A DSPACE MOBILE THEME FOR SAN DIEGO STATE UNIVERSITY

A Thesis
Presented to the
Faculty of
San Diego State University

In Partial Fulfillment
of the Requirements for the Degree
Master of Science
in
Computer Science

by
Mini Vamadevan Pillai

Spring 2014
SAN DIEGO STATE UNIVERSITY

The Undersigned Faculty Committee Approves the
Thesis of Mini Vamadevan Pillai:

A DSpace Mobile Theme for San Diego State University

Carl Eckberg
Carl Eckberg, Chair
Department of Computer Science

Alan Riggins
Department of Computer Science

Mark Jean Gawron
Department of Linguistics & Asian/Middle Eastern Languages

Nov 13, 2013
Approval Date
DEDICATION

I would like to dedicate this thesis to my family, especially my husband Abhay Vijayan, my parents Vamadevan Pillai and Rathee Devi and my brother Manoj Pillai for their valuable guidance and support during my entire graduation work. I would also like to dedicate this thesis to my professors for their constant encouragement and support. Last but not the least this thesis is dedicated to my loving daughter Shreya.
ABSTRACT OF THE THESIS

A DSpace Mobile Theme for San Diego State University

by

Mini Vamadevan Pillai
Master of Science in Computer Science
San Diego State University, 2014

DSpace is the most widely used open source repository platform and is popular in different fields like academic, research center, health, government, museums, national libraries and so on. More than hundred countries and over 1500 institutions now use DSpace. San Diego State University’s digital collection of Theses and Dissertations runs on DSpace. This thesis is an attempt to install and customize a DSpace mobile theme for San Diego State University. The work also includes development of additional features like adding navigational bars, adding administrative login capabilities, accessing administrative navigational panel via mobile theme. The mobile theme supports other features like search, advanced search, recent submissions, submissions and workflow. With the widespread use of mobile telephony, providing a mobile theme for SDSU DSpace will reach out to faculty and other interested parties.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABSTRACT</td>
<td>v</td>
</tr>
<tr>
<td>LIST OF TABLES</td>
<td>ix</td>
</tr>
<tr>
<td>LIST OF FIGURES</td>
<td>x</td>
</tr>
<tr>
<td>LIST OF ABBREVIATIONS</td>
<td>xii</td>
</tr>
<tr>
<td>ACKNOWLEDGMENTS</td>
<td>xiii</td>
</tr>
<tr>
<td>CHAPTER 1 INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>1.1 Why Use DSpace?</td>
<td>1</td>
</tr>
<tr>
<td>1.2 SDSU and DSpace</td>
<td>2</td>
</tr>
<tr>
<td>1.3 Why a Mobile Theme?</td>
<td>2</td>
</tr>
<tr>
<td>CHAPTER 2 DSPACE SYSTEM OVERVIEW</td>
<td>5</td>
</tr>
<tr>
<td>2.1 DSpace Architecture</td>
<td>5</td>
</tr>
<tr>
<td>2.1.1 Application Layer</td>
<td>6</td>
</tr>
<tr>
<td>2.1.2 Business Logic Layer</td>
<td>7</td>
</tr>
<tr>
<td>2.1.3 Storage Layer</td>
<td>8</td>
</tr>
<tr>
<td>2.2 Data Model</td>
<td>9</td>
</tr>
<tr>
<td>2.3 Metadata</td>
<td>11</td>
</tr>
<tr>
<td>2.4 Handles</td>
<td>11</td>
</tr>
<tr>
<td>CHAPTER 3 DSPACE INSTALLATION</td>
<td>13</td>
</tr>
<tr>
<td>3.1 Dspace Pre-Requisites</td>
<td>13</td>
</tr>
<tr>
<td>3.1.1 Apache Maven 2.2.x or Later</td>
<td>13</td>
</tr>
<tr>
<td>3.1.2 APACHE Ant 1.8 or Later</td>
<td>13</td>
</tr>
<tr>
<td>3.1.3 Relational Database</td>
<td>13</td>
</tr>
<tr>
<td>3.1.4 Servlet Engine</td>
<td>14</td>
</tr>
<tr>
<td>3.2 DSpace Code</td>
<td>14</td>
</tr>
<tr>
<td>3.2.1 DSpace Source Structure</td>
<td>14</td>
</tr>
<tr>
<td>3.2.2 DSpace Binary Release Structure</td>
<td>15</td>
</tr>
</tbody>
</table>
3.3 Installation Process ..........................................................................................15
  3.3.1 Creating a Database in Postgres ..............................................................15
  3.3.2 Maven Build Process ..............................................................................16
  3.3.3 Deployment Process (Ant)......................................................................17
  3.3.4 Deployment Process (Tomcat).................................................................18

4  MANAKIN ARCHITECTURE ...................................................................................19
  4.1 Introduction ......................................................................................................19
  4.2 The Cocoon Sitemap........................................................................................19
    4.2.1 Cocoon Sitemap Structure ......................................................................20
    4.2.2 Components ............................................................................................20
    4.2.3 Generators ...............................................................................................21
    4.2.4 Transformers ...........................................................................................21
    4.2.5 Serializers ................................................................................................21
    4.2.6 Matchers ..................................................................................................22
  4.3 Three Components ...........................................................................................22
    4.3.1 DRI Schema ............................................................................................22
    4.3.2 Aspects ....................................................................................................26
    4.3.3 Themes ....................................................................................................27

5  IMPLEMENTATION OF DSPACE ON MOBILE FOR SDSU ................................31
  5.1 SDSU DSpace Database ..................................................................................31
  5.2 Mobile Theme File Structure ...........................................................................32
  5.3 Themes.xmap ..................................................................................................33
  5.4 CSUGeneralTheme.xsl ....................................................................................34
  5.5 Mobile Sitemap.xmap ......................................................................................35
  5.6 Mobile.xsl ........................................................................................................36
    5.6.1 The Header ..............................................................................................37
    5.6.2 The Body .................................................................................................38
    5.6.3 The Footer ...............................................................................................39
  5.7 Password Login ................................................................................................40
  5.8 Admin Panel.....................................................................................................40
    5.8.1 Create Community ..................................................................................41
    5.8.2 EPeople ...................................................................................................42
LIST OF TABLES

PAGE

Table 2.1. MIT Libraries' Definitions of Bitstream Format Support Levels .........................11
Table 2.2. Example of Data Model in SDSU DSpace ..........................................................12
LIST OF FIGURES

