the devised workflow management solution. Testing involved the process of examining the different functional areas of the RCH WfMS including message passing, service invocation, workflow monitoring, etc., as shown in Section 6.6.

6.1 Introduction

RCH is a digital repository that interacts with a disparate set of users and systems (for example, collection management systems in museums). Additionally, RCH depends on a number of data-centric operations (e.g. archival and retrieval operations) that require dynamic and complex data processing and mapping operations, as outlined in Chapter 4. The complexity and intersecting nature of the RCH components made it a good case with which to experiment (i.e. test the validity of workflows for this type of system), and provide an innovative solution for managing the system's workflows. Such a

solution needed to suit the underlying RCH SOA as well as the way that its different components operate and interact with each other. The concept of the devised solution is oriented around the idea of providing an embedded workflow management component, as will be explained in Section 6.3.

Integrating a WfMS with RCH had a number of goals including:

- the provision of a layered WfMS that is capable of managing the RCH components;
- maximizing the underlying usability of the RCH resources;
- enhancing RCH's scalability and ability to handle multiple parallel running workflows;
- the effective support and utilization of the SO nature of RCH that is based on the DISPLAYS framework;
- enhancing RCH's flexibility, scalability, interpretability and expandability;
- providing RCH's users with a management UI to enable them to interact with and utilize the RCH components.

The devised solution was built to provide a flexible medium to host and manage the RCH modular components (archival, retrieval and presentation). Figure 6-1 illustrates the idea of hosting the RCH components in a workflow management host. The workflow host comprises a number of services that perform the different workflow management tasks (more on this in Section 0). Effective communication between the workflow host and the RCH components is maintained via message passing (request parameters and other commands) as illustrated in Section 6.6.2.

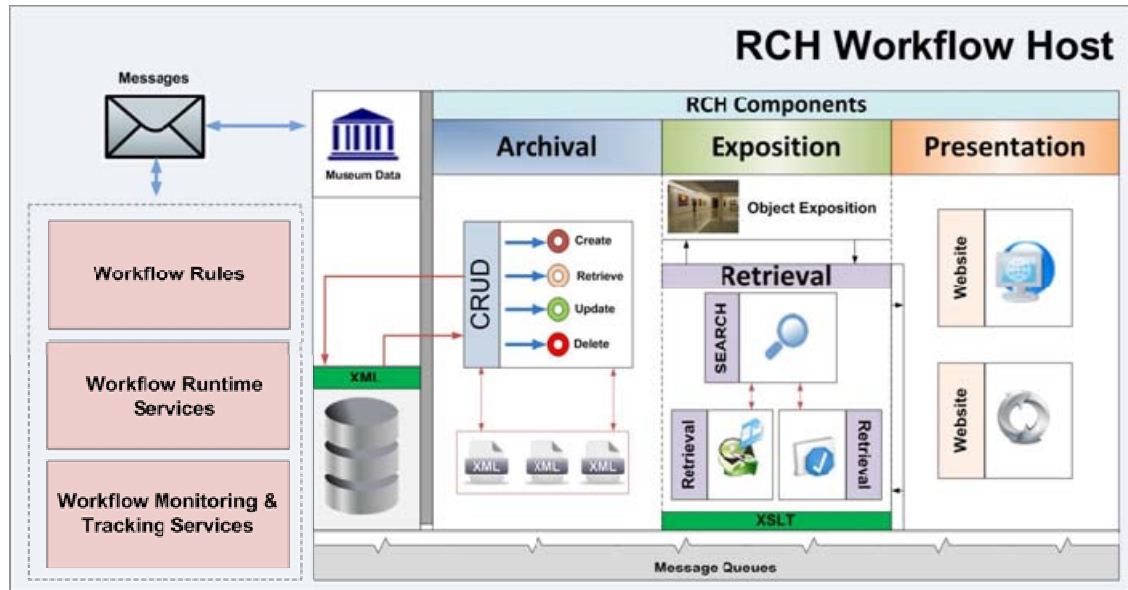


Figure 6-1 Hosting RCH Components in a WfMS

6.2 Solution Formulation

The implementation of the RCH WfMS took the form of an integrated dynamic workflow solution that hosted the RCH components. The devised solution provided a UI element, which acted as a content management medium for RCH; this was called the RCH Content Management System (RCMS) as detailed in Section 6.5. One of the main concepts behind the RCH WfMS implementation was the process of hosting it within an appropriate workflow host. This paradigm is based on the idea of deploying workflow runtime services by the WfMS based on user actions. For example, Service invocation is enabled via a UI that enables the end-users to call the services that they need, such as the search and retrieval services. The UI itself (RCMS) acts as a host for the RCH WfMS while giving it the ability to interact with the end-users. In the prototype illustrated in this chapter, the UI is implemented as a simple windows desktop application for convenience (see Figure 6-10). In reality, this could be a web page admin function, which is normally accessed by an admin link or typing /admin from the root URL.

The RCH WfMS host application hosted the devised workflow management components (including the workflow runtime services as shown in Section 6.4.1). Based on this, the devised WfMS works at a higher level above the RCH components where it manages their operation and interaction with each other. Such an approach allows for better user control over the managed RCH's components, which are – in this case – the archival, retrieval and presentation components.

The first layer of the workflow solution formulation was to host the SO RCH components within the designed workflow management ‘middle layer’. In such an approach, the effective message passing between the WfMS and the system components contributes to a coordinated and managed operation of the different system processes. It is worth noting here that some of the core workflow management processes are also prompted and managed through message passing and exchange, as will be illustrated in Section 6.6.2.

The main workflow management services and activities within the RCH WfMS revolve around the following areas:

- **Service Invocation**

Service calling, or invocation, has the purpose of invoking the RCH services according to user actions. User interaction, which leads to service invocation, is achieved through the RCMS. This interaction can be in the form of a button click, opening a form, etc. Service invocation leads the devised WfMS to call the services of the concerned system components; for example, initiating the object mapping process as a part of the archival process (see Section 5.3.2). It is worth noting here that the RCH system components may, in turn, call services of some other system components. For example, the heritage object search and retrieval operations depend on the coordination between the object retrieval and presentation components, as was detailed in Sections 5.4 and 5.5.

- **Managing Data Flow between the RCH Components**

RCH is a data-intensive application as data processing and exchange come at the forefront of its operations. RCH’s data model was discussed in detail in Chapter 4, where it was indicated that RCH’s functionality revolves around its lite XML-based data model. Therefore, the system’s service calls are associated with certain parameters that are needed by the called services. For example, a service call to map one museum’s data to another should be associated with the source and destination museum parameters. These parameters (and many others as shown in Section 6.6.2) form a part of the data flow that accompanies the movement of the data objects within RCH.

- **Service Response (Feedback)**

Once a service is called, it performs its designated actions (data mapping, object retrieval, etc.). These actions can be associated with the production of certain data items that are passed to the workflow host while containing some indicators (for example, process execution confirmation), or data to be returned to the end-user (for example, mapped object files). This data is then presented to the end-user or consumed by other system components or processes if needed.

- **Workflow Runtime Service Invocation**

Workflow invocation refers to the process of invoking or creating a workflow instance based on the system's events [137]. The process of invoking the system workflows within RCH is the result of accessing the workflow runtime services via the workflow host application (more on this in Section 6.4.1). This action results in initiating the different workflow activities that are defined within the RCH WfMS, as will be illustrated in the testing scenario in Section 6.6.

- **Message Passing and Exchange**

According to Huang [138], message passing is one of the most important elements when implementing WfMS in SO environments. This significance is driven from the fact that message passing plays the instrumental role of initiating and coordinating the managed system's services. Message passing is conducted between three main components: the RCH WfMS, RCH components, and the RCH WfMS host, as illustrated in Section 6.6.2.

- **Correlation**

The concept of workflow correlation is described by Schmitz and Hanemann [139] as the process of addressing the correlation of events as they are reported from a system's management tools. This concept was explored by Blewett [140] where he indicates that workflow correlation largely depends on effective message passing between a caller, which can be a custom UI, and the workflow host, which responds by initiating the appropriate workflow runtime instances to run. In RCH's SO context, correlation applies to the process in which messages are passed to the WfMS to launch the indented workflow instances.

Figure 6-2 illustrates that a typical workflow scenario in RCH is initiated by a user action. This action can be in the form of a search query for instance, submitted through the system's UI (the WfMS host). The RCH WfMS that hosts the main RCH components (archival, retrieval and presentation) receives the user's requests and consequently initiates the appropriate workflow runtime services. For example, in the context of a search and retrieval operation, the RCH retrieval component is invoked, and this results in running a retrieval workflow runtime instance that will handle the retrieval process from start to finish. To present the search results, an instance of the presentation workflow runtime service is invoked and run. The operation of the archival, retrieval and presentation components is governed by a set of sequential workflow management definitions as further illustrated in Section 6.4.1.

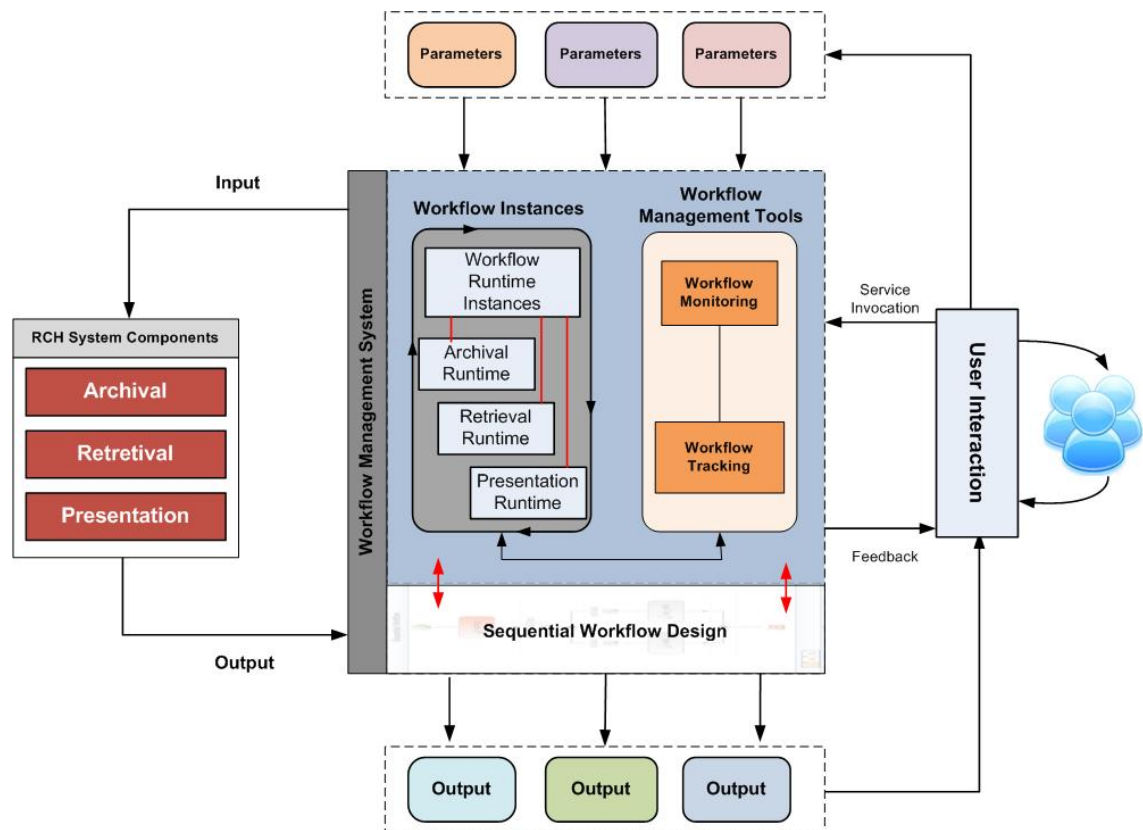


Figure 6-2 RCH Components' Interaction with the WfMS

6.2.1 The Adopted Workflow Model

The implementation of the RCH WfMS was based on a sequential workflow rule-based model. Sequential workflow implementations involve the creation of workflow models that are based on the sequence of the managed system's events [141]. Moreover, sequential workflow management applications are based on a declared set of steps that

are executed according to a certain order. The actual flow control in this model is defined by using a number of custom code constructs, including loops and if-else branching statements. Such constructs are considered to be among the most commonly used techniques in this kind of workflow model [98]. For example, the RCH archival components, including the data mapping tools, were mapped into the devised WfMS where the different steps and parameters of the mapping process were identified and controlled.

The defined execution sequence was also associated with the execution rules that controlled the sequential flow of the system's processes. Consequently, the independent SO RCH components were controlled by the workflow management components following a specified sequence of events. The workflow design is further detailed in Sections 6.4.26.4.3 and 6.4.4

6.3 RCH WfMS Implementation

The current implementation of the RCH system spans a number of different implementation technologies that collectively provided the different RCH functional components and services. As outlined in Chapter 4, the actual system implementation involved the utilization of PHP for the production of the system's web-based functionality, in conjunction with XML and XSLT to provide the system's data-oriented services. The RCH components were implemented based on the MVC design pattern that supported the SO nature of RCH, as explained in Section 5.2. Therefore, it was imperative to choose a WfMS implementation technology that would fit seamlessly with the other RCH components.

6.3.1 WfMS Implementation Technology

The chosen implementation technology was the Windows Workflow Foundation (WF), which is a part of the .NET Framework family of products. It was used to build the core WfMS that controlled the different RCH components. Incorporating WF with an existing online code infrastructure is a common approach and can be seen in a number of similar implementations [142]. Therefore, resorting to using WF suited the loosely-coupled nature of the RCH components and the way they communicate with each other, as further detailed in Section 6.4.

According to Roy [102] there are a number of features that set the WF apart from other WfMS development tools. Some of these features are particularly relevant to the workflow management needs of RCH. These features are related to workflow management: hosting, tracking infrastructure, extensible activities, programmability and workflow designer [102], and are listed in Table 6-1 below. The column named ‘Significance for RCH’ outlines their significance in relation to the workflow management needs in the context of the RCH implementation.

Table 6-1 WF Features

Feature	Description	Significance for RCH
Hosting	The ability to host the created workflow runtime services in client applications. The host application can be either a website or a normal application in the form of a management utility or a CMS.	The flexibility to provide the RCH users with multiple workflow-managed interfaces (MVC Views). The particular scenario adopted in this thesis is to devise a standalone management application for testing purposes as illustrated in Section 6.5.
Tracking infrastructure	It is possible to build customized workflow tracking infrastructure to track the managed system’s workflows.	Tracking services are necessary for tracking down the intersecting workflows of the RCH components (archival, retrieval and presentation).
Extensible activities	Extensible activities are natively supported within the WF development framework.	The unique nature of RCH requires building customized workflow management modules for some of the complex system operations such as object mapping, workflow runtime service hosting, etc.

Programmability	WF is a development framework. Thus, it requires the implementation of applications and custom modules to host its services.	The WF supports a number of programming languages and implementation paradigms to host its services. WF workflow runtime services can be hosted in either web-based or standalone applications in a SO paradigm. They can also be implemented as distributed web services within a SOAP paradigm.
Workflow Designer	Workflow management applications can be created by using the visual design tools provided by the Visual Studio, .NET's main development environment.	The WF comes with a standard workflow designer within its development environment (the Visual Studio). This designer facilitates the process of designing and building workflow management implementations. It is also possible to complement the created designed with custom written lower level code for extra functionality.

In addition to the advantages listed above, one of the WF's strengths is its suitability for scenarios that involve the management of existing applications. Such scenarios usually include UI page controller, long running business logic, dynamically updateable process flow, web service composition, and abstraction of rules from business logic [143].

The WF based RCH WfMS needed a visual interface to allow its users to interact with its services. Therefore, it was an important issue to choose an implementation technology to host the actual WfMS. The most practical and straightforward option for prototyping to test workflow concepts was to create a Windows based WfMS Application, while bearing in mind that the actual final RCH system will be a web

application, requiring a final WfMS to be mapped to this web application. In addition, it is relatively easy to map from a Windows Application to a web application. Windows Applications are a part of the .NET family of products, which meant that compatibility issues are eliminated in the process of integrating with WF solutions. The created host is represented in the form of a standalone application, called the RCH Content Management System (RCMS), that integrated with the devised RCH WfMS, as will be further illustrated in Section 6.5.

6.3.2 RCH WfMS Components

The RCH WfMS comprised a number of components that collectively provided the workflow management functionality. The key elements of the RCH WfMS contained the **Workflow Runtime Services**, the **Workflow Execution Engine** and the **Administration and Management Tools**. The functionality and role of the utilized components are summarized as follows:

- **Workflow Runtime Services**

The RCH WfMS workflow runtime services provided a set of core workflow management functionalities that were responsible for launching and managing the different RCH components. The most important elements within the runtime services are the persistence and tracking services. The persistence services provided the necessary workflow maintenance services, including functionalities such as workflow serialization and restoration. The tracking services within the created runtime services provided a set of event-based tracking functionalities in relation to the running workflow instances (these are discussed further in Section 6.4.1).

- **Workflow Execution Engine**

The workflow execution engine included a number of code routines used to execute the different workflow instances according to the system events. The execution engine was integrated within the workflow runtime services as it interacts with the data and flow control events of the RCH components. The code constructs used to build the custom workflow execution engine are based

on specialized VB.NET classes that provided the necessary workflow execution functionalities, as will be further illustrated in Section 6.4.

- **Administration and Management Tools**

The Administration and Management Tools comprised a number of code modules and controls that complemented the functionality of the other WFMS components. The administrative tools comprised the RCMS, as will be illustrated in Section 6.5.

6.3.3 The RCH WfMS Design

A layered approach was adopted in the process of designing the WfMS for RCH. Three different workflow runtime services were created to fulfil some of the core workflow management functionalities: the archival, presentation and retrieval workflow runtime services. Each workflow runtime service was designed separately as a subsystem of the overall RCH WfMS. These separate workflow implementations were then integrated together within the workflow host that managed them according to the system events (the object retrieval operations for instance).

Each individual workflow runtime service performs a set of specialized tasks related to the area that it is handling. For example, the archival runtime services are involved with invoking and managing the system's archival components, including the object mapping processes (mapping the museums data as explained in Section 5.3.2) and their associated data flows. The same applies to the interrelated retrieval and presentation workflow runtime services that perform their own specialized tasks respectively.

The reason for breaking up the RCH WfMS into three subsystems (workflow runtime services) was the need for a flexible implementation model that can be easily manipulated and adapted while maintaining certain levels of flexibility, scalability, expandability and customizability. For example, it would be possible to host the three workflows for archival, retrieval and presentation on separate machines over the internet. By building three separate workflow runtime services based on functionality, it was a straightforward process to initiate the appropriate workflow instances from the user end without interfering with the other system's functionality. These separate workflow runtime services (can be considered as sub-engines in the context of the

overall RCH WfMS) were then hosted within the devised workflow host, as explained in Section 6.3.3.1.

The implemented workflow runtime services are logically separate in a paradigm similar to the concept of encapsulation used in Object Oriented Programming (OOP). According to Snyder [144] encapsulation is usually utilized to radically minimize dependencies between separately written code components by means of writing totally separate internal code interfaces. In fact, the concept of encapsulation is implemented throughout the RCH components, as highlighted in Chapters 4 and 5.

One of the advantages that workflow runtime encapsulation provides is the support of flexible workflow management in distributed and highly heterogeneous environments [145]. Such a support is achieved via the utilization of a hierarchical modular architectural design pattern that includes a number of collaborative sub-processes. Adding more complex workflow features and enchantments involves the operation of adding more encapsulated workflow runtime services. Therefore, the adopted model has high levels of expandability and scalability. Hence, regardless of the workflow complexity, an RCH WfMS prototype will still be able to effectively deal with it. The details of each of the implemented workflow runtime services are outlined in Sections 6.4.2, 6.4.3 and 6.4.4.

It is worth noting here that the concept of encapsulation is not only applied between the workflow runtime services, it is also applied in the relationship between the workflow host and the workflow runtime services. In this instance, the workflow host for RCH, which acts as a service consumer, does not know the internal processes and procedures of the runtime services, an archival service for instance, etc. Moreover, the runtime services work to provide the most cost effective process execution. Each workflow runtime service can be considered as an independent unit of code that directly interacts with the workflow host, which directly interacts with the system users. Each workflow runtime service performs a process execution based on the messages passed to it by the other system components.

The concept of encapsulation within RCH and its workflow management components is illustrated in Figure 6-3. There are three encapsulated main components that interact with each other to achieve the workflow-managed functionality of RCH. These

components are the RCH Components (archival, retrieval and presentation), the RCH WfMS, and the WfMS host (RCMS).

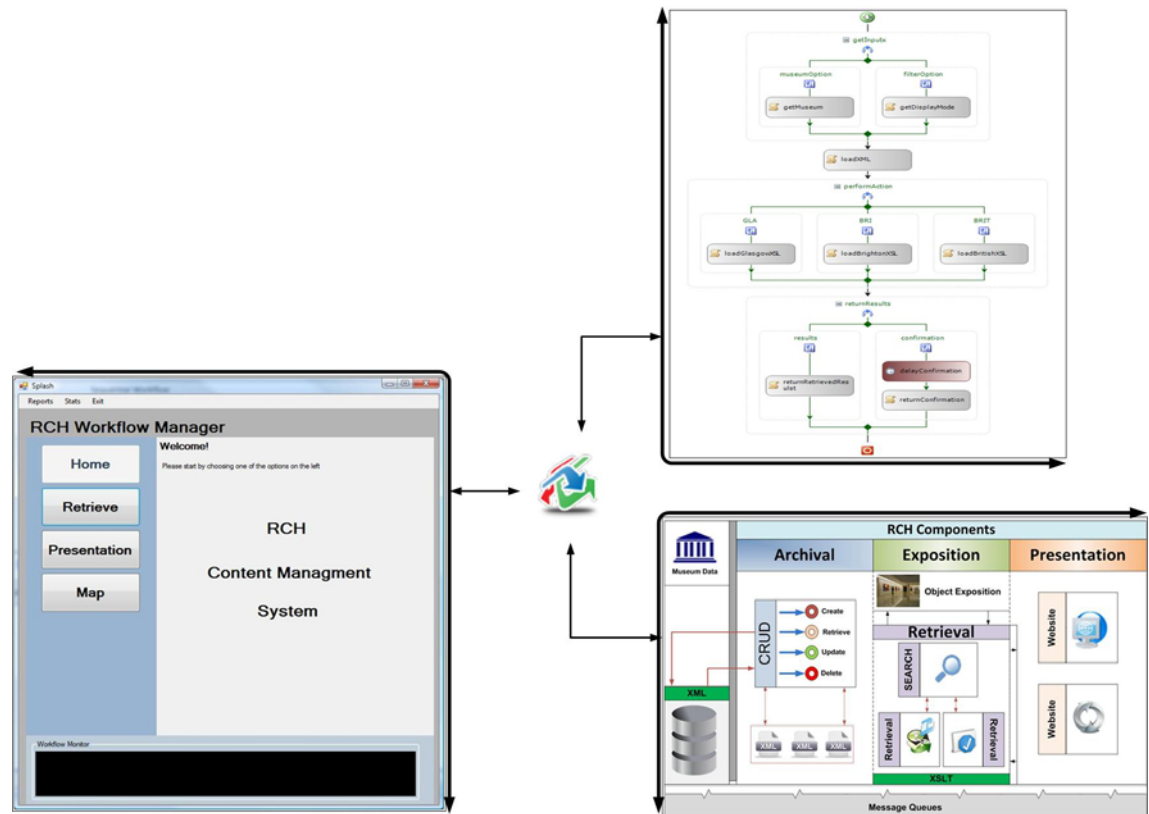


Figure 6-3 Interaction between the RCH Components, RCH WfMS and RCMS

6.3.3.1 RCH Workflow Hosting

A WF workflow is a runtime and not an application, as indicated by Allen [146]. The implication of this is that any workflow runtime built by using WF needs an appropriate workflow host to invoke and control it. This approach provides certain flexibility when implementing workflow solutions, as the underlying workflow can be hosted in an appropriate application or interface to present its services. Another advantage of this approach is highlighted by Allen [146], who indicates that a host can provide an extra functionality to the actual workflow runtime.

What is meant by “hosting the workflow system” is the process of utilizing an application to host the workflow management components and facilitate their interaction with the system users and other software components. In the case of RCH, this interaction is achieved through message and parameter passing between the created WfMS, the WfMS host, and the hosted RCH components. A host application can be a website or a standalone application, depending on the adopted implementation scenario:

in the context of this thesis, for convenience only, a Windows application was chosen to provide the host.

The hosting process aimed to provide the functionality necessary to invoke and run the designed workflow runtimes. Figure 6-4 illustrates the interaction between the WfMS, the WfMS host and the hosted RCH components. Effective message passing and collaboration between the workflow runtimes and the RCH workflow host, leads to the invocation and running of the intended workflow activities. Workflow activities are initiated via the messages that are passed from the workflow host leading to invoking the needed workflow runtime instances. The results of running the initiated workflow instances are then passed back to the host to provide some sort of feedback to the system users. This feedback may constitute confirming the running of a certain workflow instance or process termination for instance.

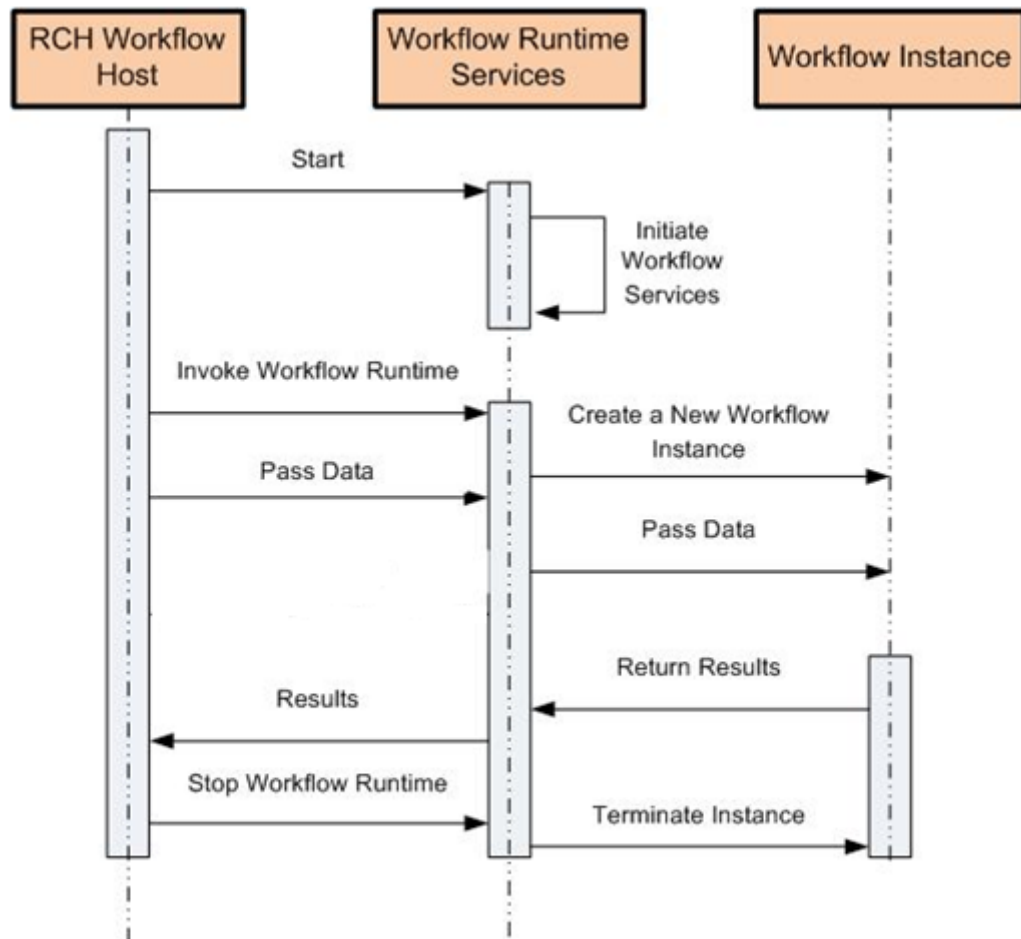


Figure 6-4 RCH Message Exchange Illustration

6.4 The Workflow System Implementation

The actual implementation of the RCH WfMS was based on utilizing the different code controls and constructs that WF provides. According to Pegasus [147], the main approach adopted in WF is the declarative creation of workflows within a visual design while using a number of standard code building blocks. These code building blocks provide comprehensive workflow functionality, such as the creation of workflow runtime, persistence, tracking and monitoring services. The WF controls that were utilized within the RCH WfMS prototype are summarized in

Table 6-2. The practical use of these controls is illustrated in Section 6.4.1

Table 6-2 The Utilized WF Code Constructs

Code Construct	Purpose
ifElse Activities	ifElse Activities are among the most common WF control-flow activity types [148]. They were used to handle the different rules and conditions that are associated with running the different workflow runtimes associated with the devised WfMS. Nested ifElse branches were also used to support different levels of code execution according the parameters passed to the WfMS.
Executable Code Activities	Executable Code Activities are considered to be among the most fundamental workflow management building blocks [149]. They were intensively utilized within the created WfMS to provide application-specific workflow management functionality. They provided most of the customized workflow management and control functionality including handling the data inputs and outputs as well as interacting with RCH functional components (archival, retrieval and presentation).

Fault Handlers	The WF Fault Handlers were used to ensure a smooth running of the workflow activities while being able to effectively handle any workflow exceptions without interrupting the running processes. They were used in the error prone areas of the system especially those that involve decision making, runtime service invocation, runtime service termination and message and parameter passing.
Listen Activities	The WF Listen Activities define a set of event-driven activities that typically wait for a specific workflow event to occur before taking the appropriate course of action [150]. These code constructs were used to exert some level of control over the operation of the underlying runtime services as well as the process execution activities that are promoted by the system components. For example, WF Listen Activities in RCH WfMS wait for a number of possible events to happen in the mapping process including file upload, source/destination specifications, etc.

6.4.1 Workflow Runtime Construction

The created workflow runtime services were built by the utilization of the WF's code constructs that are listed in Table 6-2. Each workflow runtime service was created separately and integrated within the workflow host application. What follows is a description of each of the devised workflow runtime services.

6.4.2 The Archival Components Workflow Runtime Services

The RCH archival functionality that was highlighted in Section 5.3 handled the different data-centric tasks in relation to the managed cultural heritage collections. Among the most important components of the RCH archival tools is the data mapping services that facilitated mapping the cultural objects data from one museum to another, as detailed in Section 5.3.2. The mapping component was built by the author of this thesis to act as a

test bed for the integration of WfMS within the RCH archival components. Such a complex component with its input and output data flows, needed an appropriate WfMS to host its services and govern their behaviour. In this context, all the procedural workflow activities were performed by a set of dedicated workflow runtime services, which are self-contained units of functionality within the devised WfMS.

A separate workflow runtime service was created to handle the sequential workflow requirements of the system's archival components. Such an approach was facilitated by the WF architecture that allows for the creation of specialized runtime services that can span a number of customized code modules [151]. For example, an instance of the created archival runtime service was created each time a user performed a mapping operation. This workflow runtime service instance's running and invocation is associated with messages from the WfMS host. The passed messages specify the intended operation and the data associated with it. This process leads the called runtime service to execute the right sequence of events within its sequential rule-based workflow model.

RCH is a system that has to handle simultaneous users at all times due to its distributed nature as a shared DHR. Therefore, the concept of simultaneous workflow runtime instance running proved to be a key feature in the archival components WfMS. Exploiting this feature meant that the archival components can be used by as many users as possible at any specific point of time. This is because any number of workflow runtime instances can be initiated and invoked to handle the different user sessions.

An example of the managed archival processes is the object data mapping process. This process starts with reading the user input, including the source and destination museums. A set of **ifElse** activities that execute the right code blocks depending on the user choice govern the mapping process. The internal workflow runtime processing follows the process of reading the user input where the right range of archival components are called. Successful workflow execution results in producing the correctly mapped data to be used by the system users. The produced mapped file along a confirmation message is returned to the end-user via the RCMS. On the other hand, faults and runtime errors are also handled by the customized WF fault handlers. The used fault handlers make sure that any running workflow instance continues to run until terminated by the system users or one of its components.

The workflow design of the mapping process that represents a good example of one of the archival components workflows is illustrated in Figure 6-5. The sequential workflow design that represents a typical mapping operation starts with reading the user input through an executable code activity named **readInput**. The **readInput** code activity reads the user request for a mapping operation (containing source museum, destination museum, and the details of the source file of the data to be mapped). The **inputConfirmed** code activity gets the user confirmation that the inputted parameters are correct to proceed with the mapping operation. A **responseDelay** activity is introduced after collecting the user input allowing for calling the corresponding runtime services to start the mapping process. The **mappingOptions** *ifElse* code activity branches into three executable code activities. Each of these code activities represents the mapping rules from one museum and performs the mapping operation to the target museum according to the user input. The actual mapping process is achieved via invoking an instance of the archival runtime service.

User interaction is achieved through the utilization of the workflow host. Message handling is done through customized code routines that are able to read and interpret the received messages. The main mechanism used here is based on the process of creating public workflow runtime properties that take the value of the parameters entered by the user. An example of such public properties is the public property used to determine the source museum as shown in the code snippet below:

```
Public Property sourceParameters() As String
    Set(ByVal value As String)
        sourceMuseum = value
    End Set
    Get
        Return sourceMuseum
    End Get
End Property
```

The public workflow runtime property, which is called **sourceParameters**, takes a single value, the **sourceMuseum**, and passes it to be used within the workflow management operations when performing data mapping operations. Public property

parameter passing is a two-way operation, as parameters are passed from the workflow host to the workflow management components and vice versa.

Another example of the process of parameter passing is highlighted in the code snippet below:

```
WorkflowInstance=myWfRuntime.CreateWorkflow(GetType(RCHAR  
chivalWorkflowWorkflowApplication1.Workflow1),  
WfParameters)
```

WfParameters represent the actual parameters that get passed to the initiated workflow runtime service to be used in association with the executed processes. The actual parameters are the result of the operation of the host application code classes and methods that act on passing the correct parameters to the WfMS host. Furthermore, the same mechanism is followed in all the other host/runtime interactions, regardless of the type of workflow runtime service instance being initiated. The data mapping scenario is illustrated in the testing scenario in Section 6.6.1.

The code snippet below illustrates the actual process of running the created workflow runtime services via the `Instance.Start()` method. The workflow runtime indicators are collected in a variable called `wfIndicators`, which is used within the workflow tracking monitor (see Figure 6-17) to output different workflow indicators to the end-users.

```
WorkflowInstance Instance.Start()  
  
Dim wfIndicators = "Workflow Instance ID:" &  
WorkflowInstance.InstanceId.ToString & vbCrLf & "Runtime  
Name:" & WorkflowInstance.WorkflowRuntime.Name.ToString &  
vbCrLf & "Runtime Status: Running"  
  
Return wfIndicators
```

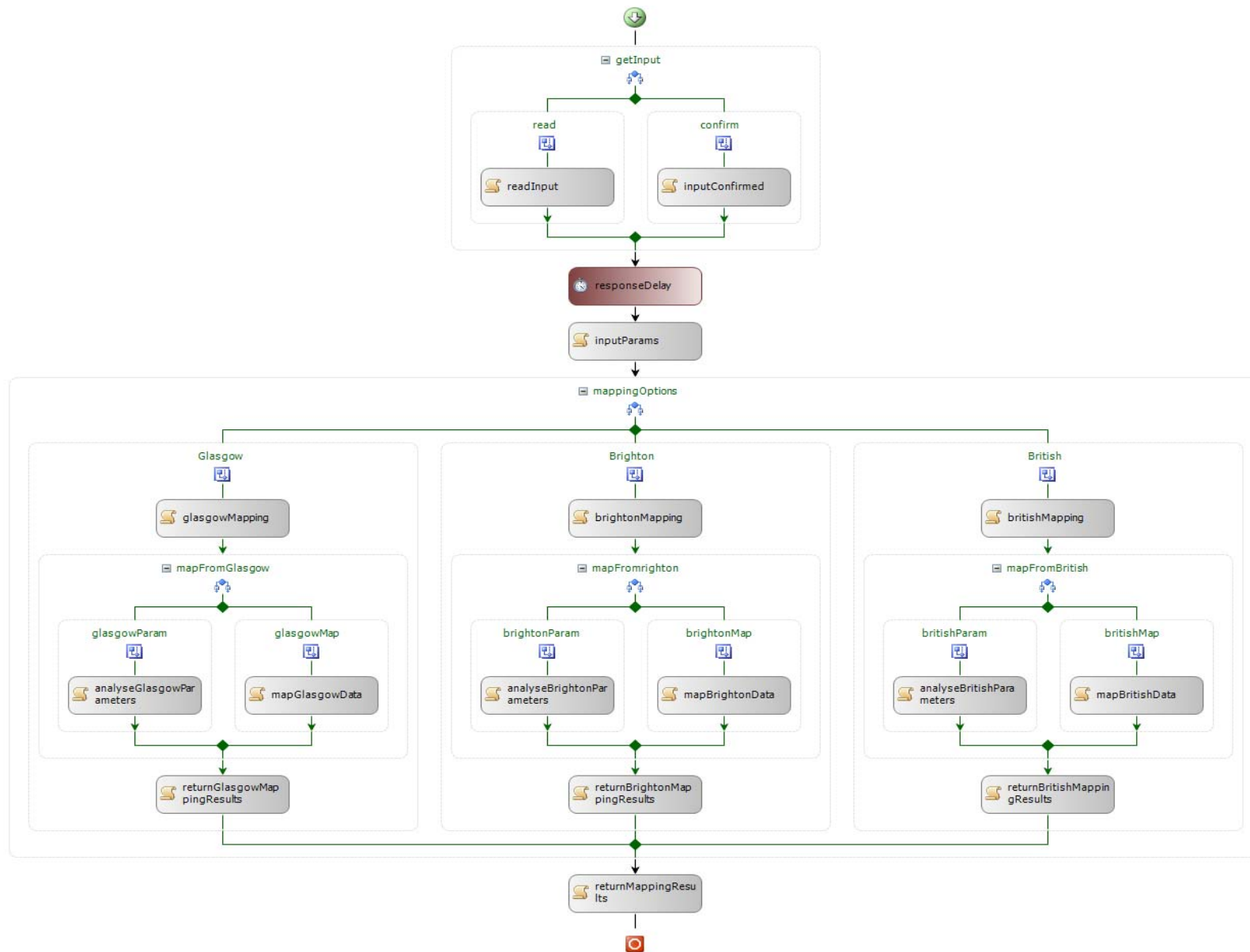


Figure 6-5 Archival Mapping Workflow Runtime

6.4.3 The Retrieval Components Workflow Runtime Services

The Retrieval components workflow runtime services formed a part of the overall RCH WfMS. This workflow runtime implementation provided the necessary functionality to handle the user's search queries to retrieve the required cultural objects. The retrieval results are then displayed to the end-user by using the RCH presentation components, as highlighted in Section 6.4.4. The main challenge here was developing an appropriate approach to handle the complex workflows of the retrieval process.

The retrieval components workflow proved to be the most complex among the implemented workflow runtime services due to the complexity of the rules that are associated with its sequential design. Figure 6-6 shows the latest search interface as a web page interface: the actual workflow prototype uses a Windows application to test the workflow concepts.

This complexity of the retrieval components workflow runtime services stemmed from the need to be able to handle all the combinations of the user queries while taking care of the search options and filters (museum, category, region, etc.) that might be applied. In the context of this thesis it should be noted that the goal was not to design an optimal search engine with appropriate interface, but to produce prototype search functionality as a part of the retrieval component to be able to validate a WfMS for the object retrieval operations within RCH.

Figure 6-6 RCH Search Interface

Figure 6-7 illustrates the design of the retrieval components workflow runtime services. In a similar fashion to the archival components workflow, the whole workflow process starts with a specific user action. User actions are represented in the search operations that come through from the RCH UI (Figure 6-6). Each search operation is associated with a number of parameters, including the keyword search (read through the **readInput** code activity) and the associated filters (such as object category, theme, cultural group, etc.) which are read through the **readFilters** code activity. These parameters are collectively handled by the **getInput** Listen Activity which passes the entered parameters to the **chooseMuseum** code activity. This code activity precedes an **ifElse** workflow activity that channels the search operations to the appropriate code activities that query the collections of the searched museums. After the completion of the retrieval process, the results alongside the appropriate workflow termination messages are sent to the workflow host via another **listenActivity** called '**returnResults**'. This listenActivity has the task of sending the retrieved results to the host in conjunction with the associated workflow indicators, which are controlled by a

delay Activity (**delayConfirmation**) so that they are delivered as the last bit of information to the end-user.

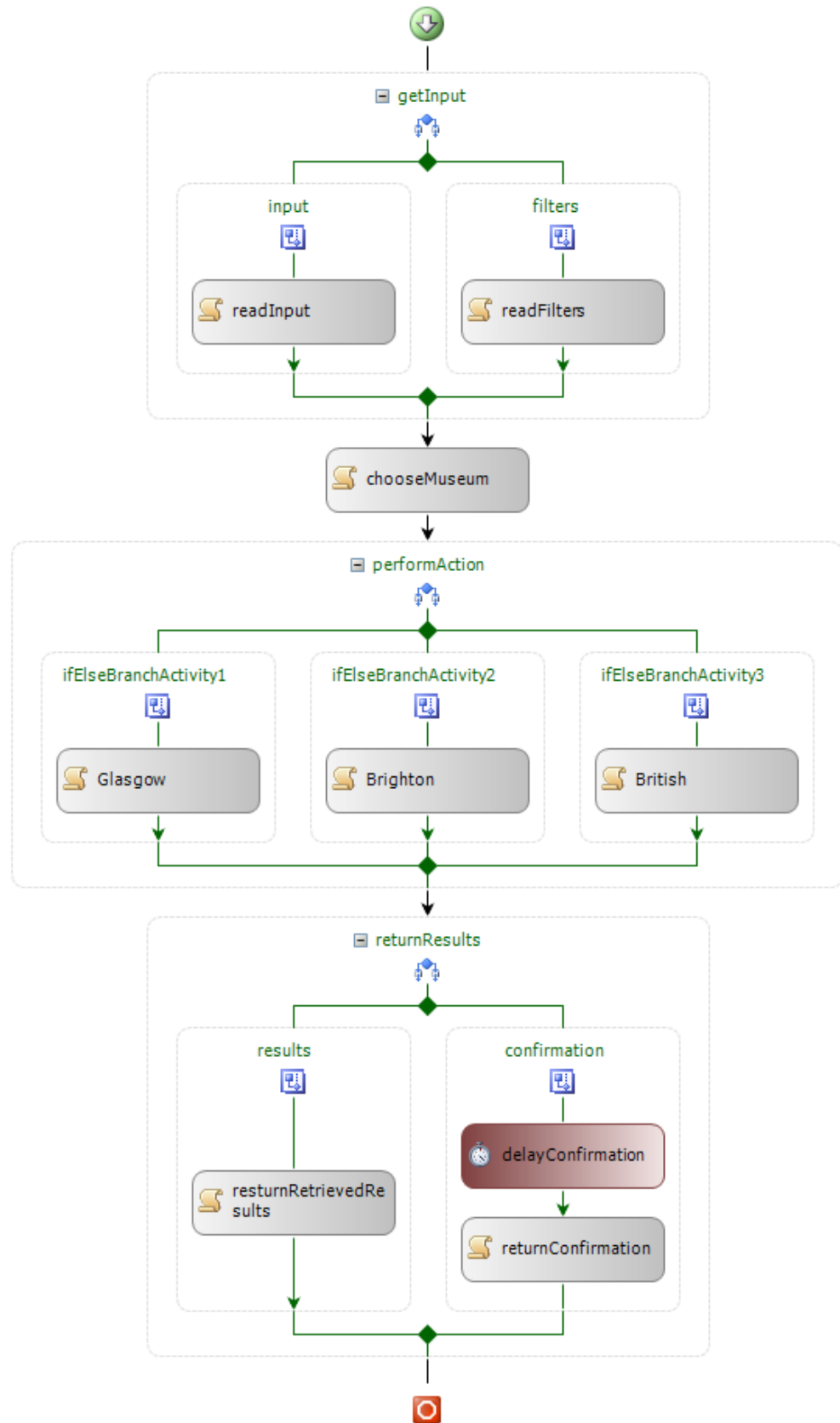


Figure 6-7 Retrieval Workflow Runtime

6.4.4 The Presentation Components Workflow Runtime Services

Although it is not a common approach to manage the presentation aspects of an application by using a WfMS, there exist some practical implementations that proved to be viable and feasible in this respect [152]. For example, this concept was explored by Chao *et al* [152] where a workflow management infrastructure was utilized to produce a workflow-based Content Management System (CMS). This system was used to manage the different operational aspects of a Learning Management System (LMS), including its presentation layer's services and activities.

Similar to the archival and retrieval components, a separate workflow runtime service was created to handle the presentation aspects of the RCH system. The main concept here was to handle all the object presentation tasks through a customized workflow runtime service that interacts with the system's workflow host. The devised presentation workflow runtime services provided a comprehensive set of presentation-oriented services as summarized in Table 6-3.

Table 6-3 The Presentation Component's workflow-managed Services

Service	Description
Object Display Services	These services involve displaying the participating museums' data in a gallery view where all objects are displayed and grouped according to the user preferences. These services also respond to the users' requests to display the museums' objects while adhering to a predefined set of XSLT based display templates. An example of object display within RCMS is illustrated in Figure 6-13.
Customised View Services	These services provide the users with the facility to choose a number of custom views to browse the stored cultural heritage objects. This process is based on loading a certain XSLT file that presents the retrieved objects in the required mode (for example, list view, gallery view, etc.).

The Filtered Display Services	These services allow the end-users to view a limited subset of the stored cultural heritage objects as requested. For example, the presentation services can confine the displayed objects to only those objects that are related to a specific museum. This process is similar to object retrieval but it relies entirely on a pre-defined set of filters (museum, tribe, etc.).
-------------------------------	---

As the RCH presentation components are dependent on XSLT, as highlighted in Chapter 5, their workflow runtime services are involved with manipulating the different XSLT files that are used to present RCH's objects. As shown in Figure 6-9, the presentation component's workflow starts with a user's request for object presentation. Such a request usually comes associated with two parameters. The first parameter is the museum/s from which the data is going to be fetched and the other is the display mode that the user wants to view. The latter option is concerned with the way that the objects are going to be displayed. There are a number of supported XSLT-governed display modes in RCH including gallery view, list view and summary view. The user's preferences are interpreted resulting in the workflow runtime requesting the corresponding XSLT file to be loaded to the workflow host's UI. Consequently, the workflow host displays the returned XSLT-generated XHTML in an embedded browser within the RCMS as illustrated in Figures Figure 6-13 Figure 6-14.

The range of the displayed objects can also be controlled by dynamically switching the loaded XSLT. For instance, if a user chooses to view just a subset of the stored objects such as the objects of a specific museum, the RCH WfMS manages the process of loading the appropriate XSLT file. Figure 6-8 illustrates an example presentation mode which is the gallery view (this time within the actual RCH website). In this mode, the displayed objects are presented in separate galleries belonging to each of the participating museums.

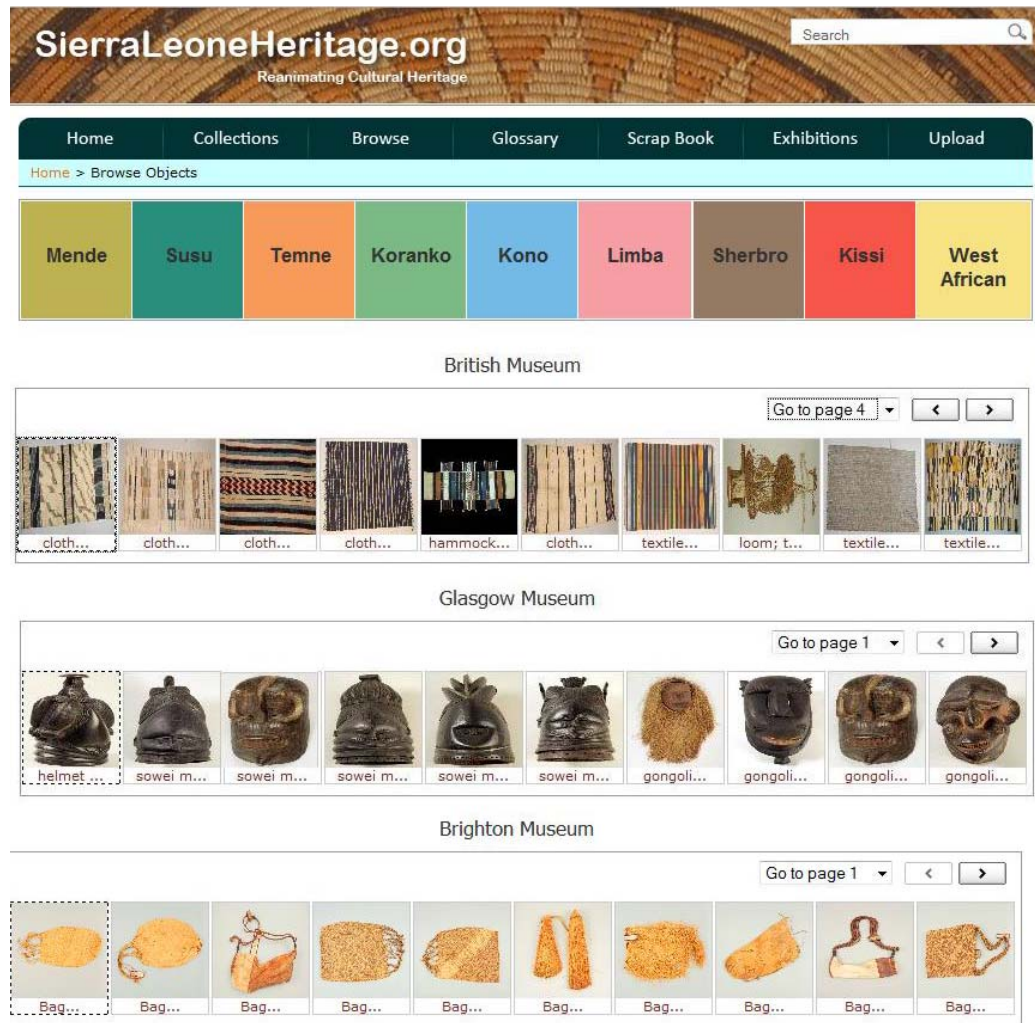


Figure 6-8 Gallery View Presentation Example

Again, a combination of workflow code constructs was utilized to allow for the efficient operation of the object presentation process, as illustrated in Figure 6-9. The **getMuseum** code activity reads the user's inputs that specify the museum from which the data is to be presented. The **getDisplayMode** activity reads the display mode specified by the user (gallery view, list view, etc.). The **loadXML** code activity is used to load the XML file from which the objects' data is to be fetched. The data is then presented by using one of the pre-built XSLT files (handled through **loadGlasgowXSL**, **loadBrightonXSL** and **loadBritishXSL** code activities). The **returnResults** **ifElseActivity** activity is used to display the rendered XHTML (produced by the loaded XSLT file).