Figure 1.1. Map view of repositories used worldwide..........................................................2
Figure 1.2. Screenshot of SDSU Library and Information Access website showing
links to Thesis and Dissertations.....................................................................................3
Figure 1.3. Screenshot of DSpace instance for SDSU..........................................................3
Figure 1.4. Screenshot of SDSU DSpace instance accessed via iPhone...............................4
Figure 2.1. DSpace System Architecture.............................................................................6
Figure 2.2. Server architecture.............................................................................................9
Figure 2.3. Pictorial representations of data hierarchy and a comparison with SDSU
data hierarchy..................................................................................................................10
Figure 3.1. DSpace source structure....................................................................................14
Figure 3.2. DSpace binary release structure.......................................................................15
Figure 3.3. DSpace config directory structure.....................................................................16
Figure 3.4. Screenshot of IntelliJ showing maven settings...............................................17
Figure 3.5. DSpace deployment directory structure............................................................18
Figure 4.1. Cocoon pipeline processing................................................................................19
Figure 4.2. Cocoon pipeline processing showing sitemap.................................................20
Figure 4.3. Cocoon pipeline processing showing aspect chain........................................23
Figure 4.4. Structure of DRI...............................................................................................23
Figure 4.5. Screenshot of SDSU DSpace body with different containers.........................25
Figure 4.6. Screenshot of SDSU DRI options (left) and the corresponding navigation
section in browser (right)..................................................................................................26
Figure 4.7. Screenshot from IntelliJ of aspects residing in dspace-xmlui-api library.........28
Figure 4.8. Pictorial representation of cocoon pipeline processing....................................28
Figure 4.9. Pictorial representation of content generation and styling..............................29
Figure 5.1. Screenshot of home page top and bottom part on a mobile..............................31
Figure 5.2. Screenshot of PgAdminIII of the database used.............................................32
Figure 5.3. Screenshot of PgAdminIII of EPerson table used............................................32
Figure 5.4. Mobile theme file structure (left) and screenshot of file structure from IntelliJ (right). ..............................................................................................................33

Figure 5.5. Screenshot of DSpace on mobile to show the navigation bar before and after logging in. ...........................................................................................................38

Figure 5.6. Screenshot of password login page and logged in page. .......................................................40

Figure 5.7. Admin panel top page and admin panel bottom page. .........................................................41

Figure 5.8. Screenshot of create community page. .................................................................................41

Figure 5.9. Screenshot of EPeople list and EPerson management. ............................................................42

Figure 5.10. Screenshot of authorizations screen. ..................................................................................43

Figure 5.11. Screenshot of metadata listing when admin clicks on “Metadata” link in admin panel. ...........................................................................................................................43

Figure 5.12. Screenshot of format screen when admin clicks on “Format” link in admin panel. .................................................................................................................................44

Figure 5.13. Screenshot of the statistics navigation and usage statistics. .............................................45

Figure 5.14. Screenshot of unfinished items and workflow tasks seen on clicking the “Submissions” link in home page of mobile theme. .........................................................45

Figure 5.15. Screenshot of actions that the administrator can perform after selecting a task from the pool. .........................................................................................................................46

Figure 5.16. Screenshot of search for “catalog” and the search results produced. ..............................47

Figure 5.17. Screenshot of advanced search filter and sort options. .....................................................47

Figure 5.18. Screenshot of a filter applied and the resulting search result. ...........................................48

Figure 5.19. Screenshot of recent submission in home page (left) and item view page (right). .................................................................................................................................50

Figure 5.20. Screenshot of Item view and bitstream attached. .................................................................51
# LIST OF ABBREVIATIONS

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AOP</td>
<td>Aspect Oriented Programming</td>
</tr>
<tr>
<td>API</td>
<td>Application Programming Interface</td>
</tr>
<tr>
<td>CSS</td>
<td>Cascading Style Sheet</td>
</tr>
<tr>
<td>DC</td>
<td>Dublin Core</td>
</tr>
<tr>
<td>DIM</td>
<td>DSpace Intermediate Metadata Format</td>
</tr>
<tr>
<td>DRI</td>
<td>Digital Repository Interface</td>
</tr>
<tr>
<td>GB</td>
<td>Giga Byte</td>
</tr>
<tr>
<td>HTTP</td>
<td>Hyper Text Transfer Protocol</td>
</tr>
<tr>
<td>JSP</td>
<td>Java Server Pages</td>
</tr>
<tr>
<td>METS</td>
<td>Metadata Encoding and Transmission Standard</td>
</tr>
<tr>
<td>MODS</td>
<td>Metadata Object Description Schema</td>
</tr>
<tr>
<td>OAI</td>
<td>Open Archives Initiative</td>
</tr>
<tr>
<td>PC</td>
<td>Personal Computer</td>
</tr>
<tr>
<td>PDF</td>
<td>Portable Document Format</td>
</tr>
<tr>
<td>RFC</td>
<td>Request for Comments</td>
</tr>
<tr>
<td>RDBMS</td>
<td>Relational Database Management System</td>
</tr>
<tr>
<td>SAX</td>
<td>Simple API for XML</td>
</tr>
<tr>
<td>TEI</td>
<td>Text Encoding Initiative</td>
</tr>
</tbody>
</table>
ACKNOWLEDGMENTS

I would like to thank my advisor, Dr. Carl Eckberg, for his assistance and guidance throughout my thesis work. I would also like to extend my thanks to Keven Jeffery (Digital Technologies Librarian at San Diego State University) for his valuable inputs, guidance and support throughout the entire thesis. I am also grateful to Prof. Alan Riggins and Dr. Jean Mark Gawron for providing additional advice and serving on this committee. In addition, I extend my sincere appreciation to the graduate students, staff and the entire faculty of the Department of Computer Science. Finally, I express my heartfelt gratitude to my parents, family and friends for their moral support, love and unfailing encouragement.
CHAPTER 1

INTRODUCTION

DSpace is an open source digital repository. It helps to manage research materials. With the increase in digital content organizing and managing the digital content for an institution was crucial. Massachusetts Institute of Technology (MIT) and Hewlett Packard (HP) developed DSpace and was first releases in November 4, 2002. More than 1500 organizations over 100 countries not use DSpace. DSpace provides a solution to manage, provide access to, preserve and share digital assets. Different file formats can be stored in DSpace- text, video, audio and data. The main advantage of DSpace is it is customizable and can be customized based on the needs of an organization. DSpace indexed the work so that users can search and retrieve an item. Long term preservation is another important feature too.

1.1 WHY USE DSPACE?

Though there are other digital repositories like Fedora, CONTENTdm, and Digital Commons. DSpace is being used widely across the country. The main reason is that DSpace is open source which has front end web based User Interface layer and has a customizable user interface. DSpace supports features like Search, Authentication, built in submission and workflows, flexible role based permission, several metadata schemes and version control.