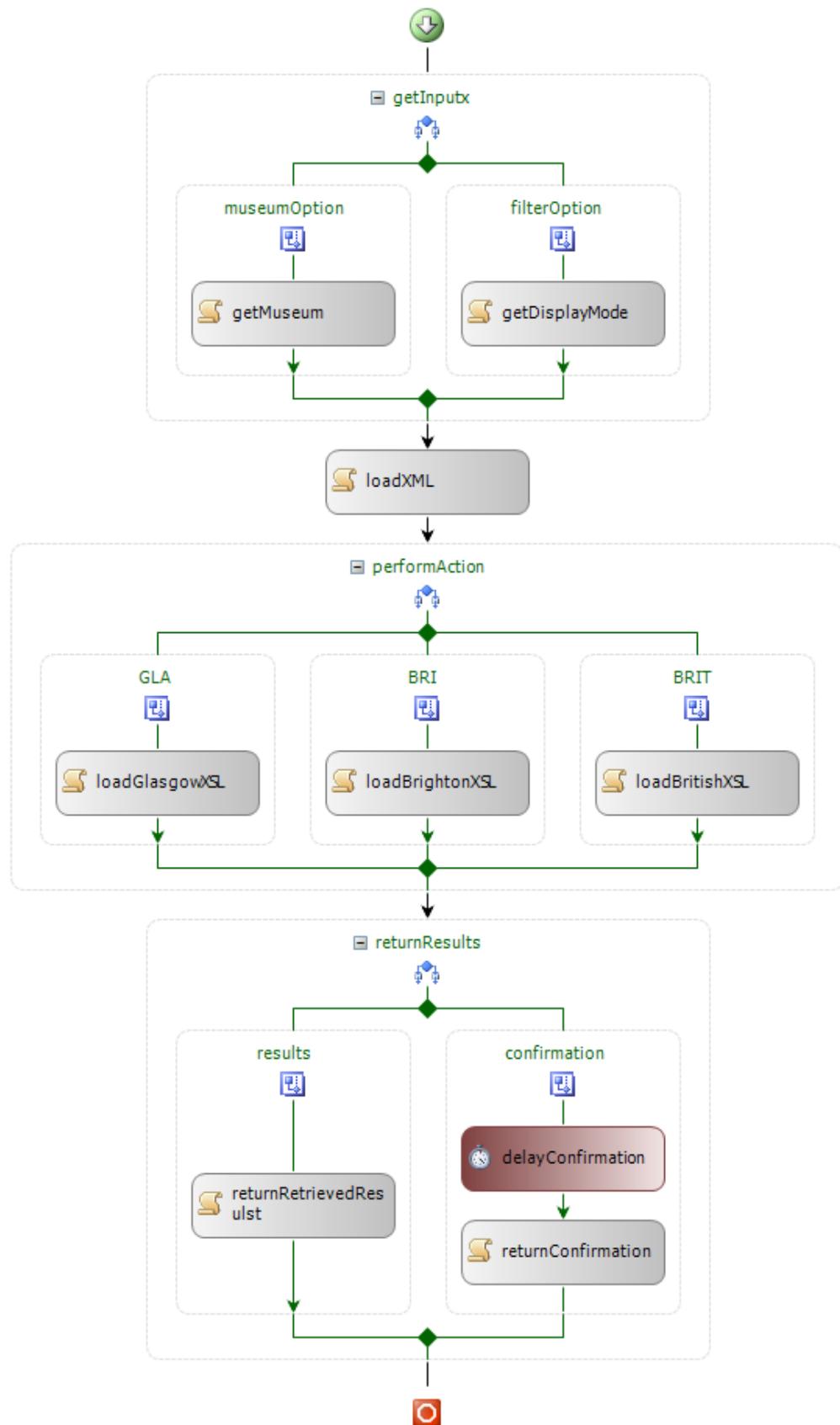


Figure 6-9 Presentation Workflow Runtime

6.5 RCH Content Management System (RCMS)

The created workflow runtime services were collectively hosted in a workflow host application. The RCMS application provided a UI that interacts with the system users while prompting and invoking the appropriate workflow runtime service instances. The chosen technological medium to implement the host application was the .NET Framework and the associated VB.NET programming language. These technologies were used to build a standalone Windows-based host application. The same can also be achieved through a web-based host, but a Windows application is used here to complement the functionality of the already existing RCH website.

Choosing a Windows application to host the custom-built workflow runtime services was justified for a number of practical reasons. Choosing a .NET implementation technology for the workflow host meant that the created application would not have compatibility or interoperability problems while interfacing with the created workflow runtime services. This is based on the fact that compatibility and interoperability between Windows applications and WF-implemented WfMSs is fully supported by the .NET Framework. Additionally, the .NET Framework allows for hosting workflow runtimes in any Windows implementations including Windows forms, regardless of the implementation language in use [153]. Therefore the created Windows application addressed the UI needs of the RCH WfMS. Such a UI component can be further customized or replaced when needed while preserving the core workflow management infrastructure.

6.5.1 RCMS Illustration

The RCMS comprised a user-friendly Windows application that included a number of Windows Forms that provided a range of managed RCH services. The different controls and options that the RCMS's UI provided facilitated the process of responding to user actions (requests for data retrieval, object mapping, presentation, etc.). The performed tasks were achieved while passing the appropriate parameters associated with each operation to be handled by the backend workflow management runtime services. The RCMS included three main interactive forms (Windows Forms): the **Archival Services Management Interface**, the **Object Retrieval Management Interface** and the **Object Presentation Management Interface**.

Figure 6-10 illustrates the home screen of the RCMS where the user can choose from one of three system management choices: Retrieval, Presentation and Mapping. Choosing each option results in displaying its relevant services and user controls as detailed below. It is important to understand that RCMS is not about building a content management system *per se*. RCMS from the UI perspective is merely the manifestation of the workflow system, which is the control flow code and rules implied in Figure 6-5, Figure 6-7 and Figure 6-9. In Figure 6-10, we can see that the grey background represents the RCH Workflow Manager; this is the host environment, which can return reports, stats, etc., in the Workflow Monitor (the black window). The RCMS is the central part which is a really simple interface, but it illustrates the concept of how WfMS can be built into a CMS.

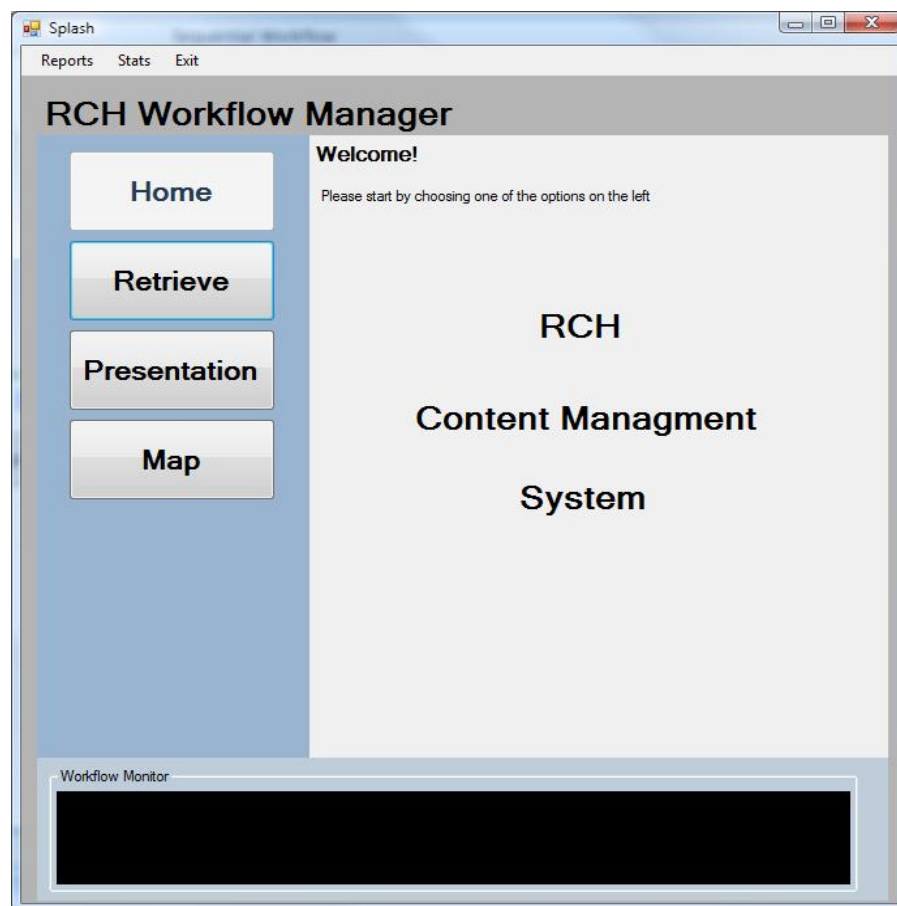


Figure 6-10 RCH Content Management System

the actual UI elements of the RCMS are built by utilizing the .NET's Visual Studio controls that were used to build the interactive elements of the RCMS UI. The used controls included buttons, text fields, panels, etc. The code snippet below illustrates the actual controls used to build the home screen shown in Figure 6-10. These controls include panels, group boxes, buttons, labels, etc.

```

Private Sub InitializeComponent()
    Me.Panel2 = New System.Windows.Forms.Panel
    Me.GroupBox1 = New System.Windows.Forms.GroupBox
    Me.Panel3 = New System.Windows.Forms.Panel
    Me.Label3 = New System.Windows.Forms.Label
    Me.ReportsToolStripMenuItem = New
System.Windows.Forms.ToolStripItem
    Me.Panel1 = New System.Windows.Forms.Panel
    Me.Button1 = New System.Windows.Forms.Button
    Me.Button2 = New System.Windows.Forms.Button
    Me.Button3 = New System.Windows.Forms.Button
    Me.StatsToolStripMenuItem = New
System.Windows.Forms.ToolStripItem
    Me.Label1 = New System.Windows.Forms.Label
    Me.ExitToolStripMenuItem = New
System.Windows.Forms.ToolStripItem
    Me.mpanel = New System.Windows.Forms.Panel
    Me.Label2 = New System.Windows.Forms.Label
    Me.MenuStrip1 = New
System.Windows.Forms.MenuStrip
    Me.Label4 = New System.Windows.Forms.Label
    Me.Button4 = New System.Windows.Forms.Button
    Me.Label5 = New System.Windows.Forms.Label
    Me.Panel2.SuspendLayout()
    .
    .
End Sub

```

Figure 6-11 illustrates the **Archival Management** interface, accessed through the Map link (the term map was chosen because in this prototype, which is testing the validity of WfMS, the main archival functionality chosen was the mapping of museum metadata to RCH metadata and vice versa). The system users are provided with the option of mapping the data of one museum to another. The UI deals with the users' preferred file location preferences, and performs the mapping operation while outputting valid mapping results in the desired format and location. For testing purposes, the file location can be either a local file directory or a web server. The latter scenario allows for the dynamic update of the RCH website frontend every time a mapping operation happens. For example, when a museum submits new objects to the shared heritage resource represented by RCH, they get reflected immediately in the relevant sections of the website, leading to the continuous availability of the most up-to-date data objects.

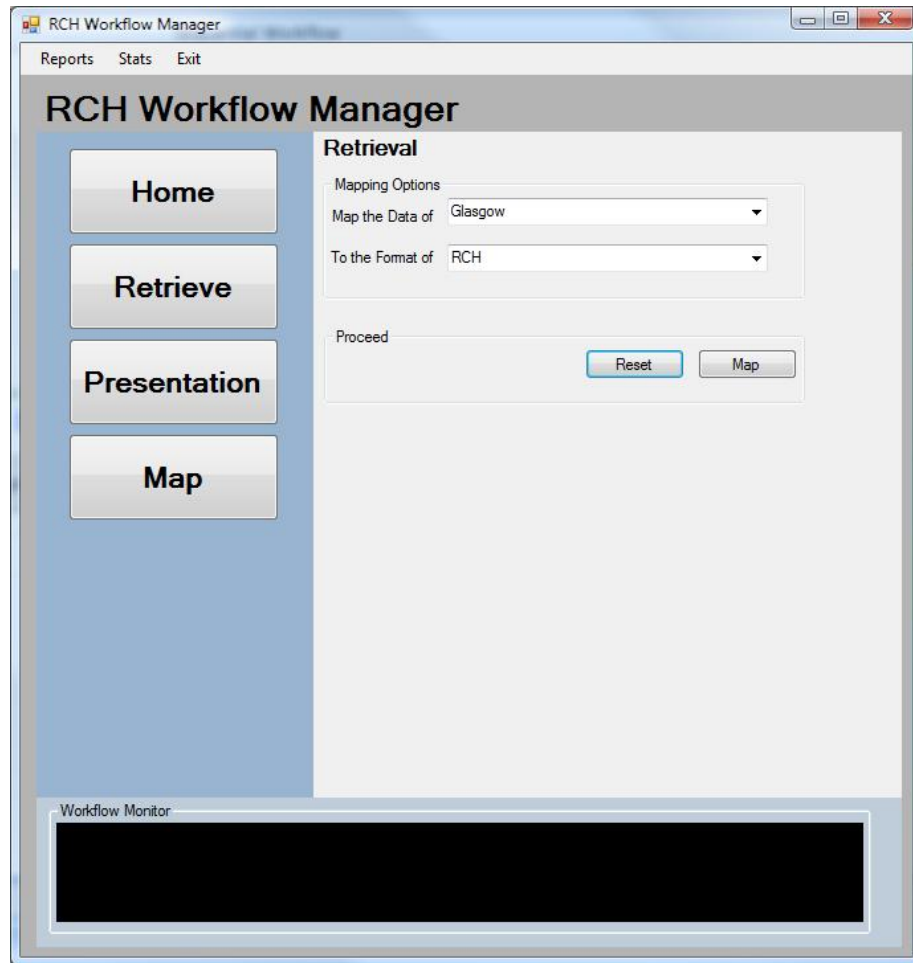


Figure 6-11 The Archival Management Interface

The **Object Retrieval** management interface is illustrated in Figure 6-12. It shows that users are provided with a set of options for object retrieval facilitated and managed by the retrieval workflow runtime services. A user can carry out a keyword search or associate the search process with a number of search filters to narrow down the range of returned objects. The provided filters include Museum, Cultural Group and Object Category. A combination of search keywords and search filters can also be used to obtain different perspectives about the objects that are being searched. Figure 5-20 above illustrates the latest web version of the RCH search interface, whereas Figure 6-12 illustrates a limited version for WfMS testing purposes.

The actual search operations are carried out by querying the backend XML files with the invoked retrieval components workflow runtime services, as was detailed in Section 6.4.3. In this case the ‘runtime service’ is, in effect, executing an XSLT search file in the Glasgow, British, and Brighton code blocks in Figure 6-7.



Figure 6-12 The Object Retrieval Management Interface

Object presentation involves the dynamic loading of an appropriate XSLT file to the embedded browser based on the user actions and the associated parameters. Figure 6-13 illustrates the **Object Presentation** management interface where the system users can carry out different presentation-oriented operations. The snapshot in Figure 6-13 shows that clicking on any museum's option results in displaying its objects in an embedded browser within the used form. Users can choose from a variety of display modes to view the managed objects.

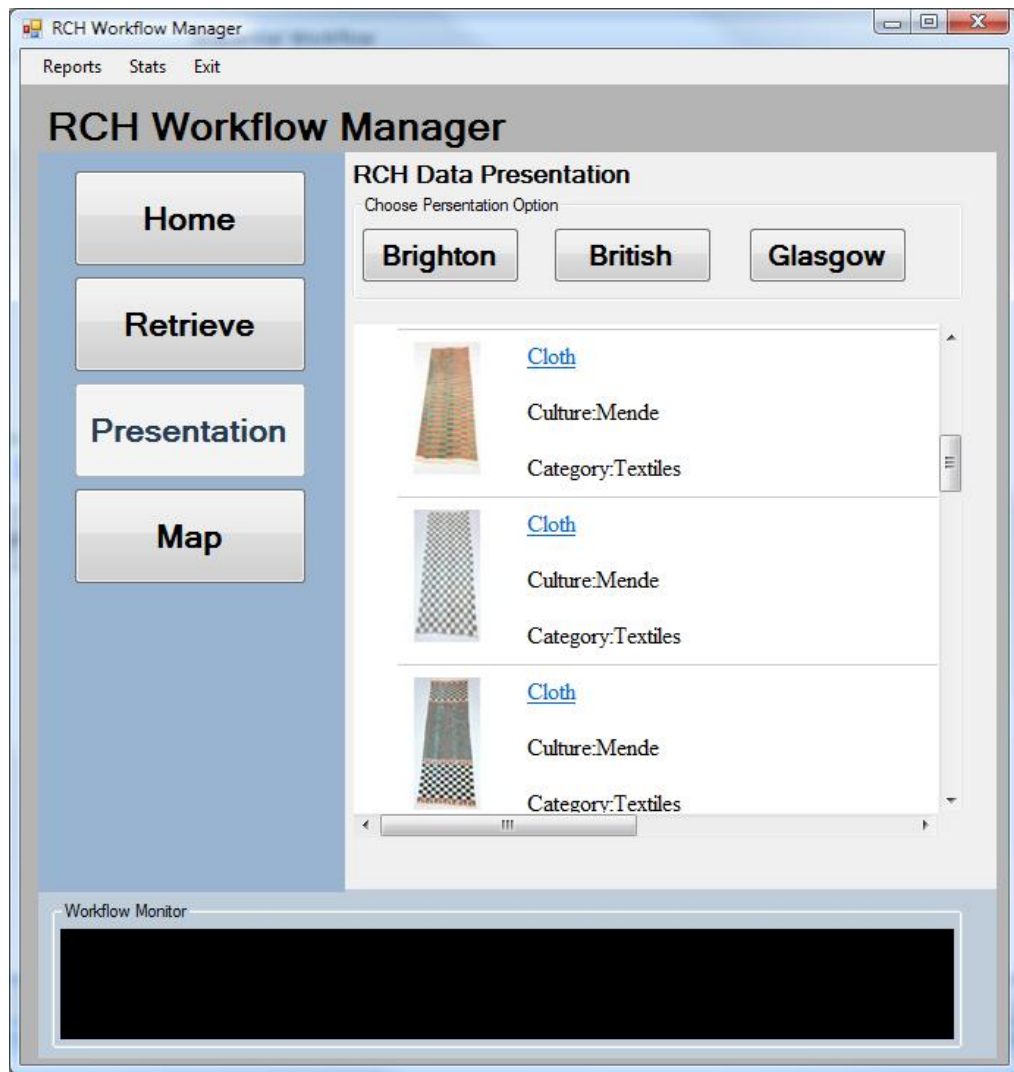


Figure 6-13 The Object Presentation Management Interface

The object presentation management tools support a number of presentation modes, as detailed in Section 6.4.4. These display modes include the detailed object display mode as illustrated in Figure 6-14.

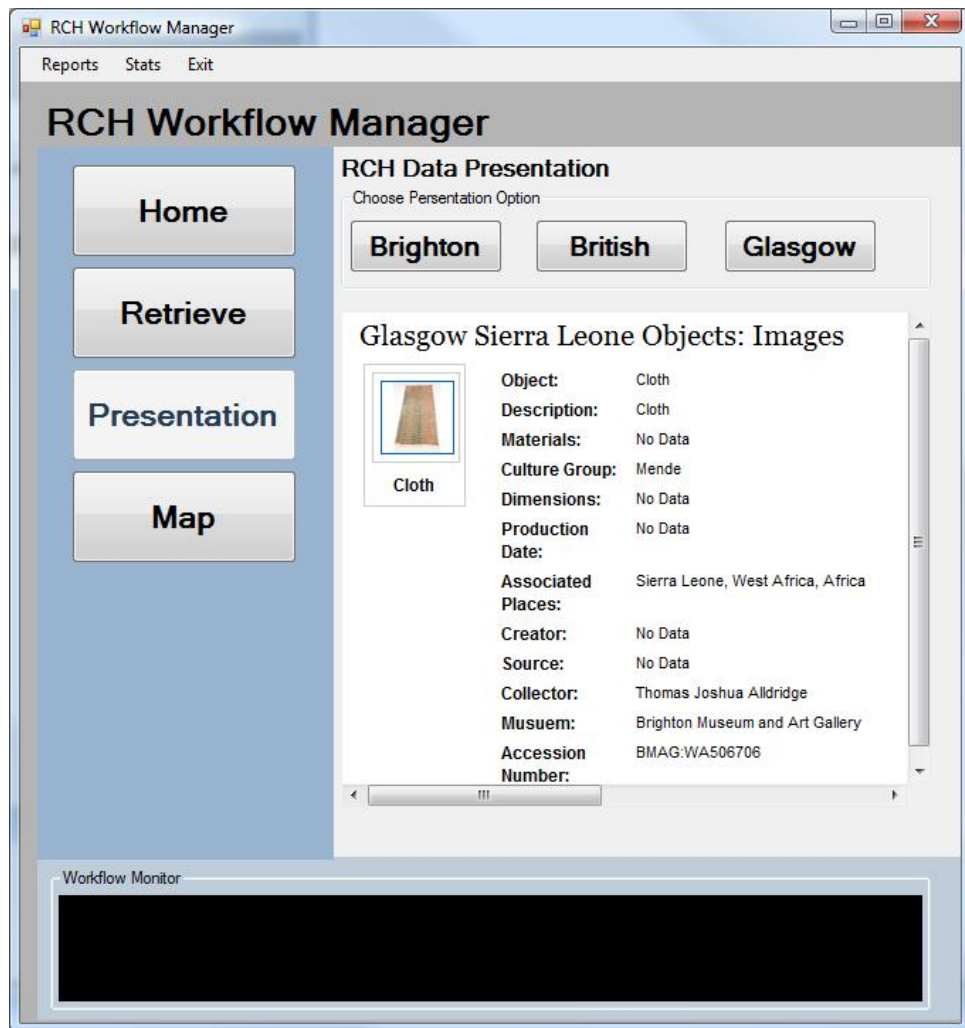


Figure 6-14 The Detailed Object Display Mode

In reality, this WfMS system would be hidden behind an admin interface, perhaps as a web page application, and the normal result would be something like that illustrated in Figure 6-15.

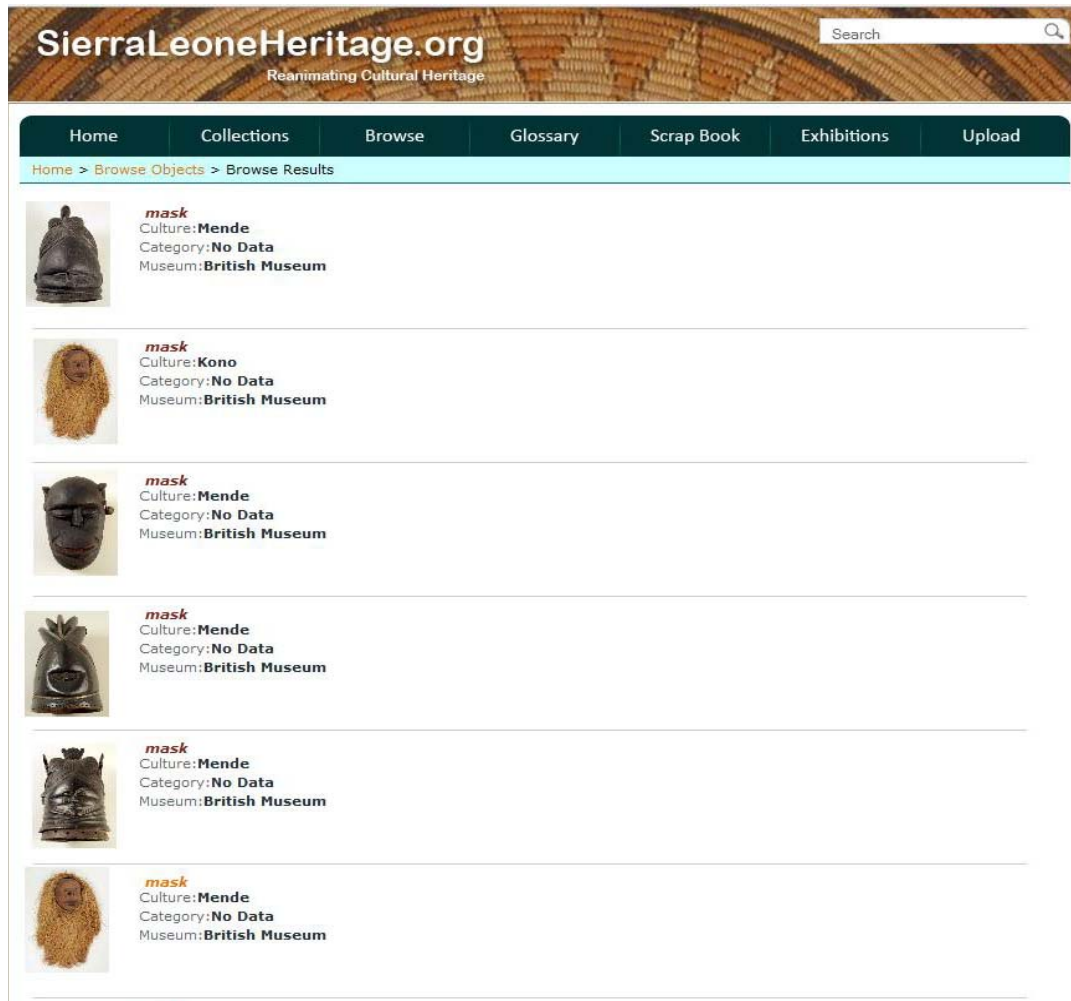


Figure 6-15 The Object Presentation Frontend

6.6 Workflow Running and Testing

The process of testing the implemented WfMS involved running and evaluating a number of test scenarios that spanned the three core workflow runtime services (archival, retrieval and presentation). Process execution was monitored through the different runtime indicators that associated the operation of the tested workflow runtime services.