Repositories like CONTENTdm and Digital Commons are not open source and it comes with a price for a customizable user interface. Fedora does not have a built in submission or workflows and version control feature. In addition Fedora cannot come with a customizable user interface and front end User Interface layer.

Figure 1.1 [1] shows the map view of the repositories used worldwide. DSpace stands as the most widely used repositories with a usage of 1126 institutions [1]. The data is registered between January 1990 and December 2013.
1.2 SDSU AND DSPACE

The digital collections in San Diego State University are stored currently in DSpace. The Library and Information Access in SDSU manages the DSpace. A user can find the SDSU digital collections here: http://library.sdsu.edu/find/digital-collections. Figure 1.2 shows the screenshot of SDSU Library and Information Access website showing links to Thesis and Dissertations which are stored in DSpace. These links redirect to http://sdsu-dspace.calstate.edu/ which is the DSpace instance for SDSU. The SDSU Syllabus collections are also stored in DSpace. Figure 1.3. shows the screenshot of DSpace instance for SDSU.

1.3 WHY A MOBILE THEME?

The current DSpace instance when accessed on a mobile device has many issues. The main problem is it is difficult to pan around, poor readability and overall a bad user experience. Figure 1.4 shows the screenshot of DSpace instance when accessed via iPhone. Only a part of the website is seen on the screen and the user has to pan around to view more.
Figure 1.2. Screenshot of SDSU Library and Information Access website showing links to Thesis and Dissertations.

Figure 1.3. Screenshot of DSpace instance for SDSU.
In order to create a mobile friendly version of DSpace there were two approaches. One was to create an app for mobile. In that case an app for every mobile platform iOS, Android, Symbian, Java was required. But there are many users with low end phones and so a better solution was to use something which will be mobile optimized and works on low and high end phones. Hence jQuery mobile (jQM) framework was used. The use of HTML5 standards in jQM allows developers and web designers to create a single robust and high branded website that work on all popular smartphones, tablets and desktop platforms [2].
CHAPTER 2

DSpace System Overview

DSpace is open source software platform which can be used to maintain an assetstore and an associated metadata store. They preserve the digital assets for long run. The web user interface provides interfaces for administration, search, deposit and ingest. The disk or file store maintains the assetstore where bitstream files are located while the database stores all other information. This includes information about organization content, metadata, e-people information, authorizations, state of currently running workflows, metadata registry, formats etc.

2.1 DSpace Architecture

DSpace Architecture consists of various layers namely: Application layer, Business logic layer and Storage layer. Each layer invokes the layer beneath it. An important point in DSpace architecture is that each layer is trusted. Each layer has API and allows for future customizations and enhancements [3]. The application layer provides user interface. It supports Web User Interface, OAI which is the Open Archives Initiative, batch loader etc.

Much advancement has been added to this layer recently. The DSpace logic lies in the second layer which is the business logic layer. This includes content management, search, workflow, authorizations, administrative etc. Figure 2.1 [4] shows the DSpace System Architecture.

The source code is organized as follows. The left hand side denotes the package.

- org.dspace.app : Application Layer
- org.dspace    : Business Logic Layer
- org.dspace.storage : Storage Layer
In the following section we will discuss each layer in detail.

### 2.1.1 Application Layer

Application layer consists of components that communicate to end users. It includes XML and JSP Web User Interface and Open Archives Initiatives protocol which is used for metadata harvesting service. DSpace Web UI is the most used component. XMLUI is built on XML and Cocoon technology while JSPUI is built on java servlet and Java server page technology. JSPUI is based on Model View Controller model. The content management API is the model, Java servlets are the controllers and JSPs are the views. When a HTTP request is received the appropriate servlet is invoked and DSpace business logic public API is invoked too and processed. Finally the JSP is invoked and is processed and sent to browser. Application layer calls the layer below it which is the Business Logic layer.
2.1.2 Business Logic Layer

This layer performs the business logic. Business Logic Layer consists of different modules like the Authorization, Search, History, Content Management, Browse tools etc. and each module has “public” API. These APIs collectively referred to as DSpace public API.

- **Content Management**: Content Management API is in org.dspace.content and it consists of Java classes for reading and changing content stored in DSpace. This is the most used component.

- **Core classes**: The package org.dspace.core consists of core classes that are used throughout DSpace. The dspace.cfg file is the main configuration file. The contents from this file are read by the Configuration Manager. The Constants class is used to represent the DSpace types and DSpace actions in database. This resides in org.dspace.constant package. DSpace types are:
  - Bitstream: 0
  - Bundle: 1
  - Item: 2
  - Collection: 3
  - Community: 4
  - Site: 5
  - Group: 6
  - Eperson: 7

DSpace actions are:

- Read: 0
- Write: 1
- Delete: 2
- Add: 3
- Remove: 4

- **Administration Toolkit**: The package org.dspace.administer consists of classes that handle administrative logic. An administrator can be created from command line after installing a DSpace instance. This is done by going to deployment directory bin folder and running the command dspace create-administrator. This invokes the class CreateAdministrator and creates an admin based on the information entered by the user in standard input.

- **EPerson/Groups**: The package org.dspace.eperson consists of class EPerson which creates and edit an EPerson in DSpace. A list of EPerson is called Groups.
Group names must be unique. It is necessary to call the update() function when modifying EPerson or Group content else the modifications are lost.

- **Authorizations**: The package org.dspace.authorize does all authorization related tasks. For example AuthorizeManager class does authorization, checks policies against groups. ResourcePolicy class defines all allowable actions for an object.

- **Search**: The DSpace search feature is one of the most widely used feature. Currently the DSpace search API wraps Lucene search engine. The fields that are to be indexed are defined in the dspace.cfg file. In the case of SDSU the following fields are indexed.

  - `search.index.1 = author:dc.contributor.*`
  - `search.index.2 = author:dc.creator.*`
  - `search.index.3 = title:dc.title.*`
  - `search.index.4 = keyword:dc.subject.*`
  - `search.index.5 = abstract:dc.description.abstract`
  - `search.index.6 = author:dc.description.statementofresponsibility`
  - `search.index.7 = series:dc.relation.ispartofseries`
  - `search.index.8 = abstract:dc.description.tableofcontents`
  - `search.index.9 = mime:dc.format.mimetype`
  - `search.index.10 = sponsor:dc.description.sponsorship`
  - `search.index.11 = identifier:dc.identifier.*`
  - `search.index.12 = language:dc.language.iso`

  For example `search.index.1 = author:dc.contributor.*` means to create an “author” index containing all dc.contributor. In this case instead of using a qualifier, a wildcard (*) has been used. This denotes to index all dc.contributor metadata. This can include dc.contributor.advisor or dc.contributor.committeeMember. In some cases instead of using Lucene search Discovery are used. In the case of SDSU Discovery is used. Hence `xmlui.xconf` file in the config directory it has been configured to use Discovery.