Each test scenario was associated with a set of messages as well as data and control flows to be exchanged between the system components. An effective technique to test the workflow management operation was the WF standard workflow runtime properties. These properties determine the workflow status at any given point during the system running. Such properties include the **Runtime Name** and **ID** that represent the unique identifiers of each workflow runtime service. Another example property that was also used in conjunction with the RCH WfMS testing is the workflow **isRunning** Boolean property, which indicates whether a workflow is being run or not.

6.6.1 Example Testing Scenario: the Object Mapping Process

The object mapping process is a representative scenario of one of the main system services that is managed by the devised WfMS. The other system operations share the same basic principles that are associated with the RCH process invocation and management. Figure 6-16 illustrates the RCH Mapping Manager, which is a part of the RCMS. A successful mapping process was associated with outputting a message to the end-user as a feedback that indicates the conclusion of the mapping process. It can also be seen that the same screen comprises what is called the **RCH Workflow Monitor** (the black panel down the bottom) which displays a number of workflow monitoring and tracking indicators. This monitor provides the system users with a real-time runtime monitoring of the different workflow activities taking place during process execution.

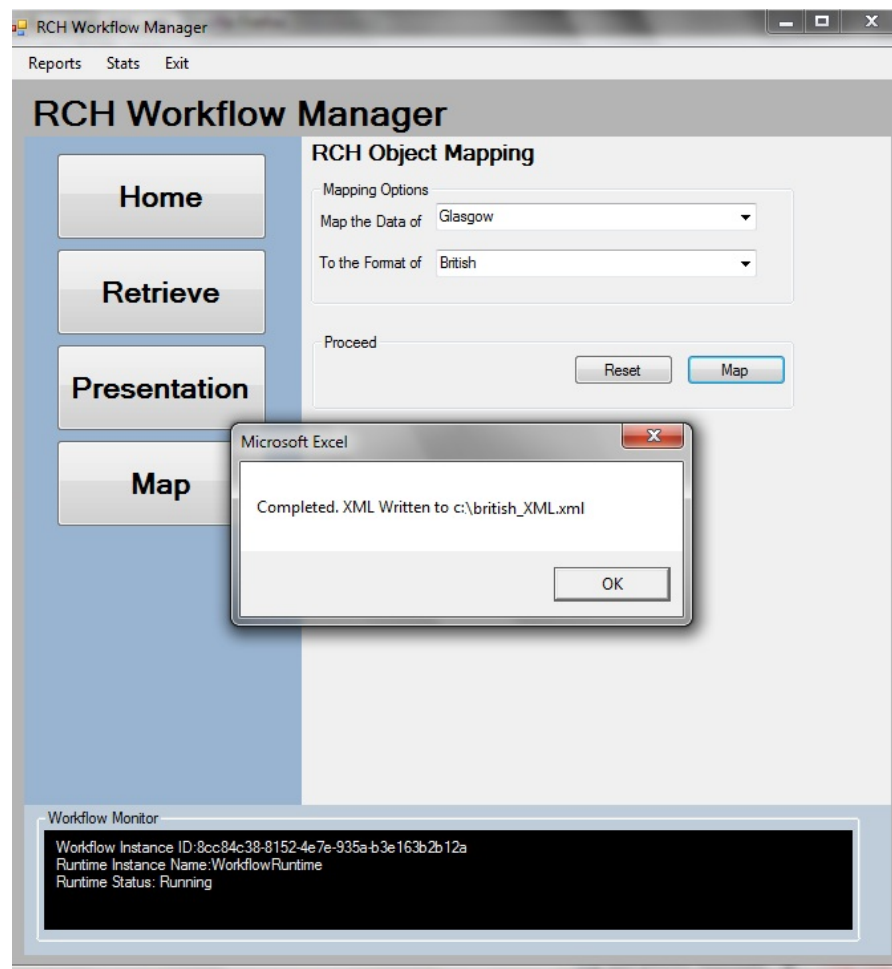


Figure 6-16 Object Mapping Confirmation

Figure 6-17 illustrates the RCH Workflow Monitor where a sample of three workflow runtime properties is displayed. These properties are the workflow **Runtime ID**, the

Workflow Runtime Name and the **Workflow Runtime Current Status** (i.e. running/not running).

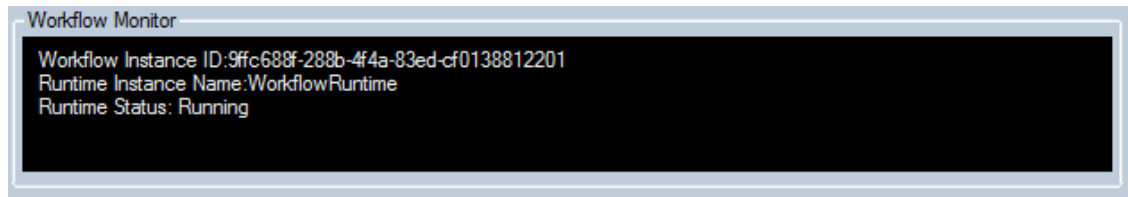


Figure 6-17 The RCH Workflow Monitor

The code snippet below illustrates a typical workflow runtime handler used within the devised WfMS. A standard WF handler is used here to handle workflow completion (`workflowRuntime.WorkflowCompleted`). Workflow runtime termination is handled through the (`workflowRuntime.WorkflowTerminated`) handler.

```
Shared Sub Main()
    Using workflowRuntime As New WorkflowRuntime()
        AddHandler workflowRuntime.WorkflowCompleted,
        AddressOf OnWorkflowCompleted
        AddHandler
        workflowRuntime.WorkflowTerminated,
        AddressOf OnWorkflowTerminated
        Dim workflowInstance As WorkflowInstance
        workflowInstance =
        workflowRuntime.CreateWorkflow_
        (GetType(mapping_workflow))
        workflowInstance.Start()
        WaitHandle.WaitOne()
    End Using
End Sub
```

6.6.2 Message Passing Verification

Another important aspect of testing the devised WfMS was examining the message exchange between the system components, namely the workflow host, the WfMS components, and the RCH functional components. Message passing was tested at two levels:

- message passing from the host application to the workflow engine and vice versa;
- message passing from the workflow engine to the RCH components and vice versa.

Figure 6-18 and Figure 6-19 show a tracked code snippet from the workflow host. These code snippets illustrate the process of passing two parameters to workflow

components (runtime services) during the object mapping process. The first parameter is the source museum, which is in this case is the Glasgow Museum. The second parameter is the destination museum, which is the British Museum.

```
Private Sub Button5_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button5.Click
    Dim mymap As New Map
    workflowMonitor.Text = mymap.map(sourceMuseum, destinationMuseum)
End Sub
```

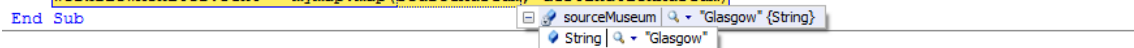


Figure 6-18 Passing the Source Museum Data from the Workflow Host

```
Private Sub Button5_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button5.Click
    Dim mymap As New Map
    workflowMonitor.Text = mymap.map(sourceMuseum, destinationMuseum)
End Sub
```




Figure 6-19 Passing the Destination Museum Data from the Workflow Host

The invocation of the required workflow runtime services is associated with the messages passed from the RCMS as shown above. During the mapping process, the mapping runtime services are invoked by the WfMS while passing a string containing the source and destination museum received through the RCMS. This process is illustrated in Figure 6-20, where two parameters are associated with the process of invoking the object mapping workflow runtime services. These parameters are the source and destination museums as highlighted in Figure 6-18 and Figure 6-19 above.

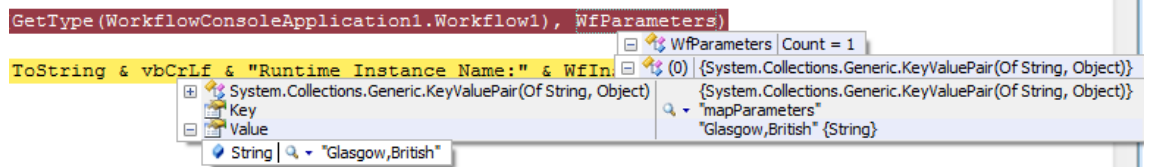


Figure 6-20 Workflow Runtime Service Invocation

As shown in Figure 6-21, the inputted parameters are then passed to the actual invoked runtime service instance to be used in its operations.

```
Public Property mapParameters() As String
    Set(ByVal value As String)
        mappingOptions = value
    End Set
    Get
        Return mappingOptions
    End Get
End Property
```

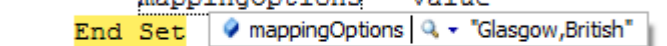


Figure 6-21 Using the Parameters within the Called Runtime Service

6.7 Results Analysis

The implemented system involved the creation of a custom-made WfMS that integrated with the RCH components. The devised RCH WfMS provided workflow management and monitoring components that were hosted in a Windows application that acted as a workflow client application. The process of integrating and running the WfMS for RCH was hallmarked with intensive message exchange and communication between the system components. This approach suited the highly encapsulated loosely-coupled code constructs that constituted the RCH components. At the level of the baseline system services, it was possible to track and monitor the different system workflow activities by using various workflow runtime properties supported by the WF tools. Such properties (Workflow ID, Status, etc.) allowed for testing the highly hierarchical RCH WfMS at different execution points.

It is worth noting here that a particular emphasis was placed on the performance and flexibility aspects of the system as “there is always a conflict between flexibility and performance, although they are two of the first important goals of WfMS”, Zhan and Xiaohui [154]. Hence, in order to improve flexibility it was very important to adopt a totally modular encapsulated approach that is independent and can seamlessly integrate with RCH components. This approach was evident in the actual RCH WfMS implementation, which comprised three separate encapsulated workflow runtime services (archival, retrieval and presentation). Flexibility was achieved here by means of facilitating the incorporation of any further workflow management functionalities by simply adding new independent workflow runtime services.

It should be noted that the incorporation of a WfMS was a valuable addition to the RCH system. First and foremost, it provided total separation between the system’s functional components and its visual elements including the content management tools. This supported the SO nature of RCH and offered a number of advantages, including better management capabilities of RCH’S components. The adopted approach represents a clear exploitation of the idea of having a workflow management software middle layer with operational benefits for the existing code infrastructure.

RCH WfMS increased the RCH’s capacity to serve its users while being able to handle the complex data being flown through its components; a characteristic that increased RCH’s efficiency and scalability especially within its distributed environment. Figure

6-22 illustrates the interaction between RCH and the components that it manages at an abstract level. It can be seen that the RCH WfMS sits as a middle workflow management layer between RCMS and the RCH repository itself.

RCH WfMS provides better scalability possibilities to RCH due to the fact that it acts as a middle management layer. This workflow management middle layer is able to accept new user sessions and extra components without changing the actual structure of the RCMS or RCH itself. As the operation of the adopted model is based on workflow runtime instances and message passing, scalability is supported by default. This is an important enhancement to RCH due to its distributed nature and varied user groups.

On the other hand, expandability is also made easier by the integrated workflow management layer. Providing total isolation between the functional modules of RCH and its UI elements meant that enhancements can be introduced easily without disturbing the operation of the unchanged components. In this context, the RCH WfMS accommodates new enchantments and components by managing their message passing and creating the appropriate workflow runtime services to manage their operation. On the other hand, new additions to the UI elements of RCH will not disturb the operations of its internal components due to their independent workflow-managed nature. The testing results and WfMS integration advantages are further detailed in Table 6-4.

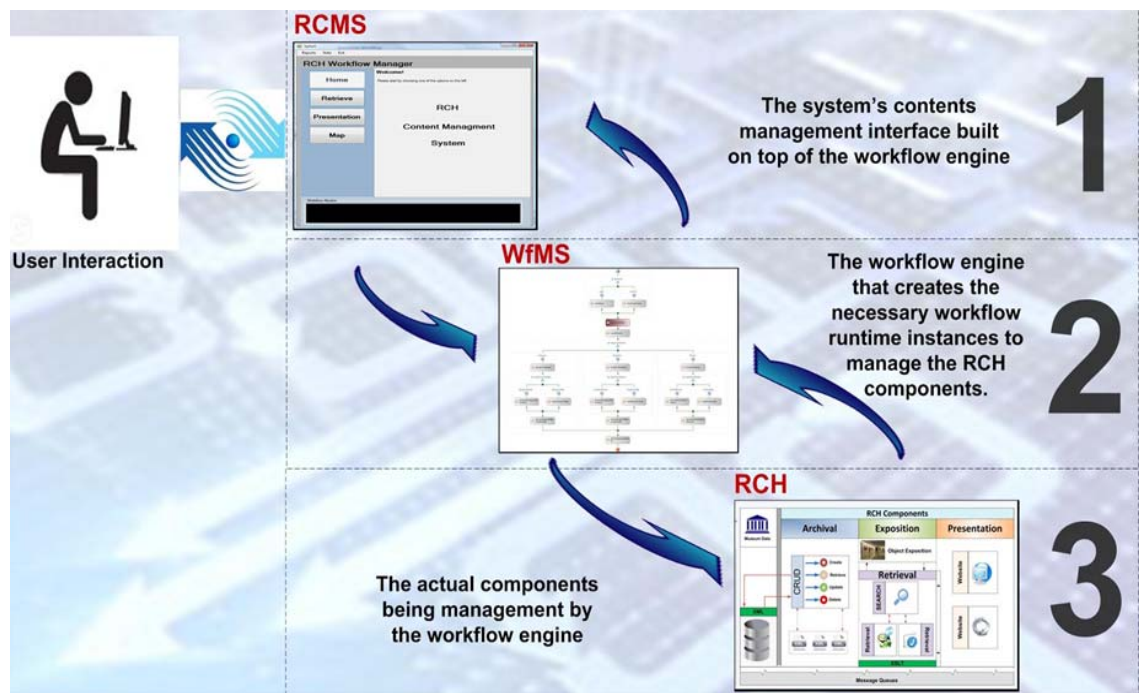


Figure 6-22 Workflow Integration Benefits

Table 6-4 RCH WfMS Testing Criteria and Results

Component/Process	Description	Criteria	Results
The workflow host (Client Application)	The main purpose here is to test the different functional and integration aspects of the workflow host.	Activation of individual work items.	Autonomous workflow initiation was achieved through the host application (RCMS) in corporation with the RCH WfMS as opposed to manual workflows/hardcoded system event management.
		Functionality of the UI controls.	The controls (buttons, dropdown menus, etc.) which we were wired up to the backend code classes functioned as expected as illustrated in Section 6.5.1.
		Integration between the workflow runtime services and the host.	Control over the workflow runtime services was achieved via the WfMS host. This allowed for the flexibility of having multiple system interfaces (MVC views) and multiple user sessions as shown in Sections 6.4.2, 6.4.3, 6.4.4 and 6.6.

Service Invocation	The aim is to test the base level workflow host and workflow engine functions that are used for the purpose of invoking the system services and workflow runtimes.	Workflow runtime service invocation.	Successful invocation was achieved through message passing as seen in Section 6.6.2.
		System function invocation and effective communication with the RCH components.	Function invocation was achieved through the interaction between the WfMS and the RCH components. For example, the mapping services are invoked through the messages that are passed to the RCH mapping component as seen in Section 6.6.2.
Workflow Runtime Instances	Testing the workflow runtime instances aimed at assessing the efficiency of workflow instance initiation and communication with the host application.	Workflow instance creation.	Successful workflow instance creation was achieved (See Figure 6-20).
		Workflow instance running and termination.	Instance creation and running was achieved through the messages passed to the WfMS from its host, e.g. creating and running the archival workflow runtime instance as shown in the test scenario in Section 6.6.1.

Message Passing	Efficiency of message passing was a determinant factor in assessing the efficiency of the RCH WfMS and its communication with the RCH components.	Data and message passing between the workflow host and the WfMS.	Successful parameter passing as shown in the examples in Figure 6-18 and Figure 6-19.
		Parameter passing to the workflow runtime services.	Successful parameter passing as shown in Figure 6-20.
		Parameter passing from the workflow runtime services to the system's components.	Successful parameter passing as shown in Figure 6-21.
Code Execution	Code Execution testing was conducted to assess RCH WfMS's efficiency in calling the right executable code blocks and executing them according to the defined workflow rules.	Method calling and code execution.	The messages and requests passed to RCH from the WfMS succeeded in achieving the required functionality by executing the right code blocks in the right sequence. For example, the data mapping process as shown in Section 6.6.1

Error Handling	Error handling testing was conducted to ensure that all runtime errors are handled properly without causing the program or process running to be interrupted in any way.	Runtime exception handling.	Building WfMS exception handling routines is outside the scope of this thesis, but the WF exception handling controls were used effectively to tackle any possible exceptions.
Other software required?	Is there a need for any other software to complement the WfMS functionality?	To assess the sufficiency and practicality of the devised WfMS.	The produced WfMS and its host are self-contained and do not require the support of any other software.

6.8 Future Adaption of RCH WfMS

RCH is a dynamic system by nature: this is due to the evolving needs of the communities of practice that use it and the varied digital objects that it manages. Therefore, it undergoes constant updates and upgrades to accommodate new functionality as well as enhancements to its current components. Such enhancements are usually accompanied with updates to the UI elements of RCH, namely its website frontend as well as the related RCH components. Figure 6-23 illustrates the object browsing page of the latest RCH version which has an enhanced look and feel.

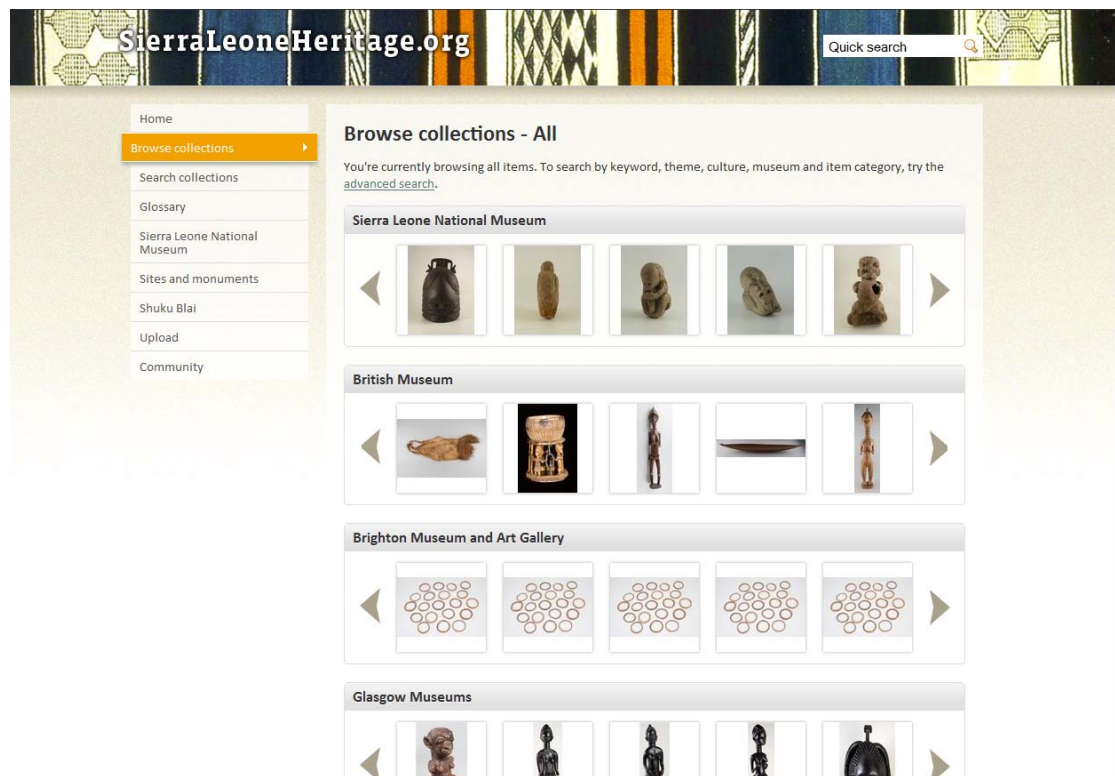


Figure 6-23 The latest RCH Search Interface

The question that manifests itself in this context is: will the RCH WfMS be able to accommodate future enhancements and new RCH components? In other words, will it be dynamic enough to adapt to the changes and upgrades that are applied to the RCH infrastructure. In fact, the RCH WfMS prototype was built in such a way that enables it to accommodate any upgrades within the RCH system (at database level, application level and presentation level). The RCH WfMS prototype was built with flexibility and adaptability in mind, as illustrated in Sections 6.26.3 and 6.4 as well as the results analysis in Table 6-4.

Typically, there are two types of change that an RCH implementation may undergo. The first is concerned with the UI elements of RCH and is related to its website frontend. These changes usually involve graphical and visual enhancements to the current RCH website at XHTML/CSS level (i.e. The View). For example, Figure 6-24 illustrates an enhanced search interface as opposed to the older version illustrated previously in Figure 5-20. The changes here are purely aesthetic and do not involve changes to the functionality of the page itself.

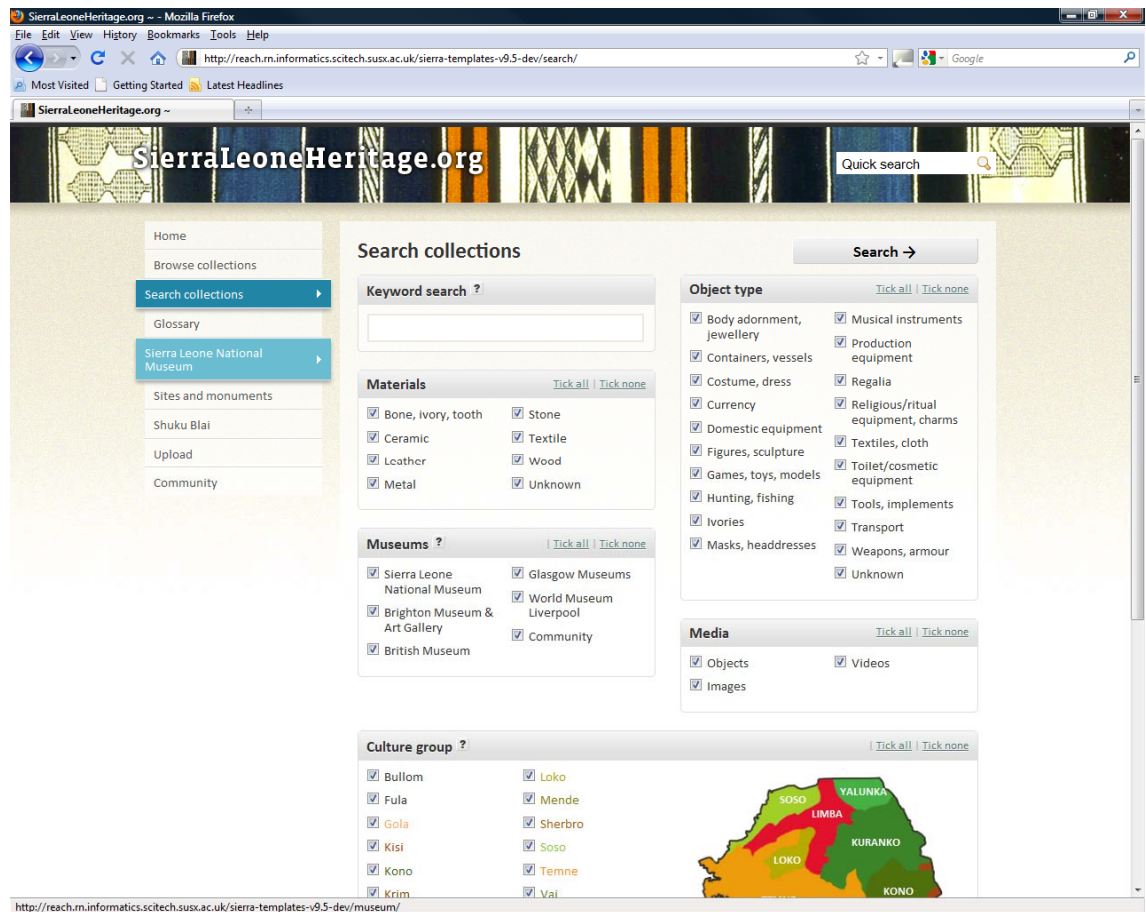


Figure 6-24 Enhanced Object Browsing View

Such changes at the UI level pose no problem to the RCH WfMS due to its SO nature and the interaction it maintains with the UI via message passing. Hence, non-functional changes do not require any changes in the devised RCH WfMS prototype.

The second types of change that the RCH WfMS has to accommodate are the ones concerned with business logic (i.e. Controller) within the application and database (i.e. Model) levels of the RCH implementation. Such changes involve actual code changes and extra functionality added to the RCH components. For example, the latest version of RCH has enhanced community-oriented functionality (community feeds) as well as

multimedia (videos) enchantments, which required making some changes to RCH's archival, retrieval and presentation components. Figure 6-25 illustrates some functional enhancements that are reflected in the RCH website frontend. Various community feeds are incorporated including videos. Such functional changes will require adding the appropriate workflow runtime services (or modifying the existing ones) to be plugged-in within the existing workflow management engine.

This flexibility is empowered by the flexible model that WF provides, as presented earlier in Table 6-1 and the way it was innovatively utilized to manage RCH components. Such newly-added/modified workflow runtime services will handle the new functionality and will perform the usual tasks of calling, invoking and managing the concerned RCH components.