  <aspect name="Discovery" path="resource://aspects/Discovery/"

- **Browse**: The browse API takes care of indexes of date, authors, title and subject. These indexes can be re indexed in any time by running the command ”dspace index-init” from the bin directory of deployment folder.

### 2.2.3 Storage Layer

Storage layer consists of relational database. The metadata and content are stored here and it consists of API which can be accessed from business logic layer. These APIs are collectively referred to as Storage API. The configuration can be adjusted to connect either
to Oracle or Postgres. Postgres is open source object relational database management system. Postgres is supported in many platforms like Linux, Microsoft Windows, Solaris and Mac OS X. Some of the features that Postgres supports are Indexes, Triggers, Data types, User defined objects, security, asynchronous notifications etc. Oracle is also object relational database management system. Oracle RDBMS stores data logically in the form of tablespaces and physically in the form of data files.

In DSpace the dspace.cfg file can be modified to mention the database that is being used. It has entries for database username and password. Figure 2.2 shows the server architecture.

![Server architecture](image)

**Figure 2.2. Server architecture.**

### 2.2 DATA MODEL

In DSpace, data is organized in a hierarchical way. The top level is called “Community” which is sub-divided into “Collection”. Collection is subdivided into “Items” which contains “Bundle” which in turn contains “Bitstream” [5]. Figure 2.3 [6] represents pictorial representation of the hierarchy and comparison with SDSU DSpace hierarchy.

- **Community**: In the case of San Diego State University, the top level communities are:
  1. Associated Students
  2. Geological Sciences Theses
  3. Instructional Technology Services
  4. San Diego State Student Newspapers
  5. SDSU course catalogs, bulletins, etc.
  6. SDSU Syllabi Collection
  7. SDSU Theses and Dissertations
8. SDSU Undergraduate Research Projects
9. Special Collections & University Archives

- **Collections**: A collection may appear in more than one community. In the case of SDSU the collection for SDSU Theses and Dissertations is Theses and Dissertations.

- **Items**: Items are basic archival elements of the archive. Even though an item can be in multiple collections, it can be owned only by one collection.

- **Bundles**: In practice, most items tend to have these named bundles [7].
  - **ORIGINAL** -- the bundle with the original, deposited bitstreams
  - **THUMBNAILS** -- thumbnails of any image bitstreams
  - **TEXT** -- extracted full-text from bitstreams in ORIGINAL, for indexing
  - **LICENSE** -- contains the deposit license that the submitter granted the host organization; in other words, specifies the rights that the hosting organization have
  - **CC_LICENSE** -- contains the distribution license.

- **Bitstream**: Bitstreams are streams of bits, usually computer files. Each bitstream has a bitstream format. Bitstream format is unique in DSpace. Each item has Dublin Core metadata record. Dublin Core is added to item either during the submission process or derived from other metadata as part of ingest process. Since preservation is crucial in DSpace, the bitstream formats of the files the users...
submit are captured. In addition each bitstream format has a support type. This indicates how well the format will be preserved in future. The three possibilities are Supported, Unsupported and Known. Table 2.1. [4] shows the DSpace System Architecture and Table 2.2 shows pictorial representation of DSpace server architecture.

Table 2.1. MIT Libraries' Definitions of Bitstream Format Support Levels

<table>
<thead>
<tr>
<th>Supported</th>
<th>The format is recognized, and the hosting institution is confident it can make bitstreams of this format useable in the future, using whatever combination of techniques (such as migration, emulation, etc.) is appropriate given the context of need.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Known</td>
<td>The format is recognized, and the hosting institution will promise to preserve the bitstream as-is, and allow it to be retrieved. The hosting institution will attempt to obtain enough information to enable the format to be upgraded to the 'supported' level.</td>
</tr>
<tr>
<td>Unsupported</td>
<td>The format is unrecognized, but the hosting institution will undertake to preserve the bitstream as-is and allow it to be retrieved.</td>
</tr>
</tbody>
</table>


2.3 METADATA

DSpace supports three types of metadata. They are (a) Descriptive (b) Administrative and (c) Structural metadata.

- **Descriptive Metadata:** Though DSpace supports other metadata, it is pre-configured to use Dublin Core metadata. Communities and Collections have some simple descriptive metadata which are stored in DBMS.

- **Administrative Metadata:** This includes preservation metadata, provenance and authorization policy data.

- Structural Metadata: The information on how to present an item or bitstreams is represented by structural metadata.

2.4 HANDLES

Each DSpace instance can get a handle prefix from CNRI Handle system. This can be specified in DSpace configuration file (dspace.cfg). DSpace handles are assigned to
Table 2.2. Example of Data Model in SDSU DSpace

<table>
<thead>
<tr>
<th>Community</th>
<th>SDSU Theses and Dissertations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Collection</td>
<td>Theses and Dissertations</td>
</tr>
<tr>
<td>Item</td>
<td>A technical report</td>
</tr>
<tr>
<td>Bundle</td>
<td>A HTML document</td>
</tr>
<tr>
<td>Bitstream</td>
<td>A single image file or pdf file</td>
</tr>
<tr>
<td>Bitstream Format</td>
<td>JPEG encoded image format or application/pdf for pdf file</td>
</tr>
</tbody>
</table>

Communities, Collections and Items. For example in the case of San Diego State University the handle.prefix specified in DSpace configuration file is “10211.10”. The handle for SDSU Theses and Dissertations is http://sdsu-dspace.calstate.edu/handle/10211.10/188.
CHAPTER 3

DSpace Installation

DSpace can be installed on PC, laptop or on server hardware as it is written in Java. For installing DSpace on a production server good server is recommended. Though minimum DSpace production requirement is 4GB of Random Access Memory, 8GB is recommended. In addition a Quad core processor, 73 GB 15,000 rpm network disks is recommended. Before working on production server it is a good practice to use a staging server with the same settings as in production server [3].

3.1 DSpace Pre-Requisites

For proper running of DSpace the following tools need to be installed. These are third party tools. Other tools may also be used for DSpace installation.

3.1.1 Apache Maven 2.2.x or Later

Maven is a Java build tool. Maven can be downloaded from the following location: http://maven.apache.org/download.html. The first build is done by using the following command:

- maven clean package

3.1.2 Apache Ant 1.8 or Later

Ant is used for second stage of the build process. Ant can be downloaded from the following location: http://ant.apache.org. Commands used are:

- ant fresh install: This copies the bin, config, solr, webapp source directories to the installation directory.
- ant update

3.1.3 Relational Database

Oracle or Postgres can be used as a database. For this thesis Postgres was used. A new database with “dspace” as the user was created. The user was given permission to add and remove tables to the database.
3.1.4 Servlet Engine

Apache Tomcat was used for this thesis. [tomcat]/conf/server.xml was modified to add <Connector> element in the configuration.

<Server port="8081" shutdown="SHUTDOWN">
<Connector port="8080" protocol="HTTP/1.1" connectionTimeout="20000"
redirectPort="8443"/>

3.2 DSpace Code

There are two releases of DSpace for each version.

1. Binary release
2. Source release

Binary Release 3.1 was installed for this thesis project. For most of the customization work this version is enough. If a development of greater depth needs to be done the source release is recommended.

3.2.1 DSpace Source Structure

If a source release is installed the directory structure will be as shown in Figure 3.1 [8].

<table>
<thead>
<tr>
<th>Directory</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dspace/</td>
<td>Will contain customizations for your repository</td>
</tr>
<tr>
<td>dspace-api/</td>
<td>Contains the core DSpace code</td>
</tr>
<tr>
<td>dspace-discovery/</td>
<td>Contains source code for the discovery feature.</td>
</tr>
<tr>
<td>dspace-jspui/</td>
<td>Contains source code for the JSP User Interface.</td>
</tr>
<tr>
<td>dspace-init/</td>
<td>Contains source code for the Lightweight Network Interface.</td>
</tr>
<tr>
<td>dspace-oai/</td>
<td>Contains source code for the OAI-PMH Interface.</td>
</tr>
<tr>
<td>dspace-stats/</td>
<td>Contains source code the statistics module.</td>
</tr>
<tr>
<td>dspace-sword/</td>
<td>Contains source code for the SWORD Interface.</td>
</tr>
<tr>
<td>dspace-swordv2/</td>
<td>Contains source code for the SWORD version 2 Interface.</td>
</tr>
<tr>
<td>dspace-sword-client/</td>
<td>Is a built-in SWORD client</td>
</tr>
<tr>
<td>dspace-xmlui/</td>
<td>Contains source code for the XML User Interface.</td>
</tr>
</tbody>
</table>

Figure 3.1. DSpace source structure. Source: @mire, Introduction to DSpace Development, n.d. 2010.
3.2.2 DSpace Binary Release Structure

If a binary release is installed structure will be as shown in Figure 3.2 [8]. The main directory will be dspace and the following directories come beneath dspace.

- *dspace/* - DSpace 'build' and configuration module

<table>
<thead>
<tr>
<th>Directory</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bin/</td>
<td>Contains shell and Perl scripts for running DSpace command-line tasks.</td>
</tr>
<tr>
<td>config/</td>
<td>Configuration files</td>
</tr>
<tr>
<td>docs/</td>
<td>DSpace documentation</td>
</tr>
<tr>
<td>etc/</td>
<td>Database upgrade and installation scripts</td>
</tr>
<tr>
<td>modules/</td>
<td>Customizations to the codebase for your repository</td>
</tr>
<tr>
<td>jspui/</td>
<td>To place customizations to the JSP user interface</td>
</tr>
<tr>
<td>ini/</td>
<td>To place customizations to the Lightweight Network Interface</td>
</tr>
<tr>
<td>cao/</td>
<td>To place customizations to the OAI-PMH interface</td>
</tr>
<tr>
<td>solr/</td>
<td>To place customizations to the SOLR server. Out of the box, used for statistics.</td>
</tr>
<tr>
<td>sword/</td>
<td>To place customizations to the SWORD interface</td>
</tr>
<tr>
<td>swordv2/</td>
<td>To place customizations to the SWORD version 2 interface</td>
</tr>
<tr>
<td>xmlui/</td>
<td>To place customizations to the XML user interface (Manakin)</td>
</tr>
<tr>
<td>solr/</td>
<td>Contains configuration files for the solr server and statistics module.</td>
</tr>
<tr>
<td>src/</td>
<td>Source location of build files</td>
</tr>
</tbody>
</table>

Figure 3.2. DSpace binary release structure. Source: @Mire Labs, *Introduction to DSpace Development*, Unpublished report, 2010.

The config directory is the most important directory and many customizations can be done just by changing the dspace.cfg. Figure 3.3 [8] shows the config directory structure and a short description of each folder in it.

3.3 INSTALLATION PROCESS

Once the source is copied to a folder, a project must be created using an IDE (Integrated Development Environment). Several other IDEs can be used like Eclipse, Netbeans etc. But for this thesis “IntelliJ” was used.

3.3.1 Creating a Database in Postgres

In Windows

- Connect to the local database as “postgres” user.
- Create a login user as “dspace” and password “dspace”.
- Create a database called “dspace-sdsu” with owner “dspace” with UTF-8 encoding.

### 3.3.2 Maven Build Process

First the DSpace project needs to be built. This is done using maven.

The command used for building is “maven clean package”. This first cleans and then packages DSpace. Once the build process is successful, the dspace.cfg from the dspace target file needs to be changed. If Windows machine is used, forward slashed must be used.

For example:

```bash
dspace.dir = c:/DSpace/dspace_installation
```

where dspace_installation will be the folder for installation. This folder must be created by the user. The next process is editing some other important fields. dspace.url must be set to the host and port. If testing locally it can be set to localhost:port as below:

```bash
dspace.url = http://localhost:8080
db.name = postgres
db.url = jdbc:postgresql://localhost:5432/dspace-sdsu
db.driver = org.postgresql.Driver
```
db.username = dspace
db.password = dspace

where dspace-sdsu is the database which is being used. Figure 3.4. Shows the screenshot of IntelliJ showing maven settings.

![Screenshot of IntelliJ showing maven settings.](image)

**Figure 3.4. Screenshot of IntelliJ showing maven settings.**

### 3.3.3 Deployment Process (Ant)

Basically there are two commands which are often used for DSpace deployment

- ant fresh install
- ant update

The ant commands are run from dspace/target/dspace-version-build/ directory.

The build file to configure the tasks that ant can execute for the DSpace project is located in dspace/src/main/config/build.xml

- ant fresh install: This command is used for the first time. This command does the database setup and registry loading.
- ant update: This command updates existing installation. This leaves the data and configuration intact.
Once the command is run, the installation directory will be updated with a bunch of
folders of which the important ones are config and bin directories. Figure 3.5 [8] shows the
Dscape deployment directory structure and a short description to each folders in it.