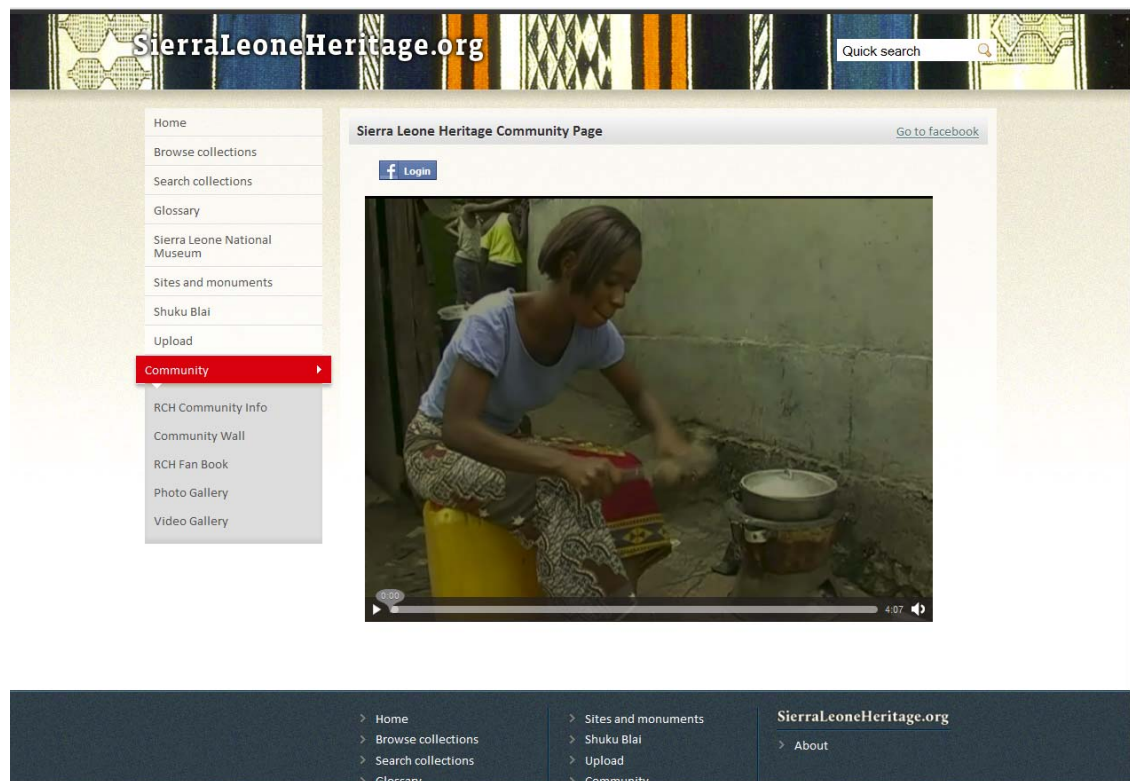


Figure 6-25 Community-oriented Multimedia Feeds

6.9 Summary

This chapter showcased the different aspects of incorporating a WfMS within the RCH system to manage its different components. The main technological medium that was used to build the prototype WfMS comprised the Windows Workflow Foundation (WF) that was used to build the actual workflow runtime services. The workflow runtime

services represented the core workflow management constructs that managed the illustrated representative RCH components (archival, retrieval and presentation).

The actual RCH WfMS is a highly hierarchical system where a number of rule-based sequential workflows were created to handle the different RCH components. The Windows .NET Framework was used to create a host (client) application in the form of a Windows Application (a web-based application can also be used but a Windows application was chosen to test the concept of integrating it with the already existing RCH website). The RCH WfMS utilized the UI provided by its host application to interact with its users through what is called the RCH Content Management System (RCMS).

The highly modular SOA that the RCH system adopted allowed for seamless integration with the encapsulated WfMS components. Such a paradigm paved the way for a smooth integration and interoperability between the integrated components. The second operational characteristic that made such integration possible was the effective message passing routines that were applied throughout the system components. Effective message exchange allowed the involved components to communicate with each other, allowing for process execution that involves various functionalities such as object mapping, search and retrieval, etc.

Running and testing the devised WfMS involved testing a number of its key operational scenarios. One of the illustrated scenarios was the object mapping process, which is a part of the archival component's services. A number of features were tested and evaluated, such as the efficiency of message passing and process execution. The WF workflow runtime indicators were used to determine the successes of runtime invocation and termination. Message passing was intercepted by utilizing the code debugging tools within the .NET Framework, allowing for assessing the efficiency of message exchange and process invocation and termination.

Incorporating the devised WfMS within the RCH system proved to be highly beneficial. RCH WfMS helped separate the visual elements of the RCH implementation from its baseline code infrastructure. This separation meant that optimal flexibility and scalability can be maintained while being able to serve the evolving system needs. Moreover, the incorporation of a WfMS provided management, monitoring and tracking

tools that helped better manage and coordinate the complex workflows that govern the RCH's behaviour. The benefits of such integration are further discussed in Chapter 7.

Figure 6-26 shows how the RCH WfMS prototype maps to some of the operational problems in the current RCH implementation. The problem of having some manual or semi-manual workflows (some steps in the archival object mapping process for example) is resolved by automated workflows that are run and managed by the devised WfMS. The WfMS provided tracking and management capabilities and contributed to enhancing the RCH system as a whole, resulting in better management of its digital heritage workflows as illustrated in Sections 6.5 and 6.6. On the other hand, the problem of inflexibility was overcome by the expansion and scalability opportunities that the RCH WfMS has offered. Inflexibility is tackled by the WfMS's tolerance for accepting enhancements both functional and at the UI levels, as was illustrated in Section 6.8. This flexibility is supported by the adopted SO approach and the modular nature of the RCH WfMS components.

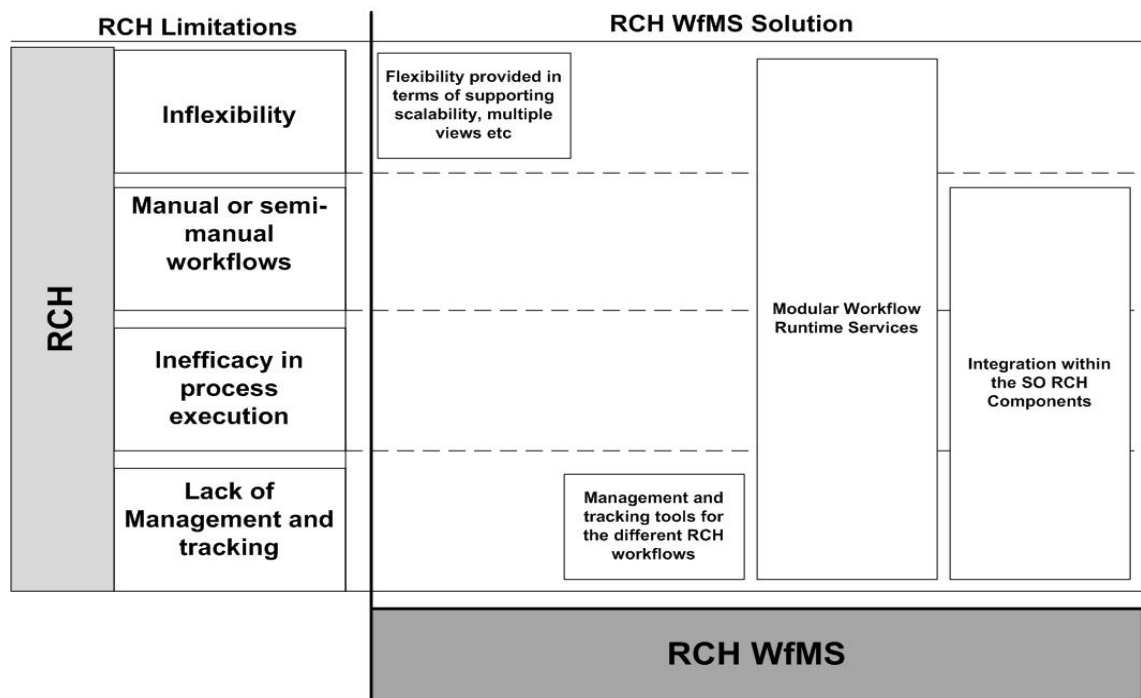


Figure 6-26 Mapping the WfMS to the RCH Problems

Chapter 7 discusses the conclusions in relation to the overall research work carried out as a part of this thesis, as well as the planned future work that will build on what has already been achieved.

CHAPTER VII

7 Conclusion and Future Work

This chapter provides the conclusions in relation to the research work carried out and presented in this thesis. It outlines the results of integrating a WfMS with an online digital heritage resource (represented in the RCH system) to manage its different components (archival, retrieval and presentation). The underlying details of the operational gains (more on this in Section 6.7) that resulted from WfMS integration are analysed in the context of the adopted architectural model and implementation paradigms (mainly the adoption of MVC within an SO implementation approach). The RCH WfMS implementation is discussed in terms of the key implementation issues that it encountered and overcame, and the actual mechanisms it used to manage the RCH components.

The contribution to knowledge that this thesis has made is discussed in conjunction with the examined concepts and the implemented prototypes (RCH and WfMS). The discussed contributions include the process of validating the DISPLAYS DLS framework by devising a DLS implementation (RCH) and building workflow prototypes of three of its core components (archival, retrieval and presentation). This was done to assess the validity of using managed workflows within a DLS. Emphasis is placed here on the advantages of integrating a WfMS within a digital heritage resource in terms of process and resource optimization and the management capabilities that it provides. The creative mix between a WfMS, a WfMS host and a digital heritage resource (RCH), is also discussed in terms of the mechanism used to make such integration a functional and viable one. This discussion is followed by elaborating on future work, planned to build on what has already been achieved in this thesis.

7.1 Results Analysis

The results analysis and findings are outlined below in sections that represent the key contributions made as a part of this thesis.

7.1.1 Validating the DISPLAYS Concept and WfMS Integration

RCH is a digital heritage resource implementation that is based on the DISPLAYS Framework (see Chapter 3). DISPLAYS acted as a DLS conceptual framework that encapsulated a number of SO services specifically designed to accommodate the functional needs of DLSs and DHRs. The main DISPLAYS services are the Digital Content: Creation, Archival, Exposition, Presentation and Interaction services for digital heritage collections. These services were mapped into actual DLS components within the RCH system, as presented in Chapters 4 and 5.

The RCH system was developed based on the DISPLAYS Framework while providing a number of functional components; the archival, retrieval and presentation components. These components were based on the equivalent DISPLAYS services and were implemented accordingly in an SO loosely-coupled manner. The RCH components were encapsulated within a flexible MVC model that separated the UI and presentation aspects of the system from its business logic. Therefore, the technological solutions (PHP, XML, XSLT and XHTML) and code modules (comprised of specialized code classes and routines) that constituted the RCH implementation were mapped into the MVC model. Consequently, the archival XML files represented the Model, the business logic XSLT/PHP files represented the Controller, and the dynamically generated XHTML code represented the View part of the model. These technologies suited an MVC implementation that met the needs of the RCH system and provided it with a flexible scalable model. The utilized XSLT files provided flexibility and efficiency in extracting data from the XML data model that held the details of the managed objects. XSLT files also suited the presentation layer (view) part of RCH (PHP, XHTML and CSS) by producing well-formatted valid XHTML to be displayed within RCH's UI.

The work presented in this thesis revolved around the idea of integrating a WfMS with an online digital heritage resource represented in the RCH prototype. Such integration aimed at managing the archival, retrieval and presentation components of RCH (built as example DLS components to validate the proposed WfMS) while offering a number of practical gains, as explained in Sections 6.7. Consideration was given to devise the RCH WfMS to fit within the SOA of RCH, without having to drastically modify or redesign the existing components to accommodate the WfMS components. The RCH WfMS was therefore built as an encapsulated component that hosted and managed the RCH components and their associated services (see Section 6.3). The RCH WfMS was

in turn hosted in an application program (Windows application) to provide the RCH users with an interactive UI, as illustrated in Section 6.5. It should be noted that the WfMS version was a limited prototype compared to the prototype RCH version, and later production versions of RCH (see Section 6.8).

The RCH components that were emphasized within the context of validating the proposed WfMS are the archival, retrieval and presentation components. These components were chosen because they represent some of the most complex components to manage due to the complex intersecting nature of their workflows. The actual implementation of the devised WfMS solution involved building an independent dynamic and adaptive workflow engine that comprised a number of workflow management components (workflow runtime services, workflow monitoring and tracking services, persistence service, etc.) as presented in Chapter 6. This solution was integrated with RCH while emphasizing its specific operational dynamics, such as message passing and object mapping operations, user interaction, etc., as can be seen in the testing scenario in Section 6.6.

The devised RCH WfMS was designed in such a way that it sat on top of the already built online heritage resource represented by RCH's functional components. This approach overcame the challenges of adapting to the dynamic changes that the system witnesses while it is being run. The adopted WfMS implementation model also comprised a consistent and flexible model that provided a number of modular services which interacted with RCH components. Such an interaction was achieved via effective communication between the WfMS and the RCH components via message passing and data exchange (for example digital heritage objects data mapping processes, archival procedures, search and retrieval, etc., as illustrated in Section 6.4).

The technological medium that was used to build the RCH WfMS prototype was the WF that was chosen for a number of practical reasons (see Section 6.3.1). One of the main reasons for choosing the WF was its support of modular workflow runtime services that can easily integrate with existing software systems. This feature was particularly important in the case of RCH due to its modular nature. The WF provided a solid and consistent implementation model that optimized the data transmission operations between the RCH components. The complex and intersecting control and data workflows of the RCH system were modelled within the devised WfMS as a set of sequential workflow runtime services as detailed in Section 6.4. A separate workflow

runtime service was created for each of the examined RCH components. The created workflow runtime services acted as a host for the RCH components and comprised the archival, retrieval and presentation workflow runtime services.

The RCH WfMS prototype was hosted in a host application represented in a standalone .NET Windows application. Hosting the WfMS was necessitated by the need to provide the system users with a UI to be able to utilize the management capabilities that the devised WfMS provides. The host application (RCMS) acted as another encapsulated modular component that interacted with the WfMS via message passing (the RCMS represented the actual manifestation of the devised WfMS functionality and services).

7.1.2 DLS Hosting within a WfMS

As explained in Chapter 6, the integration of a WfMS with the RCH system followed a layered approach that offered scalability and expandability capabilities that suited the heterogeneous nature of RCH (and potentially any similar DLS). One of the adopted techniques was to provide an independent presentation frontend that contributed to separating the workflow management logic (workflow management runtime services) from the exposition, presentation and interaction aspects of RCH. This separation proved to be a significant one imposed by the heterogeneous nature of RCH and the need for multiple views (multiple website frontends, standalone applications, touch screens, etc.) to suit the available user interaction modes (see Section 3.4.5).

The provided workflow management functionality revolved around the interaction between three main components, which are the RCH components, the WfMS prototype (the actual workflow engine), and the WfMS host. Flexibility was achieved by the development of a number of separate specialized workflow runtime services that were independent from the other RCH components (see Section 6.4.1). The RCH WfMS provided a workflow management middle layer (a medium between the RCH components and the RCMS) that is capable of adapting to the evolving system needs. The flexibility of the RCH WfMS stems from its flexibility to accommodate specialized workflow runtime services (current and future) to manage the operation of the RCH components and the needs of its communities of practice.

7.1.3 Effective Communication between the RCH Components

Effective communication was maintained through message passing between the system components. Message passing involved a number of key system processes that included RCH service invocation, data exchange, service response, workflow service invocation and correlation. Message passing provided for a simple and efficient approach for maintaining the system's state, while ensuring that the right sequence of workflow activities are being invoked and terminated in response to the system events (for example, search operations, data mapping, etc., as shown in the testing scenario in Section 6.6).

This approach had the advantage of being code and technology independent as it depended on parameter passing (requests and commands, see Table 6-4) between the RCH components, regardless of their underlying implementation details (RCH prototype is a PHP website, the WfMS is based on the WF and the RCMS is a Windows application). Successful running of message passing was illustrated in Section 6.6, where it was shown that workflow runtime service invocation and termination was adequately handled through the managed service and function calls. Message passing was also utilized at the UI level (RCMS) where parameters are passed from the frontend to the workflow host. The workflow host in turn passes the received messages to the other system components to perform the required operations.

7.1.4 WfMS Integration Gains and Advantages

Testing the WfMS as outlined in Section 6.6 and the results in Section 6.7, gave an indication that it succeeded in meeting the workflow management needs of RCH across the tested functional components. The integration of a WfMS within RCH offered a number of practical operational advantages as listed below. The examined operational scenarios in Chapter 6 show that the incorporation of the prototype WfMS led to a number of operational and management gains including:

- the provision of integrated management, tracking, threading and monitoring services that contributed to better running of RCH as well as optimum management of its resources components;
- autonomous workflow initiation and running was achieved through the host application (RCMS) in conjunction with the RCH WfMS, as opposed to manual workflows/hardcoded system event management;

- separating the system's business logic from its presentation, exposition and interaction components, paving the way for multiple MVC Views to serve the different user needs;
- modularity, scalability and expandability were enhanced by the RCH WfMS's ability to accommodate and manage any new code components without disturbing the overall operation of the system (compared to non-managed components that require restructuring to accommodate new additions);
- the utilization of a workflow management model within a number of separate runtime services succeeded in mapping the different system processes. An example is the digital heritage data mapping process as shown in Section 6.4.2 and tested in Section 6.6.1;
- the way that the devised components (RCH components, the WfMS and the RCMS) interfaced with each other meant that they had the ability to adapt to the changes that any of them may undergo.
- the adopted SO approach meant that the RCH communications are purely service-based, i.e. based on the process of calling or terminating a certain service regardless of its underlying implementation details. Hence, flexibility was provided in terms of integrating different DLS services and components;
- while the demonstrated scenarios involved a selective set of functionality involving the examined components, the prototype WfMS can be expanded to accommodate any new functional requirements by exploiting the adopted SO architectural approach.

7.2 Future Work

The planned future work will build on the code infrastructure and the developed conceptual model that formed the solution presented in this thesis. The intended future work will involve a number of steps to incorporate a more advanced WfMS implementation model, especially in terms of handling the dynamic change of sequential process running, specifically within the workflow runtime services. The adopted SOA meant that the improvement of such a model is a relatively straightforward process that involves adding more functional modules that are capable of complementing the current functionality.

RCH itself can also be improved by accommodating more complex operational scenarios that involve the integration of disparate databases and DLSs, while acting as an integration and service provision medium. Such a medium can consolidate and utilize the resources available in any given digital heritage resource environment. The provided SOA model can be further enhanced by the provision of a web-service based version that handles the system's services through published web services that comprise better distribution and accessibility features. These web services can also provide workflow management functionality. The future work plans also include improving the created WfMS prototype to be integrated within the latest version of RCH as illustrated in Section 6.8.

The automated archival mapping process can be further improved by incorporating more sophisticated functionality and error checking routines. One of the common problems in the mapping process is the existence of missing object information. Therefore, the future versions of the RCH mapping tool are planned to have intelligent logic to handle such a problem. Ideally, the mapping tool will be able to spot any missing object information, alert the user about that, suggest possible values or prompt the user to enter a suitable alternative attribute. This mechanism will enhance the validity of the outcome of the mapping process, thus minimize errors during the cultural heritage object data import/export operations.

One of the ambitious goals that form a part of the overall attempt presented in this thesis is to arrive at a more generic WfMS model (based on the RCH WfMS prototype) that can suit virtually any DLS or DHR implementation. The main goal here is to provide a set of highly customizable tools that can be easily integrated with existing code and system infrastructures to provide flexible, customizable and user-friendly workflow management capabilities.

8 Bibliography

- [1] Wei Zhang, Zeeshan Patoli, Michael Gkion, Abdullah Al-Barakati, Paul Newbury and Martin White "Reanimating Cultural Heritage through Service Orientation, Workflows, Social Networking and Mashups," in *International Conference on CYBERWORLDS*, Bradford, 2009, p. 7.
- [2] Maureen Pennock, "Digital Curation and the Management of Digital Library Cultural Heritage Resources," vol. 25, no. 2, 2006.
- [3] L. Candela, D. Castelli, P. Innocenti, Y. Ioannidis, A. Katifori, A. Nika, G. Vullo, S. Ross G. Athanasopoulos, "The Digital Library Reference Model," Glasgow, 2009.
- [4] W.M.P. Aalst, "Flexible Workflow Management Systems: An Approach Based on Generic Process Models," *Database and Expert System Application* vol. 1677, no. 1, 1999.
- [5] Abdullah Albarakati, Zeeshan Patoli, Michael Gkion, Wei Zhang, Paul Newbury, Natalia Beloff, and Martin White "A Dynamic Workflow Management Framework for Digital Heritage and Technology Enhanced Learning," in *Virtual Systems for Multimedia dedicated to Digital Heritage*, limassol, 2008.
- [6] Abdullah Albarakati, Zeeshan Patoli, Michael Gkion, Wei Zjang, Natalia Beloff, Martin White "An Integrated Workflow Management Solution for Heritage Information Mashups," in *IEEE ASONAM*, vol. 1, Athens, 2009, p. 6.
- [7] Zeeshan Patoli, Abdullah Al-Barakati, Michael Gkion, Wei Zhang, Paul Newbury, Natalia Beloff, and Martin White "A Service-Oriented Approach for a Digital Library System focused on Portable Antiquities and Shared Heritage," in *The 8th International Symposium on Virtual Reality, Archaeology and Cultural Heritage*, Brighton, 2007.
- [8] Zeeshan Patoli, Michael Gkion, Abdullah Al-Barakati, Wei Zhang, Paul Newbury, Martin White "How to Build an Open Source Render Farm based on Desktop Grid Computing," in *International Multi-topic Conference IMTIC 2008*, vol. 20, karachi, 2008.
- [9] Michael Gkion, Zeeshan Patoli, Abdullah Al-Barakati, Wei Zhang, Paul Newbury and Martin White, "Collaborative 3D Digital Content Creation Exploiting a Grid Network," in *Third International Conference on Information & Communication Technologies 2009*, Karachi, 2009, p. 6.
- [10] Ricardo Garcês, Tony de Jesus, Jorge Cardoso, "Open Source Workflow Management Systems: A Concise Survey," Madeira, 2007.
- [11] Nie Gang, "A Scheme Of Workflow Management System Based On Web Services," in *Electronic Commerce and Security, 2008 International Symposium on Electronic Commerce and Security*, vol. II, Guangzhou, 2008.
- [12] Philippa Collins, "Document and workflow management '99," vol. 44 pp.23-26, no. 1, 2000.
- [13] Edmund Balnaves. (2000) Digital Assets Content Management System. PDF.
- [14] Nabil Adam, Bharat K. Bhargava, Milton Halem, Yelena Yesha, "Digital Libraries. Research and Technology Advances: ADL'95 Forum," McLean,

Virginia, 1996.

- [15] Jan Mendling. (2006) Business Process Execution Language for Web Service (BPEL). PDF.
- [16] Zeeshan Patoli, Michael Gkion, Abdullah Al-Barakati, Wei Zhang, Paul Newbury and Martin White "An Open Source Grid Based Render Farm for Blender 3D," in *IEEE Power Systems Conference & Exhibition*, vol. 12, Seattle Washington, 2009.
- [17] Margaret McGrory, Carol Pollitt, Paivi Bente Dahl Rathje, "Designing and Building Integrated Digital Library Systems - Guidelines," Hague, 2005.
- [18] Alex Byrne, "The end of history: censorship and libraries," in *Beacon on Freedom of Expression Conference*, vol. 1, Alexandrina, 2003.
- [19] Donald G., Jr. Davis, Nicholas A. Basbanes, Stuart A. P. Murray, *The Library: An Illustrated History*. Chicago: American Library Association, 2009, vol. II.
- [20] Barbara Krasner-Khait, "Survivor: The History of the Library," New York, 2001.
- [21] Scott Bennett, "Libraries and Learning: A History of Paradigm Change," vol. 9, no. 3, 2009.
- [22] Ahmad Bakeri Abu Bakar, "Education For Digital Libraries In Asian Countries," in *Asia-Pacific Conference on Library & Information Education & Practice*, 2009, vol. 1, Tsukuba, 2009.
- [23] Duncan M. Aldrich, Gregory Stefanelli, "Library Services for a Digital Future," vol. 29, no. 15, 2006.
- [24] Yannis Ioannidis, "Digital libraries at a crossroads," vol. 5, no. 4, 2007.
- [25] Lilia Pavlova-Draganova, Desislava Paneva, "Digital Libraries For Presentation And Preservation Of East-Christian Heritage," Sofia, 2005.
- [26] Michael Lesk, "Understanding Digital Libraries, Second Edition (The Morgan Kaufmann Series in Multimedia and Information Systems)," vol. 1, no. 1, 2004.
- [27] William C. Janssen, "International Conference on Digital Libraries," vol. 14, no. 1, 2003.
- [28] Stacey Greenaway, "Digital Libraries – Literature Review," Wolverhampton, 2006.
- [29] Vittore Casarosa, "DELOS Reference Model for Digital Libraries," in *Elag 2007 Conference*, vol. 1, Barcelona, 2007.
- [30] Steven M. Schermerhorn, "Integrated Library System goals and accomplishments at Goleman Library," Stockton, 2008.
- [31] Jamie Callan, Jie Lu, "Merging retrieval results in hierarchical peer-to-peer networks," in *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, vol. 15, Sheffield, 2004.
- [32] Georg Rehm, Andreas Witt, Felix Zimmermann, Timm Lehmberg, "Digital Text Collections, Linguistic Research Data, and Mashups: Notes on the Legal Situation," vol. 57, no. 1, 2008.
- [33] LC NDLP, "Digital Libraries Initiative," 2010.
- [34] P. Sreejaya, M.G. Sreekumar, "Digital Library Initiatives and Issues in India : Efforts on Scholarly Knowledge Management," Calicut, 2005.
- [35] Leonardo Candela, David Lievens, Pasquale Pagano, Manuele Simi Fabio

- Simeoni, "Functional adaptivity for digital library services in e-infrastructures: the gCube approach," in *ECDL'09 Proceedings of the 13th European conference on Research and advanced technology for digital libraries*, vol. 36, Corfu, 2009.
- [36] Hong (Iris) Xie, "Help features in digital libraries: types, formats, presentation styles, and problems," vol. 34, no. 6, 2007.
- [37] Sueli Mara Ferreira, Denise Nunes Pithan, "Usability of digital libraries: a study based on the areas of information science and human-computer-interaction," in *World Library and Information Congress: 71st IFLA General Conference and Council*, vol. 1, Oslo, 2005.
- [38] Yaohua Yu, Zhengjie Liu, "Research on System Usability of Digital Libraries in China," in *4th IEEE International Symposium on Electronic Design, Test & Applications*, vol. 1, Hong Kong, 2008.
- [39] Bruce R. Schatz, "Information Retrieval in Digital Libraries: Bringing Search to the Net," vol. 275, no. 3, 1997.
- [40] Lloyd A. Smith, Ian H. Witten, Clare L. Henderson, Sally Jo Cunningham, Rodger J. McNab, "Towards the digital music library: tune retrieval from acoustic input," in *the first ACM international conference on Digital libraries*, vol. 84, Bethesda, 1996.
- [41] William Y. Arms, Christophe Blanchi, Edward A. Overly , "An Architecture for Information," Reston, Virginia, 1997.
- [42] Brian Hoffman, "Supporting Digital Archival Workflows at NYU," New York, 2000.
- [43] Amy Friedlander, Roger Schonfeld, Sayeed Choudhury Lorraine Eakin, "A Selective Literature Review on Digital Preservation Sustainability," La Jolla, 2007.
- [44] Nicolas Spyrtos, Carlo Meghini, Jitao Yang, (2009) A Data Model for Digital Libraries. Power Point.
- [45] Daniel Greenstein, "Digital Libraries and Their Challenges," vol. 49, no. 3, 2001.
- [46] Edward A. Fox, Devika P., Madalli Hussein Suleman, "Design and Implementation of Networked Digital Libraries: Best Practices," Bangalore, 2003.
- [47] Alan Hopkinson, "Challenges for the Digital Libraries and Standards to Solve them," in *7th International CALIBER-2009*, vol. 2, Puducherry, 2009.
- [48] Alan McCord, "Overview of Digital Asset Management Systems," Michigan, 2002.
- [49] Enrico Motta, John Domingue, Simon Buckingham Shum, "ScholOnto: an ontology-based digital library server for research documents and discourse," vol. 3, no. 1, 2000.
- [50] Laura C. Savastinuk, Michael E. Casey, "Service for the next-generation library," vol. 33, no. 14, 2006.
- [51] Jane Secker, "Social Software, Libraries and distance learners: literature review," London, 2008.
- [52] Hu Xiaojing, Fan Bingsi, "Library 2.0: Building the New Library Services," 2006.
- [53] Jack M. Maness, "Library 2.0 Theory: Web 2.0 and Its Implications for Libraries," vol. 8, no. 2, 2006.
- [54] Donatella Castelli, Pasquale Pagano, Constantino Thanos, Leonardo Candela, "Setting the Foundations of Digital Libraries," vol. 13, no. 1, 2007.