<table>
<thead>
<tr>
<th>dspace/</th>
<th>Asset store files</th>
</tr>
</thead>
<tbody>
<tr>
<td>assetstore/</td>
<td>Shell and Perl scripts for running DSpace command-line tasks</td>
</tr>
<tr>
<td>bin/</td>
<td>Configuration, with sub-directories</td>
</tr>
<tr>
<td>config/</td>
<td>Database upgrade and installation scripts</td>
</tr>
<tr>
<td>etc/</td>
<td>Handle server files</td>
</tr>
<tr>
<td>handle-server/</td>
<td>JARs, including the DSpace classes</td>
</tr>
<tr>
<td>lib/</td>
<td>Log file</td>
</tr>
<tr>
<td>log/</td>
<td>Reports generated by statistical report generator</td>
</tr>
<tr>
<td>search/</td>
<td>Lucene search index files</td>
</tr>
<tr>
<td>solr/</td>
<td>Contains solr and statistics configuration and statistics data and the solr engine configuration for the discovery feature.</td>
</tr>
<tr>
<td>upload/</td>
<td>temporary directory used during file uploads etc.</td>
</tr>
<tr>
<td>webapps/</td>
<td>location where DSpace installs all Web Applications</td>
</tr>
</tbody>
</table>

**Figure 3.5. DSpace deployment directory structure. Source: @Mire Labs, *Introduction to DSpace Development*, Unpublished report, 2010.**

### 3.3.4 Deployment Process (Tomcat)

After completing the above steps the dspace_installtion directory will be updated.

Tomcat can be configured to use this directory as the base of webapps.

The files that need to be changed are:

1. apache-tomcat/conf/Calalina/localhost/ROOT.xml
   a. ROOT.xml: In ROOT.xml the following parameters should be set. The value of these dspace-config and dspace.dir should point to the installation directory.

2. apache-tomcat/conf/Calalina/localhost/solr.xml
   a. Solr.xml: In the case of solr.xml in addition to setting the dspace-config and dspace.dir parameters, the environment name should be set. The value should point to solr directory in installation folder.

3. apache-tomcat/conf/server.xml
   a. Server.xml: In server.xml the connector port and server port should be set. The connector port should be same as used in dspace.cfg
CHAPTER 4

MANAKIN ARCHITECTURE

4.1 INTRODUCTION

Manakin is built on top of apache Cocoon web development framework. This uses an XML based pipeline architecture which passes SAX Events. Figure 4.1 [9] shows a cocoon pipeline processing. Here the SAX events are passed through each pipeline which is a File Generator, XSLT Transformer and HTML Serializer. The three major types of pipeline components are Generators, Transformers and Serializers [9]. The details of these components are presented in this chapter. Others are readers, selectors, matchers, pipes and actions.


4.2 THE COCOON SITEMAP

Sitemap consists of configuration information for the pipelines that are being used [10]. The main sitemap is in Cocoon Web application folder. This is called sitemap.xmap. This root sitemap may contain links to sub sitemaps. The sub sitemaps are in root of its own application folder. Figure 4.2 [11] shows cocoon pipeline processing showing sitemap.
4.2.1 Cocoon Sitemap Structure

Cocoon sitemap has the following structure [12]:

```xml
<map:sitemap xmlns:map="http://apache.org/cocoon/sitemap/1.0">
    <map:components/>
    <map:views/>
    <map:resources/>
    <map:action-sets/>
    <map:pipelines/>
</map:sitemap>
```

4.2.2 Components

Components defines the first section. This may have various categories. Some of them are:

```xml
<map:components>
    <map:generators/>
    <map:transformers/>
    <map:serializers/>
    <map:readers/>
    <map:selectors/>
    <map:matchers/>
    <map:actions/>
    <map:pipes/>
</map:components>
```
4.2.3 Generators

Generator takes a component and produces SAX events. Simplest generator is a FileGenerator. Here is an example.

```xml
<map:generators default="file">
  <map:generator label="content" logger="sitemap.generator.file" name="file" pool-max="32" src="org.apache.cocoon.generation.FileGenerator"/>
</map:generators>
```

4.2.4 Transformers

Transformers consume SAX events and produce SAX events. Simplest transformer is XalanTransformer. This is similar to XML where it gets an XML document and generates another XML document [12]. An example is as follows:

```xml
<map:transformers default="xslt">
  <map:transformer logger="sitemap.transformer.xslt" name="xslt" pool-max="32"
                   src="org.apache.cocoon.transformation.TraxTransformer">
    <use-request-parameters>false</use-request-parameters>
    <use-session-parameters>false</use-session-parameters>
    <use-cookie-parameters>false</use-cookie-parameters>
    <xslt-processor-role>xalan</xslt-processor-role>
    <check-includes>true</check-includes>
  </map:transformer>
</map:transformers>
```

4.2.5 Serializers

Serializers gets the SAX Events and transforms to a presentation format. Simplest serializer is XMLSerializer which streams SAX event into XML. There are many built in serializers for generating XML, PDF, HTML [13].

```xml
<map:serializer name="html" logger="sitemap.serializer.html"
                src="org.apache.cocoon.serialization.HTMLSerializer" mime-type="text/html; charset=utf-8"
                pool-grow="2" pool-max="64" pool-min="2">
  <doctype-public>-//W3C//DTD XHTML 1.0 Transitional//EN</doctype-public>
  <doctype-system>http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd</doctype-system>
</map:serializer>
```
4.2.6 Matchers

When a request comes into Cocoon, matchers parsers and sends it to respective pipelines for processing.

```xml
<map:matchers default="wildcard">
  <map:matcher logger="sitemap.matcher.wildcard" name="wildcard"
              src="org.apache.cocoon.matching.WildcardURIMatcher"/>
  <map:matcher logger="sitemap.matcher.regexp" name="regexp"
              src="org.apache.cocoon.matching.RegexpURIMatcher"/>
</map:matchers>
```

4.3 THREE COMPONENTS

Manakin architecture defines three unique concepts

1. DRI schema (Digital Repository Interface)
2. Aspects
3. Themes

The DRI and Aspects takes part in content generation while the themes style the application.

Figure 4.3 [11] shows cocoon pipeline processing showing the aspect chain.

4.3.1 DRI Schema

DRI is a schema which defines the abstract representation of repository page. The structural elements are based on TEI (Text Encoding Initiative) while metadata elements are based on METS (Metadata Encoding and Transmission Standard), MODS (Metadata Object Description Schema), DIM (DSpace Intermediate Metadata Format) or simple DC (Dublin Core). The structure of DRI schema can be represented in Figure 4.4 [11].