- [55] M. Chandrashekara, N. Varatharajan, "Digital Library Initiatives at Higher Education and Research Institutions in India," vol. 1, no. 1, 2007.
- [56] Rick Prelinger, Mary E. Jackson, Brewster Kahle, "Public Access to Digital Material," vol. 7, no. 1, 2001.
- [57] Mark Martinez, Jeff Scott, Mariella Di Giacomo, "A Large-Scale Digital Library System to Integrate Heterogeneous Data of Distributed Databases," vol. 3149, no. 10, 2004.
- [58] Kostas Saidis, Mara Nikolaidou, Vassilios Karakoidas, George Pyrounakis, "Introducing Pergamos: A Fedora-based DL System Utilizing Digital Object Prototypes," Athens, 2005.
- [59] David Bainbridge, David M. Nichols, Ian H. Witten, *How to Build a Digital Library (Morgan Kaufmann Series in Multimedia Information and Systems)*. Massachusetts: Morgan Kaufmann, 2009, vol. 2.
- [60] Hwwilson. (2010, Mar.) Hwwilson. [Online].
<http://www.hwwilson.com/databases/flyers/ERICflyer.pdf>
- [61] ERIC. (2010, Apr.) ERIC. [Online].
http://www.eric.ed.gov/ERICWebPortal/resources/html/about/about_eric.html
- [62] Cesare Concordia, Carlo Meghini, Nicola Aloia, "Querying A Bricks Digital Library," in *IADIS International Conference WWW/Internet*, vol. 1, Vila Real, 2007.
- [63] Robert Hecht, Bernhard Haslhofer, "Joining the BRICKS Network - A Piece of Cake," Wien, 2005.
- [64] Thomas Risse, Predrag Knežević, Carlo Meghini, Robert Hecht, Fiore Basile, "The BRICKS Infrastructure - An Overview," 2005.
- [65] Nicholaos Mourkoussis, Joe Darcy, Panos Petridis, Fotis Liarokapis, Paul Lister, Krzysztof Walczak, Rafał Wojciechowski, Wojciech Cellary, Jacek Chmielewski, Mirosław Stawniak, Wojciech Wiza, Manjula Patel, James Stevenson, John Manley, Fabr Martin White, "ARCO — An Architecture for Digitization, Management and Presentation of Virtual Exhibitions," in *Computer Graphics International 2004*, Crete, Greece, 2004, pp. 622-625.
- [66] Chris Rusbridge, Peter Burnhill, Seamus Ross, Peter Buneman, David Giaretta, Liz Lyon, Malcolm Atkinson, "The Digital Curation Centre: A Vision for Digital Curation," in *The Digital Curation Centre: a vision for digital curation*, vol. II, Edinburgh, 2005.
- [67] Gary Marchionini, Chirag Shah, "Capturing Relevant Information for Digital Curation," in *Bulletin of IEEE Technical Committee on Digital Libraries*, vol. 4, North Carolina, 2008.
- [68] Christopher A. Lee, Helen R. Tibbo, John C. Schaefer, "Defining what digital curators do and what they need to know: the digcurr project," in *7th ACM/IEEE-CS joint conference on Digital libraries*, vol. II, Vancouver, 2007.
- [69] Joyce Ray, "Managing the Digital World: the Role of Digital Curation," Edinburgh, 2008.
- [70] Joe Lin, Charley Ho, Wasim Sadiq, Maria E. Orlowska, "Using Workflow Technology to Manage Flexible e-Learning Services," vol. 1436, no. 3, 2002.
- [71] Wil M.P. van der Aalst Hajo A. Reijers, "The effectiveness of workflow management systems: Predictions and lessons learned," vol. 25, no. 1, 2005.

- [72] van der Aalst, Kees van Hee, *Workflow Management: Models, Methods, and Systems*. Cambridge, US: The MIT Press, 2004, vol. I.
- [73] Yang Guang-Xin, Xiang Yong, WU Shang-Guang SHI Mei-Lin, "WFMS:WORKFLOW MANAGEMENT SYSTEM," vol. III, 1999.
- [74] Thomas Schael, *Workflow management systems for process organisations*. Rome: Springer, 1996, vol. 12.
- [75] Lalitha Munaga, Kamalakara Karlapalem, Rupa Krishnan, "XDoC-WFMS: A Framework for Document Centric Workflow Management System," vol. 2465/2002, no. I, 2002.
- [76] WfMC. (1996) Reference model and API specification. PDF.
- [77] Hongbing Liang, Bin Xu, Mingkui Yang, "S-WFMS: A Service-based Workflow Management System," in *International Conference on Advanced Information Networking and Applications (AINA'05)*, vol. V1, Taipei, 2005.
- [78] Jia Yu, Rajkumar Buyya, "A Taxonomy of Workflow Management Systems for Grid Computing," vol. 3, no. 1, 2006.
- [79] Qinglong Zhan, "Workflow Technology Enabled e-Training System: Toward Work Process," in *Information Engineering and Electronic Commerce, 2009. IEEEC '09. International Symposium on*, vol. 108, Tianjin, 2009.
- [80] François Charoy, Adnene Guabtni, Miguel Valdes Faura, "A Dynamic Workflow Management System for Coordination of Cooperative Activities," *Business WorkShop* vol. 4103, no. 1, 2006.
- [81] M Zisman. (1977) Representation, Specification and Automation of Office. PDF.
- [82] Clarence A. Ellis, "Information Control Nets: A Mathematical Model of Office Information Flow," vol. 8, no. 1, 1979.
- [83] Ingmar A. Frey, "Measuring the Effectiveness of a Workflow Management System in a Pre-Surgical Process," vol. 1, no. 1, 2009.
- [84] David Hollingsworth, "Workflow Management Coalition The Workflow Reference Model," vol. 1003, no. 1.1, 1994.
- [85] J.H.P. Eloff, R.A. Botha, "A Security Interpretation of the Workflow Reference Model," in *th Working Conference of WG11.1 and WG11.2 of IFIP TC11*, vol. 34, Vienna, 1998.
- [86] Daniel Rolli, Rudolf K. Keller, Peter Kropf, Sarita Bassil, "Extending the Workflow Reference Model to Accommodate Dynamism," Montréal, 2003.
- [87] David Hollingsworth, "Workflow Management Coalition The Workflow Reference Model," Winchester, 1995.
- [88] Martinho Fernandes, Sergio Miguel, (2006) A Workflow Virtual Machine. PDF.
- [89] Saini. (2007) Seven Showstopper Problems with BPEL Servers for Event-Driven SOA. Website.
- [90] Lachlan Aldred, Marlon Dumas, Arthur H.M. ter Hofstede, Wil M.P. van der Aalst, "Design and Implementation of the YAWL System," vol. 3084/2004, no. I, 2004.
- [91] Wil van der Aalst, Carmen Bratosin. (2007) Workflow Management Systems for Grid Computing. Web.
- [92] Wil van der Aalst, A.H.M. ter Hofstede, "YAWL: yet another workflow language," vol. 245–275, no. 12, 2004.

- [93] Jordi Anguela Rosell, Christophe Loridan, "BONITA – Workflow patterns support," Les Clayes-sous-Bois, 2006.
- [94] Bilal Siddiqui. (2010) Bonita for business process management, Part 1: Configure a simple workflow. PDF.
- [95] BonitaSoft. (2009) Bonita Open Solution, a comprehensive Open Source BPM Suite. Website.
- [96] Farzad Farahbod, Ajay Aggrwal, "Create a Windows workflow application using Windows Workflow Foundation in IBM Database Add-ins for Visual Studio," USA, 2008.
- [97] Allen Scott, *Programming Windows Workflow Foundation: Practical WF Techniques and Examples using XAML and C#: A C# developer's guide to the features and programming interfaces of Windows Workflow Foundation*. Birmingham: Packt Publishing, 2006, vol. II.
- [98] Bruce Bukovics, *Pro WF: Windows Workflow in .NET 3.5*. New York: Apress, 2008, vol. 1.
- [99] Marcel de Vries. (2007) Practical Windows Workflow Foundation. PDF.
- [100] Inala Uma Shankar. (2008) Using the Windows Workflow Foundation (WF) for Developing an Issue Management System. Website.
- [101] Y. Wu, F. Hernandez, F. Ortega, P. J. Clarke, R. France, "Measuring the effort for creating and using domain-specific," in *Proceedings of 10th DSM Workshop*, 2010.
- [102] Robin Roy. (2009, July) Comparison between Windows Workflow Foundation and Biz Talk Server. [Online]. <http://www.codeproject.com/KB/biztalk/BizTalk-WF-Compare.aspx>
- [103] Julian Jang, Alan Fekete, Paul Greenfield, Surya Nepal, "An Event-Driven Workflow Engine for Service-based Business Systems," in *Enterprise Distributed Object Computing Conference - EDOS*, Hong Kong, 2006, pp. 233-242.
- [104] Muriel Foulonneau. (2007) Digital repositories infrastructure vision for European research - Review of technical standards. PDF.
- [105] Donatella Castelli. (2007) DRIVER: Digital Repository Infrastructure Vision for European Research. PDF.
- [106] Amy Shuen, *Web 2.0: A Strategy Guide*.: O'Reilly Media, 2008, vol. 1.
- [107] Barbara Clubb, Jennifer-Lynn Draper, Alexandra Yarrow, "Public Libraries, Archives and Museums: Trends in Collaboration and Cooperation," Edinburgh, 2008.
- [108] Minerva. (2008) Technical Guidelines for Digital Cultural Content Creation Programmes. PDF.
- [109] Jeffrey Pomerantz. (1999) Integrating Digital Reference Service into the Digital Library Environment. PDF.
- [110] Radoslav Pavlov, Desislava Paneva, , "Toward Ubiquitous Learning Application of Digital Libraries with Multimedia Content," vol. 6, no. 3, 2006.
- [111] NISO. (2008, June) National Information Standards Organization. [Online]. <http://framework.niso.org/node/8>
- [112] NRGL. (2008) Comparison of Selected Software Systems for Creation of Digital Libraries. PDF.
- [113] Stephan Kiemle, "From Digital Archive to Digital Library - A Middleware for

- Earth-Observation Data Management," in *ECDL '02 Proceedings of the 6th European Conference on Research and Advanced Technology for Digital Libraries*, vol. 22, Roma, 2002.
- [114] Konstantin Rangochev, Detelin Luchev, Desislava Paneva, "Knowledge Technologies For Description Of The Semantics Of The Bulgarian Folklore Heritage," in *Fifth International Conference (Information Research And Applications 2007)*, vol. 2, Varna, 2007.
- [115] David Raitt. (2000) Digital Library Initiatives Across Europe. Website.
- [116] Oystein Pettersen, Nicole Bordes, Sean Ulm, David Gwynne, Terry Simmich, Bernard Pailthorpe, "Grid services for e-archaeology," in *AusGrid '08 Proceedings of the sixth Australasian workshop on Grid computing and e-research*, vol. 82, Australasian, 2008.
- [117] Gert Brettlecker, Tiziana Catarci, Stavros Christodoulakis, Tom Crecelius, Nektarios Gioldasis, Hans-christian Jette , Mouna Kacimi, Diego Milano, Paola Ranaldi, Harald Reiterer, Hans-jörg Schek, Heiko Schuldt Ceri Binding, "DelosDLMS: Infrastructure and Services for Future Digital Library Systems," Basel, 2007.
- [118] Nick J. Avis, Omer F. Rana, Gao Shu, "Bringing semantics to visualization services," vol. 39, no. 1, 2008.
- [119] Ray R. Larson, Robert Sanderson, "Grid-based digital libraries: cheshire3 and distributed retrieval," in *JCDL '05 Proceedings of the 5th ACM/IEEE-CS joint conference on Digital libraries*, vol. 15, Denver, 2005.
- [120] P. Jabisetti, N. Joshi, U. Lee, Y. Uppuluri, "P2P grid: service oriented framework for distributed resource management," in *Services Computing, 2005 IEEE International Conference on*, vol. 1, Orlando, 2005.
- [121] Paul Basu. (2009) Reanimating cultural heritage: digital repatriation, knowledge networks and civil society strengthening in post-conflict Sierra Leone. [Online]. <http://projects.beyondtext.ac.uk/reanimatingculturalheritage/index.php>
- [122] Sallieu Turay. (2004) Sierra Leone Book Trust Sierra Leone Book Trust For The Dialogue Of African Partners-Ii. PDF.
- [123] Paul Basu, "Confronting the Past? Negotiating a Heritage of Conflict in Sierra Leone," *Journal of Material Culture*, vol. 13, no. 2, pp. 153-167, 2008.
- [124] Paul Basu, Sharon Macdonald, *Exhibition Experiments*, 1st ed.: Wiley-Blackwell, 2007.
- [125] Akmal B. Chaudhri, Awais Rashid, and Roberto Zicari, *XML Data Management: Native XML and XML-Enabled Database Systems.*: Addison-Wesley Professional, 2003, vol. 1.
- [126] Jay Liebowitz, *Social Networking: The Essence of Innovation.*: The Scarecrow Press, Inc, 2007.
- [127] Ralph LeVan, Bruce Washbur, Günter Waibel. (2010) Museum Data Exchange: Learning How to Share. PDF.
- [128] Tony Marston. (2010, May) Rapid Application Development toolkit for building Administrative Web Applications. [Online]. <http://www.tonymarston.net/php-mysql/model-view-controller.html>
- [129] Frederic P. Miller, Agnes F. Vandome, John McBrewster, *Model-view-controller: Software Engineering, User Interface, Active Record Pattern, Architectural Pattern (computer science), Model 1, Observer Pattern, Presentation-*

abstraction- control.: Alphascript Publishing, 2010.

- [130] Sun. (2009, Jan.) Designing Enterprise Applications with the J2EE. [Online]. http://java.sun.com/blueprints/guidelines/designing_enterprise_applications_2e/web-tier/web-tier5.html
- [131] Hiroyuki Kawano, "Towards Digital Archive Systems: Architecture," in *The Eighth International Symposium on Operations Research and Its Applications*, Zhangjiajie, China, 2009.
- [132] Shana D. Kelley, Amy J. Hatfield. (2007) Case study: lessons learned through digitizing the National Commission for the Protection of Human Subjects of Biomedical and Behavioral Research Collection. Document.
- [133] Martin White, Manjula Patel, Jacek Chmielewski, Krzysztof Walczak Nicholas Mourkoussis, "AMS: metadata for cultural exhibitions using virtual reality," in *Dublin Core Conference 2003: Supporting Communities of Discourse and Practice - Metadata Research and Applications*, Seattle, Wa., USA, 2003.
- [134] Gregory Sherman, "A Critical Analysis of XSLT Technology for XML Transformation," 2008.
- [135] IBM. (2008, Aug.) DB2 solution Information Center home. [Online]. <http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/index.jsp?topic=/com.ibm.db2.luw.xml.doc/doc/c0050648.html>
- [136] Victor Pavlov. (2003) Build an XML/XSLT driven Website with.NET. PDF.
- [137] Ruizhi Sun, Dong Wang, *Realization Of Workflow Service Invocation Interface For Integration Of Agricultural Network Resources.*: Springer, 2009, vol. 2.
- [138] Qifeng Huang, Yan Huang, "WS-based workflow description language for message passing Symposium on ," in *Cluster Computing and the Grid, 2005. CCGrid 2005. IEEE International* , vol. 1, 2005.
- [139] David Schmitz, Andreas Hanemann, "Service-Oriented Event Correlation - Workflow and Information Modeling Approached," Munich, 2005.
- [140] Richard Blewett, "Windows Workflow Foundation," Los Angeles, 2005.
- [141] Ann Q. Gates, Leonardo Salayandía, "Towards a workflow management system for service," vol. 3, no. 18-25, 2007.
- [142] Philipp Leitner, Anton Michlmayr, Predrag Celikovic, Schahram Dustdar, Florian Rosenberg, "Towards Composition as a Service - A Quality of Service Driven Approach," in *IEEE International Conference on Data Engineering*, vol. III, 2009.
- [143] Kent Brown. (2007, February) BizTalk Server 2006 or WF? Choosing the Right Workflow Tool for Your Project. [Online]. http://msdn.microsoft.com/en-us/library/cc303238.aspx#BizTalk06_or_WF_topic3
- [144] Alan Snyder, "Encapsulation and Inheritance in Object-Oriented Programming Languages," vol. 204-7, 1986.
- [145] Suk-Ho Kang, Dongsoo Kim, Joonsoo Bae, Kyung-Joon, Ju, Yeongho Kim, "WW-FLOW: Web based workflow management with runtime encapsulation," in *Internet Computing, IEEE* , vol. 4, 2000.
- [146] K. Scott Allen. (2006, August) odeTocode.com. [Online]. <http://odetocode.com/Articles/457.aspx>
- [147] Pegasus. (2008) Building Image-Based Workflows with Windows Workflow Foundation. PDF.

- [148] Kenji Takeda, Simon J., Cox A.Paventhian, "Leveraging Windows Workflow Foundation for Scientific Workflows in Wind Tunnel Applications," in *the 22nd International Conference on Data Engineering Workshops*, vol. II, 2006.
- [149] C. S. Shin, T. S. Chou, Y.C. Wang, H. Y. Huang, W. S. Chen, J.W. S. Liu, "EMWF: A Middleware for Flexible Automation and Assistive Devices," Taipei, 2009.
- [150] G. Salaün, C. Canal, E. Pimentel, Javier Cubo, "Relating Model-Based Adaptation and Implementation Platforms: A Case Study with WF/.NET 3.0," berlin, 2007.
- [151] Brian Noyes, "Workflow Driven Windows Applications," Las Vegas, 2006.
- [152] Hubert Chao, Theodore Chao, Yim Cheng, Raymond Doyle, Sergey Grankin, Jon Guarino, Saikat Guha, Pei-Chen Lee, Dan Perry, Christopher Re, Ilya Rifkin, Tingyan Yuan, Dora Abdulla, Chavdar Botev, "Supporting Workflow in a Course Management System," USA, 2005.
- [153] Chappell & Associates David Chappell, *Introducing Microsoft Windows Workflow Foundation: An Early Look.*: Microsoft Corporation, 2005, vol. 1.
- [154] Zhao Xiaohui, Zhan Dechen "Performance Analysis of Hierarchical Workflow Management Systems on the Basis of BCMP-Queueing Networks," in *European Accounting Information Systems Conference*, vol. 3, Copenhagen, 2002.
- [155] Kristof Steurbaut, Sofie Van Hoecke, Filip De Turck, Bart Dhoedt, Bart J.F. De Smet, "Dynamic Workflow Instrumentation for Windows Workflow Foundation," in *International Conference on Software Engineering Advances (ICSEA 2007)*, Cap Esterel, French Riviera, France , 2007.
- [156] Frank Rennie, Robin Mason, *e-Learning and Social Networking Handbook*. New York: Routledge, 2008, vol. 1.
- [157] Meredith G. Farkas, *Social Software in Libraries Building Collaboration, Communication, and Community Online.*: Information Today, Inc., 2007, vol. 1.
- [158] Marc Mezquita, *Performance Characteristics of Windows Workflow Foundation.*: Microsoft Corporation, 2006.
- [159] Kevin McArthur, *Pro PHP: Patterns, Frameworks, Testing and More.*: Apress, 2008, vol. 1.
- [160] Steve Holzner, *Real World XML (2nd Edition).*: Peachpit Press, 2002, vol. 2.
- [161] Hans Gellersen, "Conference on Computer-Supported Cooperative Work," in *ECSCW 2005: Proceedings of the Ninth European*, Paris, 2005.
- [162] Hsueh-hua Chen, "Digital Library Projects in Taiwan," vol. 3, no. 1, 2006.
- [163] Nicole Ellison, Charles Steinfield, Cliff Lampe, "A face(book) in the crowd: social Searching vs. social browsing," in *the 2006 20th anniversary conference on Computer supported cooperative work*, New York, 2006.
- [164] Prasun Dewan, Vassil Roussev, *Supporting High Coupling and User-Interface Flexibility*. New York: Springer-Verlag , 2005.
- [165] Tanmoy Pal, Barnan Das, "Development of a Digital Library using DSpace Open Source Platform," Kharagpur, 2008.
- [166] Martin Fowler, *Patterns of Enterprise Application Architecture (The Addison-Wesley Signature Series).*: Addison Wesley, 2002, vol. 1.
- [167] Manoj Mansukhani. (2005) Service Oriented Architecture White Paper. PDF.
- [168] Eric C. Kansa, Jason M. Schultz, Sarah Whitcher Kansa. (2007) An Open Context

- for Near Eastern Archaeology. PDF.
- [169] Peter Thiemann, *A typed representation for HTML and XML documents in Haskell*. New York: Cambridge University Press, 2002, vol. 12.
 - [170] Mengchi Liu, Tok Wang Ling, Zhiyong Peng Shijun Li, "Automatic HTML to XML Conversion," in *Advances in Web-Age Information Management*.: Springer, 2004, vol. 3129.
 - [171] Spyros Tsipidis, Kostas Kotsakis, Alexandra Kousoulakou, Markos Katsianis, "A 3D digital workflow for archaeological intra-site research using GIS," vol. 35, no. 3, 2008.
 - [172] Lu Liu, Baosen Yang, *A Case Study of Enterprise Application Integration Based on Workflow Management System*.: Springer, 2008, vol. 1.
 - [173] Yang Yu, Grace Agnew, "The Rutgers Workflow Management System: Migrating a Digital Object Management Utility to Open Source," vol. II, no. 1, 2007.
 - [174] MSDN, "Designing Workflow Components," 2009.
 - [175] J. Jackson, N. Araujo, D. Guo, N. Gautam, Y. Simmhan, R.S. Barga, "The Trident Scientific Workflow Workbench," in *eScience, 2008. eScience '08. IEEE Fourth International Conference on*, vol. II, Redmond, WA, USA, 2008.
 - [176] Roger Barga1, Beth Plale, Nelson Araujo, Eran Chinthaka, "Workflow Evolution: Tracing Workflows Through Time," Redmond, Washington, 2009.
 - [177] Ke Liu, Ke Liu, Joel Lignier, Hai Jin, Yun Yang, "Peer-to-Peer Based Grid Workflow Runtime Environment of SwinDeW-G," in *Third IEEE International Conference on e-Science and Grid Computing (e-Science 2007)*, Bangalore, India, 2007.
 - [178] Alberto B. Raposo, Marcelo Gattass, Ismael H. F. dos Santos, "A Software Architecture for an Engineering Collaborative Problem Solving Environment," in *Software Engineering Workshop, 2008. SEW '08. 32nd Annual IEEE*, Kassandra, 2008.
 - [179] Song Ouyang, *Implementation of Policy Based Management in Workflow Management System*.: Springer, 2007, vol. 4402/2007.
 - [180] Jean Barmash, Ranju Saroch. (2007) Architecting a Knowledge-Management System. Report.
 - [181] Chappell, Associates David Chappell, "The Workflow Way," USA, 2009.
 - [182] Joe Fallon Rockford Lhotka, *Expert VB 2008 Business Objects*. New York: Apress, 2009.
 - [183] Bruce Bukoics, *Pro Windows Workflow in .NET 3.0*. New York: Apress, 2007.
 - [184] P Grefen, J Vonk M Koetsier, "Contracts for Cross-Organizational Workflow Management," in *First International Conference on Electronic Commerce and Web Technologies*, London, 2000.
 - [185] Christian Hocken, Thomas M. Deserno, Christoph Grouls, Rolf W. Günther Petra Welter, "Workflow management of content-based image retrieval for CAD support in PACS environments based on IHE," vol. 5, no. 4, 2010.
 - [186] Gerardo Canfora, Andrea De Lucia, Pierpaolo Gallucci, Lerina Aversano, "Integrating Document and Workflow Management Systems," vol. 328, 2001.
 - [187] Kwanghoon Kim, "An Enterprise Workflow Grid/P2P Architecture for Massively Parallel and Very Large Scale Workflow Systems," vol. 3842/2006, no. 1, 2006.

- [188] Ewan Fairweather, Rama Ramani, Mike Sexton, Stephen W. Thomas, Richard Seroter, *Applied Architecture Patterns on the Microsoft Platform*. Birmingham: PACKT PUBLISHING, 2007, vol. 12.
- [189] Shaohua Zhang, Johann, Schlichter, Guangwen Yang, Jinlei Jiang, "Workflow management in grid era: from process-driven paradigm to a goal-driven one," vol. I, no. 1, 2007.
- [190] William Y. Arms, "Automated Digital Libraries: How Effectively Can Computers Be Used for the Skilled Tasks of Professional Librarianship?," vol. 6, no. 1, 2000.
- [191] Sushan Dhakal, "Open Digital Library, Implementation In Open Educational Perspective," vol. III, no. 1, 2007.
- [192] Matthew Battles, *Library: An Unquiet History*.: Paw Prints, 2008, vol. 1.
- [193] Paul Miller. (2005) Web 2.0: Building the New Library. HTML.
- [194] N. Ferro, M. Agosti, "Adding Advanced Annotation Functionalities to an Existing Digital Library System," vol. II, no. 1, 2008.
- [195] Kostas Saidis, Mara Nikolaidou, Irene Lourdi George Pyrounakis, "Designing an Integrated Digital Library Framework to Support Multiple Heterogeneous Collections," vol. 3232, no. 1, 2004.
- [196] B.K. Choudhury, Y. Srinivasa Rao, "Availability of Electronic Resources at NIT Libraries in India : A Study," in *International Conference on Academic Libraries (ICAL-2009)*, vol. 1, Delhi, 2009.
- [197] Rajkumar Buyya, Mustafizur Rahman, "An Autonomic Workflow Management System for Global Grids," in *Cluster Computing and the Grid, 2008. CCGRID '08. 8th IEEE International Symposium on*, vol. 1, Lyon, 2008.
- [198] N.C. Russell, W.M.P. van der Aalst, A.J. Moleman, P.J.M. Bakker R.S. Mans, "Augmenting a workflow management system with planning facilities using colored petri nets," in *Ninth workshop and tutorial on practical use of coloured petri nets and the CPN tools*, vol. 588, Aarhus, 2008.
- [199] E.R.M. Madeira, C.B. Medeiros, W.M.P van der Aalst E. Bacarin, "SPICA's Multi-party Negotiation Protocol: Implementation using YAWL," Londrina,PR Brazil, 2009.
- [200] Todor A. Stoilov, Krasimira P. Stoilova, "Evolution of the workflow management systems," Bulgarian, 2006.
- [201] B.J.F. Steurbaut, K. Van Hoecke, S. De Turck, F. Dhoedt, B. De Smet, "Dynamic Workflow Instrumentation for Windows Workflow Foundation," in *International Conference on Software Engineering Advances, 2007. ICSEA 2007*, vol. 10.11, Cap Esterel, 2007.
- [202] C S Somu, M V Sunil, N S Harinarayana. (2009) Digital Rights Management in Digital Libraries: An Introduction to Technology, Effects and the Available Open Source Tools. PDF.
- [203] Cathy Nelson Hartman, Daniel Gelaw Alemneh. (2002) Meeting Digital Resources Preservation Challenges: University of North Texas Libraries Initiative. Power Point.
- [204] Preet Kanwal, Payare Lal, Anita Chhatwal, "Digital Heritage Archiving in India: A Case Study of Panjab University Library, Chandigarh," in *2nd International Conference The Future of Information Sciences (INFuture)*, vol. 14, Zagreb, 2009.

- [205] J. David Schloen, "Archaeological Data Models and Web Publication Using XML," vol. 35, no. 123, 2001.
- [206] Microsoft. (2006, Dec.) Italian Railway. [Online].
<http://www.google.com/url?sa=t&source=web&cd=1&ved=0CBQQFjAA&url=http%3A%2F%2Fwww.microsoft.com%2Fcasestudies%2FServeFileResource.aspx%3F1000000309&rct=j&q=Italian%2E%80%99s%20Railway%20Police%20document%20management%20system%20%5BMicrosoft%202006%5D&ei=7>
- [207] S. Esakkirajan, S. Sumathi, *Fundamentals of Relational Database Management Systems (Studies in Computational Intelligence)*. USA: Springer, 2007, vol. 35.
- [208] Paul Basu. (2007, November) Reanimating Cultural Heritage. [Online].
<http://sites.google.com/site/drpaulbasu/projects/reanimating-cultural-heritage>
- [209] Paul Basu. (2007) Reanimating Cultural Heritage. [Online].
<http://sites.google.com/site/drpaulbasu/projects/reanimating-cultural-heritage>
- [210] (2010, July) Beyond the Text Project. [Online].
<http://projects.beyondtext.ac.uk/reanimatingculturalheritage/index.php>
- [211] W.Zhang, M. Z. Patoli, M. Gkion, Al-Barakati, P. Newbury and M. White
 "Reanimating Cultural Heritage through Service Orientation, Workflows, Social Networking and Mashups," , Bradford, UK, 2009.