The main elements of DRI are:

1. Document
2. Body

3. Options

4. Meta

- **Document**: The root of all DRI page is `<document>` tag. All other elements come as a sub list of `<document>` tag. Hence structure will be as given below.

```
<document version="1.1">
    <body></body>
    <options></options>
    <meta></meta>
</document>
```

- **Body**: The `<body>` is the main top level element which consists of structural elements in a DRI document. Any structural elements can be written in DSpace which can be transformed by XSL templates into other formats. This makes it feasible to apply uniform templates and styling rules to all DSpace pages.

```
<body>
    <div id="aspect.artifactbrowser.CommunityBrowser.div.comunity-browser" rend="primary" n="comunity-browser"></div>
    <div id="aspect.artifactbrowser.FrontPageSearch.div.front-page-search" interactive="yes" rend="primary" action="/search" n="front-page-search" method="get">
    </div>
</body>
```

The Figure 4.5 shows the screenshot of SDSU body with the container mentioned above. It has “FrontPageSearch” and “CommunityBrowser”.

- **Options**: The `<option>` element consists of all navigation information. There are stored as lists instead of metadata elements.

```
<options>
    <list id="aspect.viewArtifacts.Navigation.list.browse" n="browse">
    </list>
    <list id="aspect.viewArtifacts.Navigation.list.account" n="account">
    </list>
    <list id="aspect.viewArtifacts.Navigation.list.context" n="context"/>
    <list id="aspect.viewArtifacts.Navigation.list.administrative" n="administrative"/>
</options>
```
Figure 4.5. Screenshot of SDSU DSpace body with different containers.

The Figure 4.6 shows the DRI options tag and the corresponding navigation section for SDSU on browser.

- *Meta:* The `<meta>` element consists of all the metadata elements in a DRI document. The metadata contained in `<meta>` element has three main components namely `<userMeta>`, `<pageMeta>` and `<repositoryMeta>`. All these store metadata information about their respective component. Below is an example of meta element.

```html
<meta>
  <userMeta authenticated="no">
    <metadata element="identifier" qualifier="loginURL">/login</metadata>
    <metadata element="language" qualifier="RFC3066">en_US</metadata>
    <metadata element="language" qualifier="RFC3066">en</metadata>
  </userMeta>
  <pageMeta>
    <metadata element="contextPath"/>
  </pageMeta>
</meta>
```
4.3.2 Aspects

Aspects are interactive components and they can add or modify features in a digital repository. In Aspect Oriented Programming (AOP) programs are divided into aspects which overlap as little as possible. Aspects can be chained so that each aspect can modify a page by
adding its own components. Aspects take DRI as input, modifiers and produces DRI output.

The four core aspects in DSpace are:

1. **Artifact Browser**: Responsible for:
   - Browsing communities, collections, items, bitstreams
   - Viewing individual items
   - Searching the repository

2. **E-Person**: Responsible for:
   - Logging In
   - Logging Out
   - Registering new users
   - Dealing with forgotten passwords, changing passwords
   - Editing Profile

3. **Submission**: Responsible for:
   - Submitting new item
   - Determining workflow process
   - Ingesting the new item in digital repository

4. **Administrative**: Responsible for:
   - Create/Modify/remove all communities, collections, e-persons, registries, authorizations.

Figure 4.7 shows the screenshot of aspects from IntelliJ.

Aspects reside in self-contained packages. They are written in Java, javascript flow scripts and XML. The Figure 4.8 [11] shows the pipeline processing. The generator creates initially an empty DRI and as it goes through each aspect it adds its own content. At the end the new DRI will be sent to the Serializer and set to browser.

### 4.3.3 Themes

Themes stylize content into a particular look and feel [14]. Themes may be applied to

- Single page
- All pages in a community
Figure 4.7. Screenshot from IntelliJ of aspects residing in dspace-xmlui-api library.

Themes like aspects remain in self-contained packages. They include:

- XSL stylesheets
- CSS (Cascading Style Sheet)
- Images
- Static resources

Figure 4.9 [15] shows the themes and styling added after the content is generated in DRI.

The Cascading Style Sheets to be used are mentioned in the sitemap and those will be added to the content.

In the config folder is a file called xmlui.xconf. This file specifies which theme to activate.

This is specified in `<themes>` section. The `<aspects>` section lists the aspects that will be used for the repository.

```
<aspects>
  <aspect name="Displaying Artifacts" path="resource://aspects/ViewArtifacts/"/>
```
<aspect name="Browsing Artifacts" path="resource://aspects/BrowseArtifacts/" />
<aspect name="Searching Artifacts" path="resource://aspects/SearchArtifacts/" />
<aspect name="Administration" path="resource://aspects/Administrative/" />
<aspect name="E-Person" path="resource://aspects/EPerson/" />
<aspect name="Submission and Workflow" path="resource://aspects/Submission/" />

</aspects>

<themes>

<!-- Example configuration -->

<theme name="Test Theme 1" handle="123456789/1" path="theme1/"/>
<theme name="Test Theme 2" regex="community-list" path="theme2/"/>

<!-- Mirage theme, @mire contributed theme, default since DSpace 3.0 -->

<theme name="Atmire Mirage Theme" regex=".*" path="Mirage/"/>

</themes>
CHAPTER 5

IMPLEMENTATION OF DSPACE ON MOBILE FOR SDSU

SDSU DSpace consists of many features and in this thesis a new mobile theme customized for SDSU has been implemented. This theme makes it easier for mobile users to search, browse and view the items and bitstream. The administrators can login and view all admin facets and access them in mobile theme. This chapter covers all features in detail. The main page of the mobile theme is shown in Figure 5.1 and Figure 5.2 shows the screen shot when scrolled down.