Appendix A

RCH New Interface

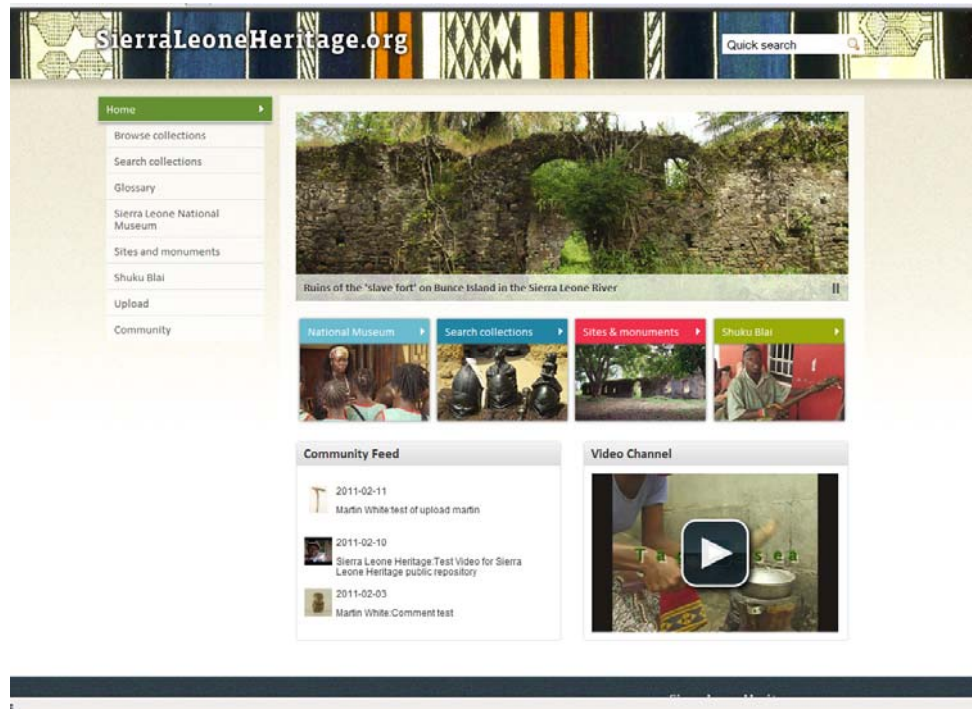


Figure A-1. The Home Page

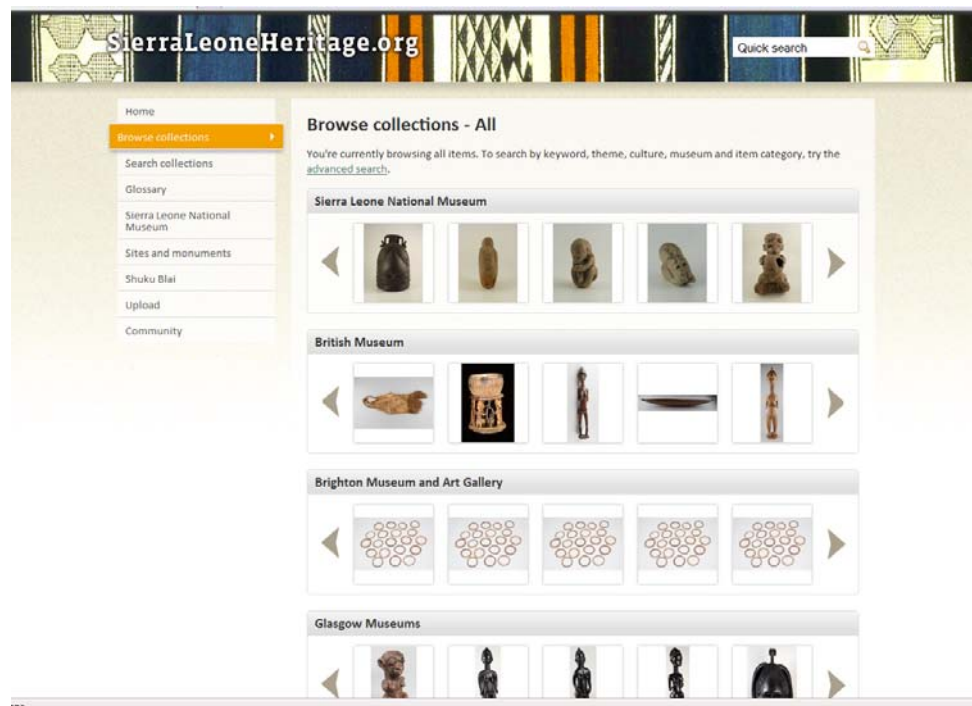


Figure A-2. The Object Browsing Page

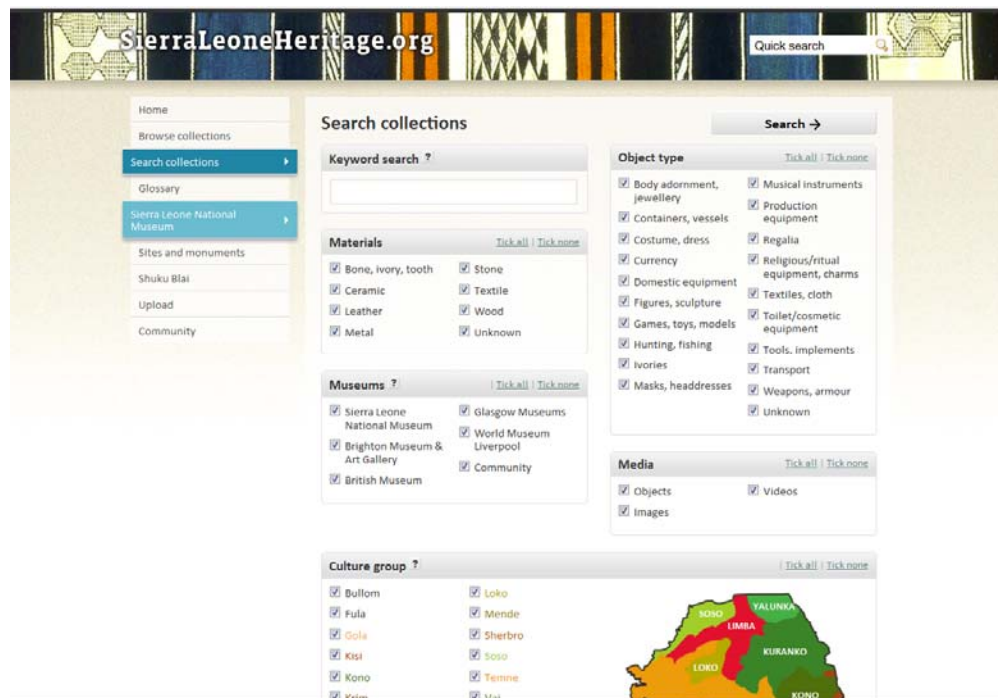


Figure A-3. The Object Retrieval Page

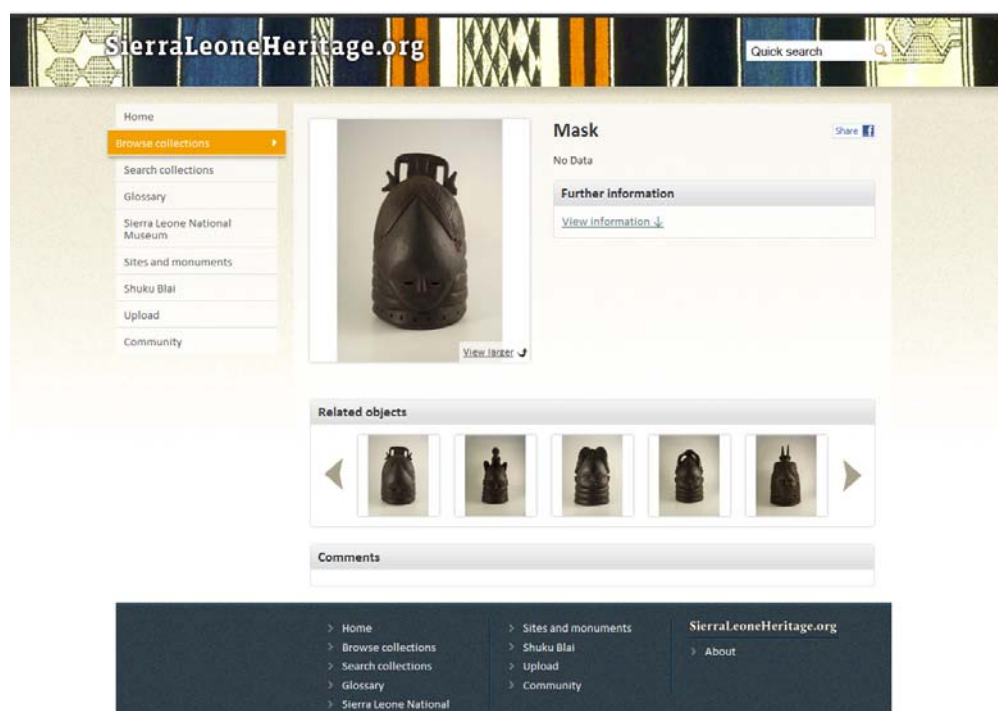


Figure A-4. Summary View of a Cultural Object

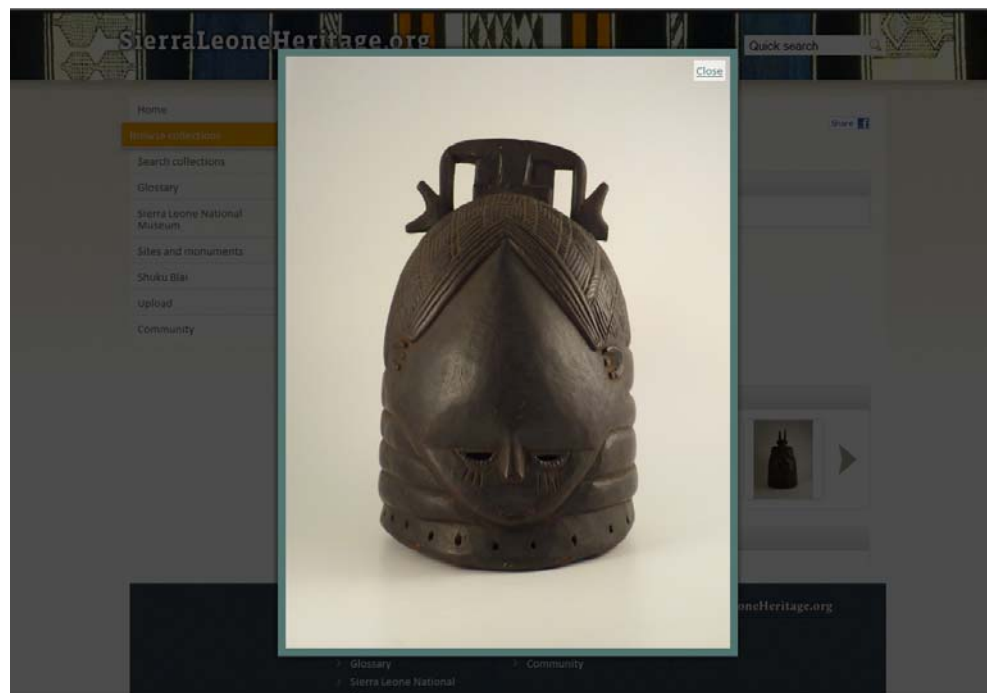


Figure A-5. Enlarged Image View

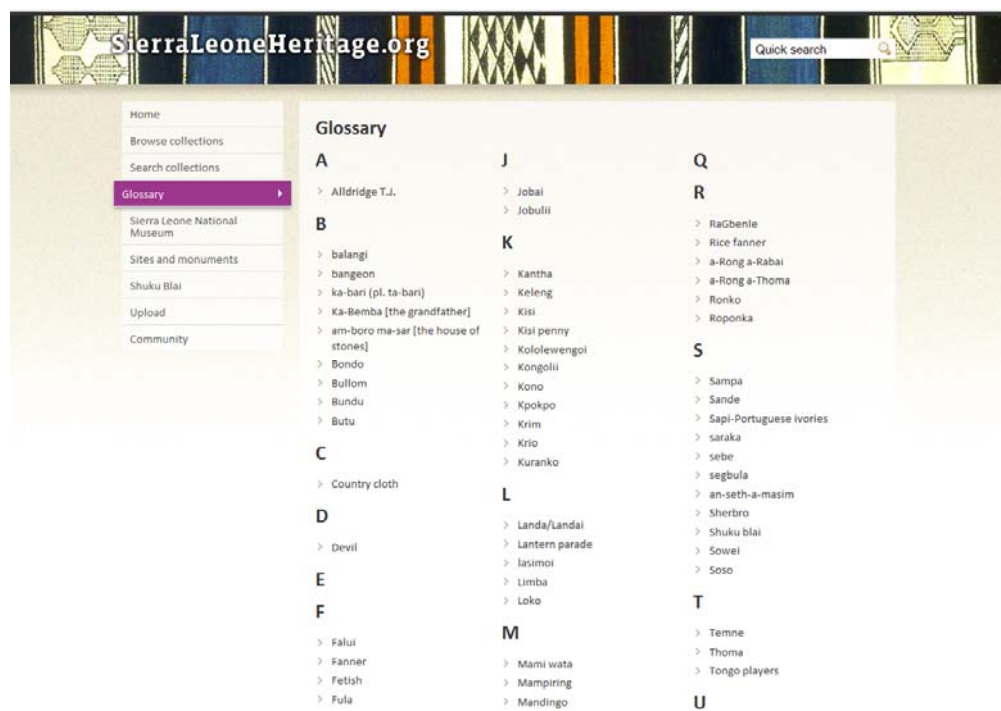


Figure A-6. Glossary

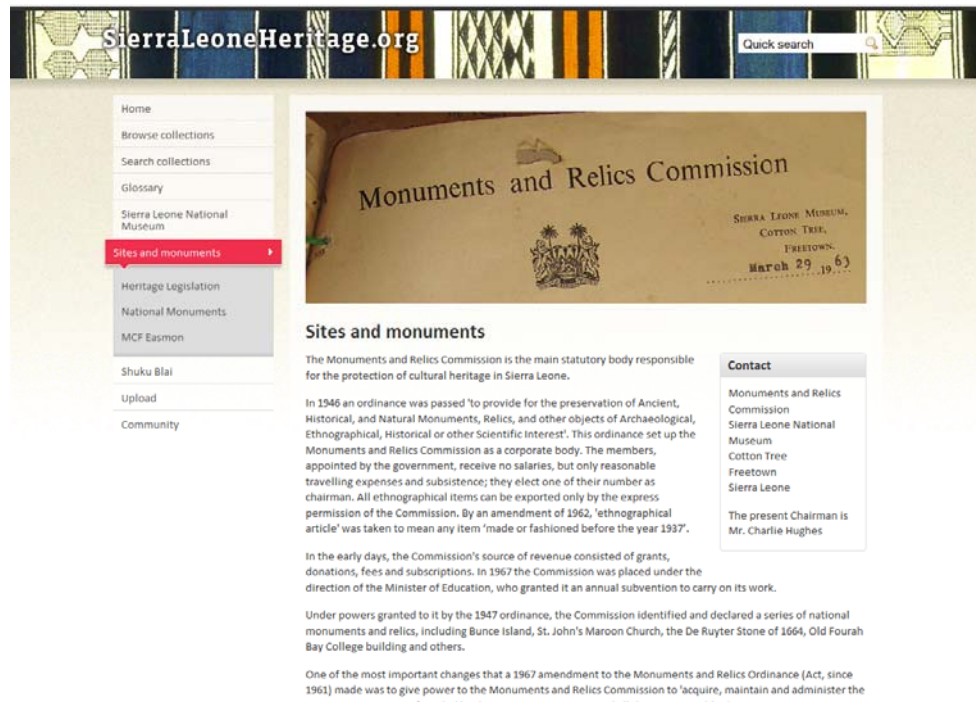


Figure A-7. Glossary Term Details



Figure A-8. Glossary Term Details

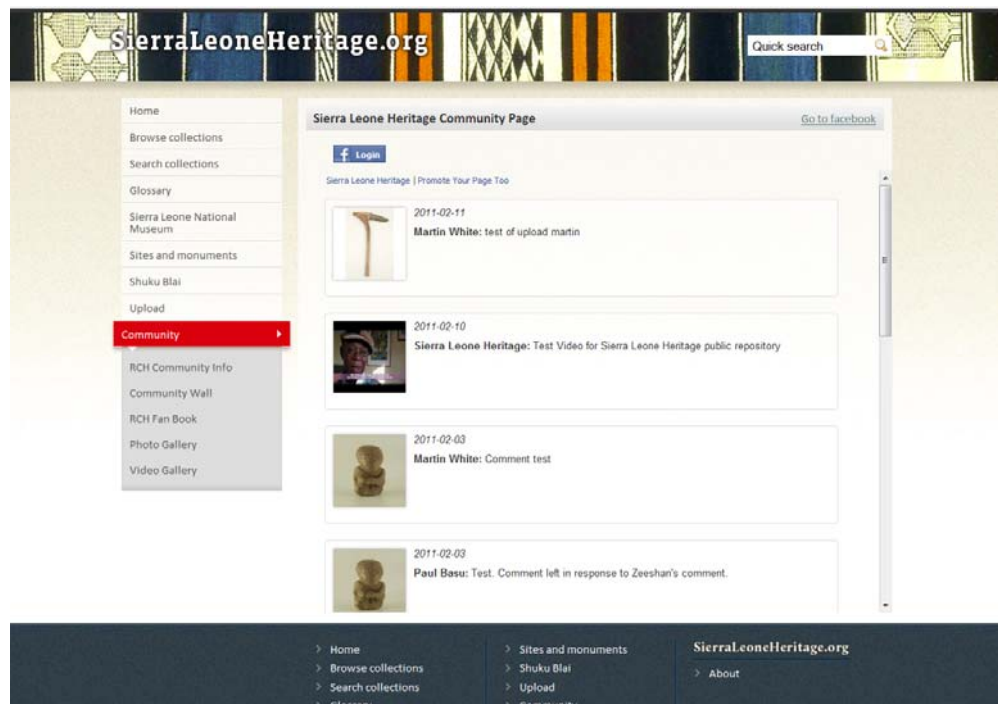


Figure A-9. Object Data Retrieval from Facebook

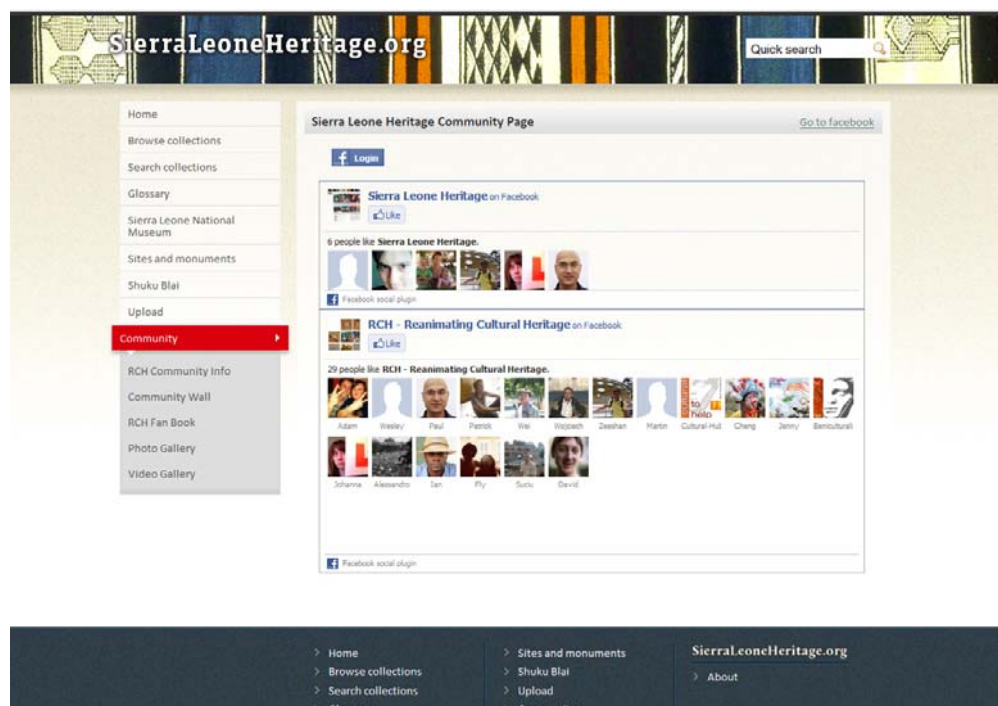


Figure A-10. Facebook Integration

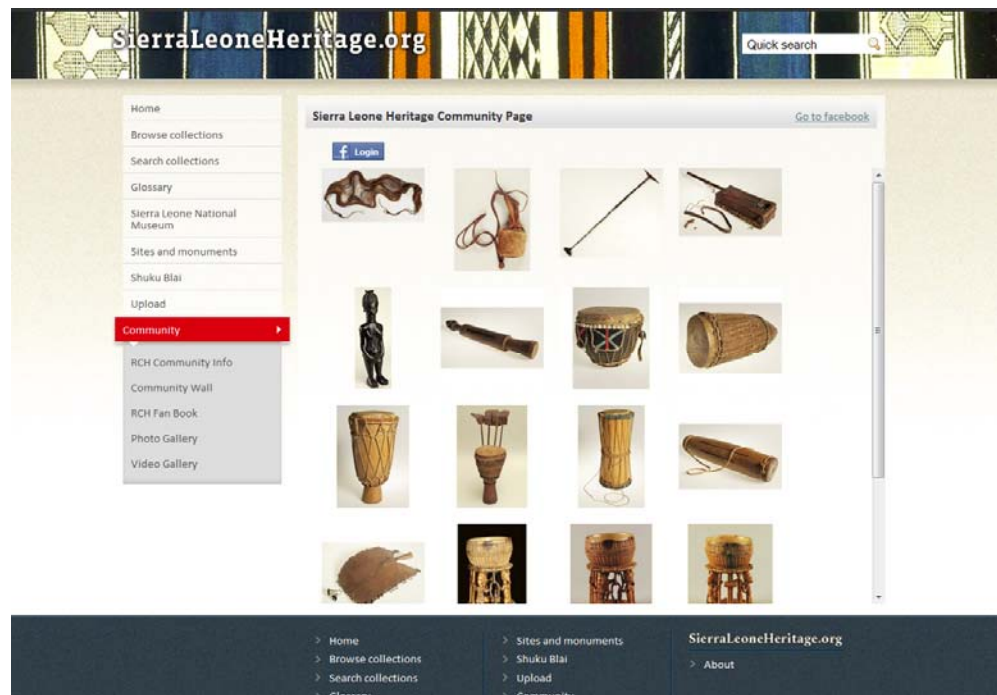


Figure A-11. Social Feeds

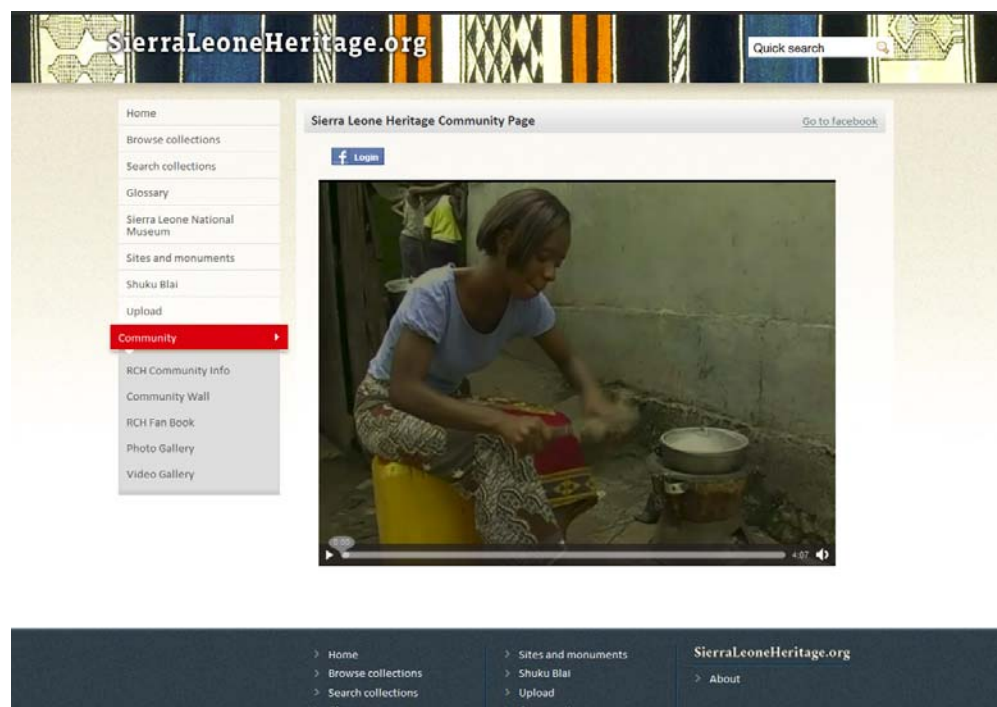


Figure A-12. Video Page

Appendix B

Museum Metadata Examples

```
<CulturalObject>
<Classifications>furniture</Classifications>
<ObjectName>hammock</ObjectName>
<Description>Hammock of dyed grass, from Sierra Leone,
West Africa.</Description>
<Materials>dyed grass</Materials>
<CultureSchool>West African</CultureSchool>
<Measurements>overall: 1151.5 g</Measurements>
<DateMade>No Data</DateMade>
<PlaceMade>Africa, West Africa, Sierra Leone (place of
manufacture)</PlaceMade>
<Maker>No Data</Maker>
<Source>Seanlan, Dr K</Source>
<Collector>No Data</Collector>
<Museum>Glasgow Museum</Museum>
<IDNumber>GLAMG:1878.137.a</IDNumber>
</CulturalObject>
```

Glasgow Museum

```
<CulturalObject>
<ObjectCategory>furniture</ObjectCategory>
<ObjectName>hammock</ObjectName>
<Description>Hammock of dyed grass, from Sierra Leone,
West Africa.</Description>
<Material>dyed grass</Material>
<EthnicName>West African</EthnicName>
<Dimensions>overall: 1151.5 g</Dimensions>
<Date>No Data</Date>
<ProductionPlace>Africa, West Africa, Sierra Leone
(place of manufacture)</ProductionPlace>
<ProducerName>No Data</ProducerName>
<AcquisitionDetails>Seanlan, Dr K</AcquisitionDetails>
<Collector>No Data</Collector>
<Museum>Glasgow Museum</Museum>
<RegistrationNumber>GLAMG:1878.137.a</RegistrationNumber>
>
</CulturalObject>
```

British Museum

Glasgow Museum's cultural heritage objects metadata, and attributes and the results of converting them to the British Museum's format.

```

<CulturalObject>
<SubCollection>furniture</SubCollection>
<ObjectName>hammock</ObjectName>
<Description>Hammock of dyed grass, from Sierra Leone,
West Africa.</Description>
<Materials>dyed grass</Materials>
<CultureGroup>West African</CultureGroup>
<Measurements>overall: 1151.5 g</Measurements>
<DateProduced>No Data</DateProduced>
<PlaceProduced>Africa, West Africa, Sierra Leone (place
of manufacture)</PlaceProduced>
<Producer>No Data</Producer>
<Source>Seanlan, Dr K</Source>
<Collector>No Data</Collector>
<Museum>Brighton Museum</Museum>
<IDNumber>GLAMG:1878.137.a</IDNumber>
</CulturalObject>

```

```

<CulturalObject>
<ObjectCategory>furniture</ObjectCategory>
<Name>hammock</Name>
<Description>Hammock of dyed grass, from Sierra Leone,
West Africa.</Description>
<Material>dyed grass</Material>
<CultureGroup>West African</CultureGroup>
<Dimension>overall: 1151.5 g</Dimension>
<ObjectProductionDate>No Data</ObjectProductionDate>
<ObjectProductionPlace>Africa, West Africa, Sierra
Leone (place of manufacture)</ObjectProductionPlace>
<Creator>No Data</Creator>
<AcquisitionSource>Seanlan, Dr K</AcquisitionSource>
<FieldCollector>No Data</FieldCollector>
<CurrentLocation>Brighton Museum</CurrentLocation>
<Source>GLAMG:1878.137.a</Source>
</CulturalObject>

```

Brighton Museum

AMS Format

Brighton Museum's cultural heritage objects metadata and attributes, and the results of converting them to the AMS format.

```

<CulturalObject>
<ObjectCategory>No Data</ObjectCategory>
<Name>sheath</Name>
<Description>Leather sheath with fringe, from Sierra
Leone, West Africa. For knife
1916.73.b.[1]</Description>
<Material>leather</Material>
<CultureGroup>No Data</CultureGroup>
<Dimension>No Data</Dimension>
<ObjectProductionDate>No Data</ObjectProductionDate>
<ObjectProductionPlace>West Africa, Sierra Leone
(place found)</ObjectProductionPlace>
<Creator>No Data</Creator>
<AcquisitionSource>Robb, James</AcquisitionSource>
<FieldCollector>No Data</FieldCollector>
<CurrentLocation>Glasgow Museum</CurrentLocation>
<Source>GLAMG:1916.73.b.[2]</Source>
</CulturalObject>

```

AMS Format

```

<CulturalObject>
<Classifications>No Data</Classifications>
<ObjectName>sheath</ObjectName>
<Description>Leather sheath with fringe, from Sierra
Leone, West Africa. For knife
1916.73.b.[1]</Description>
<Materials>leather</Materials>
<Culture/School>No Data</Culture/School>
<Measurements>No Data</Measurements>
<DateMade>No Data</DateMade>
<PlaceMade>West Africa, Sierra Leone (place
found)</PlaceMade>
<Maker>No Data</Maker>
<Source>Robb, James</Source>
<Collector>No Data</Collector>
<Museum>Glasgow Museum</Museum>
<IDNumber>GLAMG:1916.73.b.[2]</IDNumber>
</CulturalObject>

```

Glasgow

Conversion from the AMS format to the Glasgow format.


```

</CulturalObject>
<ObjectName>dagger and sheath</ObjectName>
<Description>Dagger, back of blade inlaid with brass,
in carved wooden sheath. Used by followers of Mahomet
at Sierra Leone. From collection of African
ethnological specimens.</Description>
<Materials>metal, brass, wood</Materials>
<CultureGroup>No Data</CultureGroup>
<Measurements>overall: 437 mm x 32 mm x 15 mm 263.5
g</Measurements>
<DateProduced>No Data</DateProduced>
<PlaceProduced>Africa, Equatorial Africa (place of
manufacture)</PlaceProduced>
<Producer>No Data</Producer>
<Source>Neil, Thomas and John</Source>
<Collector>No Data</Collector>
<Museum>Brighton Museum</Museum>
<IDNumber>BR:1877.18.x</IDNumber>
</CulturalObject>

```

```

<CulturalObject>
<ObjectName>dagger and sheath</ObjectName>
<Description>Dagger, back of blade inlaid with brass, in
carved wooden sheath. Used by followers of Mahomet at
Sierra Leone. From collection of African ethnological
specimens.</Description>
<Material>metal, brass, wood</Material>
<EthnicName>No Data</EthnicName>
<Dimensions>overall: 437 mm x 32 mm x 15 mm 263.5
g</Dimensions>
<Date>No Data</Date>
<ProductionPlace>Africa, Equatorial Africa (place of
manufacture)</ProductionPlace>
<ProducerName>No Data</ProducerName>
<AcquisitionDetails>Neil, Thomas and
John</AcquisitionDetails>
<Collector>No Data</Collector>
<Museum>Brighton Museum</Museum>
<RegistrationNumber>BR:1877.18.x</RegistrationNumber>
</CulturalObject>

```

Brighton

British

Brighton Museum's cultural heritage objects metadata and attributes and the results of converting them to the British Museum's format.

Appendix C

RCMS Code Samples

1. Sample controls used in the home screen.

```
'Addition of various RCMS controls including labels,
buttons, panels, etc.

<System.Diagnostics.DebuggerStepThrough()> _Private Sub
InitializeComponent()
    Me.Panel2 = New System.Windows.Forms.Panel
    Me.GroupBox1 = New System.Windows.Forms.GroupBox
    Me.Panel3 = New System.Windows.Forms.Panel
    Me.Label3 = New System.Windows.Forms.Label
    Me.ReportsToolStripMenuItem = New
System.Windows.Forms.ToolStripItem
    Me.Panel1 = New System.Windows.Forms.Panel
    Me.Button1 = New System.Windows.Forms.Button
    Me.Button2 = New System.Windows.Forms.Button
    Me.Button3 = New System.Windows.Forms.Button
    Me.StatsToolStripMenuItem = New
System.Windows.Forms.ToolStripItem
    Me.Label1 = New System.Windows.Forms.Label
    Me.ExitToolStripMenuItem = New
System.Windows.Forms.ToolStripItem
    Me.mpanel = New System.Windows.Forms.Panel
    Me.Label2 = New System.Windows.Forms.Label
    Me.MenuStrip1 = New System.Windows.Forms.MenuStrip
    Me.Label4 = New System.Windows.Forms.Label
    Me.Button4 = New System.Windows.Forms.Button
    Me.Label5 = New System.Windows.Forms.Label
    Me.Panel2.SuspendLayout()
    Me.GroupBox1.SuspendLayout()
    Me.Panel3.SuspendLayout()
    Me.Panel1.SuspendLayout()
    Me.mpanel.SuspendLayout()
    Me.MenuStrip1.SuspendLayout()
    Me.SuspendLayout()
    '
    'Panel2
    '
    Me.Panel2.BackColor =
System.Drawing.SystemColors.InactiveCaption
    Me.Panel2.Controls.Add(Me.GroupBox1)
    Me.Panel2.Location = New System.Drawing.Point(12,
573)
    Me.Panel2.Name = "Panel2"
    Me.Panel2.Size = New System.Drawing.Size(653, 115)
    Me.Panel2.TabIndex = 28
```



```

        'GroupBox1
        '
        Me.GroupBox1.Controls.Add(Me.Panel3)
        Me.GroupBox1.Location = New
System.Drawing.Point(9, 8)
        Me.GroupBox1.Name = "GroupBox1"
        Me.GroupBox1.Size = New System.Drawing.Size(621,
100)
        Me.GroupBox1.TabIndex = 0
        Me.GroupBox1.TabStop = False
        Me.GroupBox1.Text = "Workflow Monitor"
        '
        'Panel3
        '
        Me.Panel3.BackColor =
System.Drawing.SystemColors.ControlText
        Me.Panel3.Controls.Add(Me.Label3)
        Me.Panel3.Location = New System.Drawing.Point(6,
19)
        Me.Panel3.Name = "Panel3"
        Me.Panel3.Size = New System.Drawing.Size(609, 75)
        Me.Panel3.TabIndex = 1
        '
        'Button4
        '
        Me.Button4.Enabled = False
        Me.Button4.Font = New
System.Drawing.Font("Microsoft Sans Serif", 16.0!,
System.Drawing.FontStyle.Bold,
System.Drawing.GraphicsUnit.Point, CType(0, Byte))
        Me.Button4.Location = New System.Drawing.Point(24,
12)
        Me.Button4.Name = "Button4"
        Me.Button4.Size = New System.Drawing.Size(159, 66)
        Me.Button4.TabIndex = 4
        Me.Button4.Text = "Home"
        Me.Button4.UseVisualStyleBackColor = True

```

2. Some Workflow Handlers.

```

Shared Sub Main()
    Using workflowRuntime As New WorkflowRuntime()
        AddHandler
workflowRuntime.WorkflowCompleted, AddressOf
OnWorkflowCompleted
        AddHandler
workflowRuntime.WorkflowTerminated, AddressOf
OnWorkflowTerminated
        Dim workflowInstance As WorkflowInstance
        workflowInstance =
workflowRuntime.CreateWorkflow(GetType(retrieval_workflow)
)
        workflowInstance.Start()
        WaitHandle.WaitOne()
    End Using
End Sub

Shared Sub OnWorkflowCompleted(ByVal sender As
Object, ByVal e As WorkflowCompletedEventArgs)
    WaitHandle.Set()
End Sub

Shared Sub OnWorkflowTerminated(ByVal sender As
Object, ByVal e As WorkflowTerminatedEventArgs)
    Console.WriteLine(e.Exception.Message)
    WaitHandle.Set()
End Sub

```

Appendix D

RCH Website Code Samples

1. Object Browser Page Code Sample

1.1. JavaScript function calls within HTML and PHP

Displaying contents of a specific tribe by calling various JavaScript function embedded within the XHTML and PHP code blocks of the page.

```

<!-------/ Cultural Links ----->

<div id=brows-outer>
<div class="links large0" >

<a href =
"javascript:displayResult('Mende'),displayResult2('Mende'
),displayResult3('Mende')">
    <div class="style2 link1" >
        <div align="center" id="style3">
            <h5>Mende </h5>
        </div>
    </div>
</a>

<a href =
"javascript:displayResult('Susu'),displayResult2('Susu'),
displayResult3('Susu')">
    <div class="style2 link2">
        <div align="center" id="style3">
            <h5> Susu </h5>
        </div>
    </div>
</a>

<a href =
"javascript:displayResult('Temne'),displayResult2('Temne'
),displayResult3('Temne')">
    <div class="style2 link3">
        <div align="center" id="style3">
            <h5> Temne </h5></div>
    </div>
</a>
.
.
.

```

1. 2. Content Presentation Blocks

Example block used to display various page contents.

```

<!--/ Display British Museum Images Here ----->

<div class="title_box1 title">
  <div id="title_text" >
    <h3>British Museum</h3>
  </div>
</div>

<div class="British-objects large" >
<div id="dom1" style="display:none;"></div>
<div id="dom2" style="display:none;"></div>
<div id="dom3" style="display:none;"></div>
<div id="dom4" style="display:none;"></div>
<div id="dom5" style="display:none;"></div>
<div id="dom6" style="display:none;"></div>
<div id="dom7" style="display:none;"></div>
<div id="dom8" style="display:none;"></div>
<div id="example2" class="yui-skin-sam"></div>
</div>

<!-------/ Display Glasgow Museum Images Here ----->
<div class="title_box2 title">
  <div id="title_text" >
    <h3>Glasgow Museum </h3>
  </div>
</div>
<div class="glasgowmuseum large">
<div id="dom1" style="display:none;"></div>
<div id="dom2" style="display:none;"></div>
<div id="dom3" style="display:none;"></div>
<div id="dom4" style="display:none;"></div>
<div id="dom5" style="display:none;"></div>
<div id="dom6" style="display:none;"></div>
<div id="dom7" style="display:none;"></div>
<div id="dom8" style="display:none;"></div>
<div id="example1" class="yui-skin-sam"></div>
</div>
.
.
.

```

2. Example XSLT Snippets

3.1. Parameter Passing from PHP pages

Catching and reading the parameters that are coming from the PHP search frontend.

```
<!--Variables coming through from search.php -->
<xsl:param name="title"/>
<xsl:param name="cult"/>
<xsl:param name="cat"/>
<xsl:template match="/">
```

3.2. XSLT Parameter Definitions

Search parameter definition within the main retrieval XSLT file.

```
<!--The main search variables-->
<!--This is to get the image-->
<xsl:variable name="link"
select='concat(MediaObjects/MediaObject1/MediaFileName,"")'
'/>
<!--This is to get the object description to search
within-->
<xsl:variable name="te" select='concat(Description,"")' />
<!--This is to get the object name to display in the
search results-->
<xsl:variable name="n1" select='concat(Object,"")' />
<!--This is to trim the object name for display purposes-->
<xsl:variable name="name"
select='concat(substring($n1,0,8),"...")' />
<!--This is to create the link to display the object
details; the object ID will determine which object to
display-->
<xsl:variable name="AccNumb"
select='concat(AccessionNumber,"",object="")' />

<!--Variables below to create the link to the object to
open in a new page-->
<xsl:variable name="final" select='concat($AccNumb,$n1)' />
<xsl:variable name="olink"
select='concat("Browse_Results.php?id=", $final)' />
<xsl:variable name="ID"
select='concat(AccessionNumber,"")' />
<!--To determine the museum, as the first few characters
in the ID determine the museum-->
<xsl:variable name="selector"
select='concat(substring($ID,0,4),"")' />
<!--This is the link when clicking on the object image-->
<xsl:variable name="object"
select='concat("object.php?id=",AccessionNumber)' />
.
.
```

3.3. Retrieval Results Presentation

The following XSLT code snippet shows how the search results are rendered as HTML code.

```
<!--Check if the selected cultural group matches the one
in the object, this is passed from search.php -->
<!--The same applies to category and title-->
<xsl:if test="contains($cult,$culte)">
  <xsl:if test="contains($cat,$cate)">
    <xsl:if test="contains($te,$title)">
      <!--Output the result-->
      <div class="object-result clearfix"
style="position:relative;left:20; width:800;border-
bottom:#CCC 1px solid;PADDING-top: 8px;">

        <h2 id="resultsTitle" class="clearfix">
          <span class="large"></span>
        </h2>

        <div style="width:85px; height:105px;
float:left;">

          <xsl:choose>

            <xsl:when test="contains($link,'No Data')">
              <A HREF="{ $object}" >
                 </img>
              </A>
            </xsl:when>

            <xsl:when test="contains($selector,'BM:')">
              <A HREF="{ $object}" >
                 </img>
              </A>
            </xsl:when>

            <xsl:when test="contains($selector,'BMA')">

              <A HREF="{ $object}" >
                 </img>
              </A>
            </xsl:when>

            .
            .
            .
```

3.4. Online Mapping Tool

3.4.1. A function to upload an XML file to the desired location

```
Protected Sub Upload_This_File(ByVal upload As FileUpload)
    'If upload.HasFile Then
        fileName = upload.FileName
        ext = fileName.Substring(fileName.LastIndexOf("."))
        xmlFileName = fileName.Substring(0,
fileName.LastIndexOf(".")) & ".xml"
        'This is where the upload location is specified
        Dim theFileName As String =
Path.Combine(Server.MapPath("~/App_Data"), upload.FileName)
        upload.SaveAs(Server.MapPath(xml_name))
    End Sub
```

3.4.2. A function to produce the final mapped XML file

```
Function Make_XML()
    'Perform mapping by calling the mapping function
    map_mim()

    'Create DB/XML connections
    Dim myConnection As New
System.Data.SqlClient.SqlConnection
    myConnection.ConnectionString =
ConfigurationManager.ConnectionStrings("conString").Connection
String

    'SQL Commands for reading/writing operations
    'Data goes to DB in this version for filtering and then gets
    Dim cmdXML As System.Data.SqlClient.SqlCommand
    Dim DScmdXML As System.Data.SqlClient.SqlDataAdapter
    Dim DSXML As New System.Data.DataSet()
    Dim sqlXML = "SELECT * FROM ObjectData"
    cmdXML = New System.Data.SqlClient.SqlCommand(sqlXML,
myConnection)
    DScmdXML = New
System.Data.SqlClient.SqlDataAdapter(cmdXML)
    DScmdXML.Fill(DSXML, "ObjectData")
    Response.Flush()

    'Write the results to the XML file
    DSXML.WriteXml(tabelname.ToString & ".xml",
XmlWriteMode.WriteSchema)
    Dim xmlSW2 As System.IO.StreamWriter = New
System.IO.StreamWriter(Server.MapPath("xml/" &
tabelname.ToString & ".xml"))

    DSXML.WriteXml(xmlSW2, XmlWriteMode.IgnoreSchema)
    xmlSW2.Flush()
    xmlSW2.Close()
End Function
```

3.4.3. A function to determine the required mapping type

```
'Get Mapping Type
'Analyse the gathered data in the DB
    Dim myconnection As New
System.Data.OleDb.OleDbConnection
    myConnection.ConnectionString =
"Provider=Microsoft.Jet.OLEDB.4.0; Data Source=" & dbname &
" "
    myConnection.Open()
    Dim delTable As New
'Read data
System.Data.OleDb.OleDbCommand()
    delTable.Connection = myconnection
    delTable.CommandType = CommandType.Text
    delTable.CommandText = "select mType from type
where ID=1"
    dim type = delTable.ExecuteScalar
    myConnection.Close()
```

3.4.4. A function to handle Excel files in case data is coming from them

```
Dim FileName As String = lblFileName.Text
    Dim Extension As String =
Path.GetExtension(FileName)
    Dim FolderPath As String = Server.MapPath( _
    ConfigurationManager.AppSettings("FolderPath"))
    Dim CommandText As String = ""
    Select Case Extension
        Case ".xls"
            CommandText = "px_ImportFromExcel"
            Exit Select
        Case ".xlsx"
            'Excel 07
            CommandText = "px_ImportFromExcel07"
            Exit Select
    End Select
'Read Excel Sheet using Stored Procedure
'and import the data into Database Table
Dim strConnString As String = ConfigurationManager
-
    .ConnectionStrings("conString").ConnectionString
    Dim con As New SqlConnection(strConnString)
    Dim cmd As New SqlCommand()
    cmd.CommandType = CommandType.StoredProcedure
    cmd.CommandText = CommandText
    cmd.Parameters.Add("@SheetName",
SqlDbType.VarChar).Value = ddlSheets.SelectedItem.Text
    cmd.Parameters.Add("@FilePath",
SqlDbType.VarChar).Value = FolderPath + FileName
    cmd.Parameters.Add("@HDR", SqlDbType.VarChar).Value
= 1
    cmd.Parameters.Add("@TableName",
SqlDbType.VarChar).Value = "ObjectData"
```