![Figure 5.1. Screenshot of home page top and bottom part on a mobile.](image)

5.1 SDSU DSPACE DATABASE

The first section consists of all the communities in SDSU DSpace. For this a dump of SDSU database was created and restored in test environment, pgAdmin III was used for restoring the SDSU database dump. The Figure 5.2 shows the list of tables that were restored and the Figure 5.3 shows the community tables restored.
5.2 MOBILE THEME FILE STRUCTURE

The first mobile theme was introduced in DSpace 3.0 [2]. Hence this theme was taken as the base and customizations done on it. The mobile theme has the following file structure as shown in Figure 5.4 [16]. The “mobile” folder comes in same level as other default theme [16]. In SDSU the default theme used is Mirage. All the javascripts, CSS (Cascading style Sheet) is in “lib” directory. The images used for the mobile themes are in “images” folder which is under “lib” directory.
5.3 THEMES.XMAP

The main themes.xmap under the themes directory was modified to handle requests coming from mobile.* site. A new match was written to match the new domain and limit it to use mobile theme. The pattern “mobile.*” denotes to use the theme in folder named mobile when a request comes from mobile.*. Others will use the theme which is configured in xmlui.xconf under the deployment config directory.

```xml
<map:match type="WildcardHostMatcher" pattern="mobile.*">
  <map:mount uri-prefix="" src="mobile/"/>
</map:match>
```

Other themes will be matched under the below match by the ThemeMatcher transformer.

```xml
<map:match type="ThemeMatcher">
  <map:mount uri-prefix="" src="{theme}"/>
</map:match>
```
5.4 CSUGeneralTheme.xsl

The default theme used by SDSU was CSUGeneralTheme. Hence two javascript files were added to the CSUGeneralTheme. They are detectmobile.js and cookies.js. The script type is set as “text/javascript” and the src attribute value is set as path to the javascript file. cookies.js and detectmobile.js are placed in the lib directory in mobile theme.

```xml
<script type="text/javascript">
    <xsl:attribute name="src">
        ../../themes/mobile/lib/cookies.js</xsl:text>
    </xsl:attribute>&#160;
</script>

<script type="text/javascript">
    <xsl:attribute name="src">
        ../../themes/mobile/lib/detectmobile.js</xsl:text>
    </xsl:attribute>&#160;
</script>

In page-structure.xsl of mirage theme a link “View mobile site” which allows the users to view full site on their mobile devices was added. The cookies.js saves the preference and is erased when the session is closed [16].

```xml
<div id="mobile-link">
    <a href="mobile.localhost:8095" onclick="eraseCookie('viewfull');window.location='http://mobile.localhost:8095';">View mobile site</a>
</div>
```
5.5 Mobile Sitemap.xmap

Based on cocoon framework, each theme has its own main sitemap. This sitemap for mobile theme is located in the folder named “mobile’ under the “themes” directory. This is the heart of the theme and the configuration information about how to process DRI document passed to it from aspect chain. The mobile sitemap has the following configuration and steps.

- **Component configuration:** The component-configuration section has the information on the theme location and theme name. Theme path determines the directory name in which the theme is located. The theme name is used for descriptive purpose.

  ```xml
  <map:component-configurations>
    <global-variables>
      <theme-path>mobile</theme-path>
      <theme-name>Mobile theme for SDSU</theme-name>
    </global-variables>
  </map:component-configurations>
  ```

- **Generate the DRI page:** This created DRI page.
  ```xml
  <map:generate type="file" src="cocoon://DRI/{1}"/>
  ```

- **Add page metadata:** This adds an CSS reference entry in pageMetadata of DRI document. XSL then finds those entries and adds to head of XHTML output. The entries are added based on the browser.

  ```xml
  <map:select type="browser">
    <map:transform type="IncludePageMeta">
      <map:parameter name="stylesheet.screen#1" value="lib/style.css"/>
      <map:parameter name="stylesheet.screen#2" value="lib/style-ie6.css"/>
      <map:parameter name="theme.path" value="{global:theme-path}"/>
      <map:parameter name="theme.name" value="{global:theme-name}"/>
    </map:transform>
  </map:select>
  ```

- **Transform to XHTML:** This step transforms to XSL to XHTML. This may be done either by default dri2html.xsl or the theme can have its own XSL code.
• **Localize the page:** All the `<i18n:text>` tags are transformed to mobile friendly labels
  
  ```xml
  <map:act type="locale">
    <map:transform type="i18n">
      <map:parameter name="locale" value="mobile"/>
    </map:transform>
  </map:act>
  ```

• **Serialize to browser:** Serializes result to display in browser
  
  ```xml
  <map:serialize type="xhtml"/>
  ```

### 5.6 MOBILE.XSL

Then come the mobile theme structure. Most of the customization for mobile template is done in this XSL. XSL processing is done starting at the root level of DRI document. As mentioned in the previous chapter, the root of DRI document is `<document>`. Hence template is written to match `<document>` tag and HTML document is created giving it a head and body.

```xml
<xsl:template match="dri:document">
  <html>
    <!-- First of all, build the HTML head element -->
    <xsl:call-template name="buildHead"/>
    <!-- Then proceed to the body -->
    <body>
      <!-- first HTML5 line -->
      <div data-role="page" data-fetch="always">
        <xsl:call-template name="buildHeader"/>
        <xsl:apply-templates/>
        <xsl:call-template name="buildFooter"/>
      </div>
    </body>
  </html>
</xsl:template>
```
5.6.1 The Header

The buildHead template contains SDSU logo. Other headings like the title, subtitle can be added here. The buildHeader template is shown below. It consists of <a> tag which is a link to the <img> tag.

```xml
<xsl:template name="buildHeader">
  <!-- new header with a data-icon linking to the homepage -->
  <div data-role="header">
    <a>
      <xsl:attribute name="href">
        <xsl:text>http://library.sdsu.edu</xsl:text>
      </xsl:attribute>
      <xsl:attribute name="class">
        <xsl:text>logo_button</xsl:text>
      </xsl:attribute>
      <img>
        <xsl:attribute name="src">
        </xsl:attribute>
        <xsl:attribute name="width">
          <xsl:text>285</xsl:text>
        </xsl:attribute>
        <xsl:attribute name="height">
          <xsl:text>45</xsl:text>
        </xsl:attribute>
        <xsl:attribute name="style">
          <xsl:text>border:0</xsl:text>
        </xsl:attribute>
      </img>
    </a>
  </div>
</xsl:template>
```
5.6.2 The Body

<xsl: apply template> goes over the document tag's children elements: body, options, meta. The body template generates the ds-body div that contains all the content. The options template generates the ds-options div that contains the navigation and action options available to the user. The meta element is ignored since its contents are not processed directly, but instead referenced from the different points in the document.

- Adding navigation bar

jQuery mobile has a navbar widget that can be used to make up to 5 options with optional icon in a bar [17]. In this theme three options are used. They are:

1. Home
2. Administrative
3. Statistics

A nav bar is coded as an unordered list of links wrapped in a container that has data-role=“navbar” attribute. Navigation bar is added in the ds-body. This is added by matching the dri:body template. Figure 5.5. shows the screenshot of DSpace on mobile to show the navigation bar before and after logging in.

![Figure 5.5. Screenshot of DSpace on mobile to show the navigation bar before and after logging in.](image-url)
An example of creating the Home navigation bar is shown below. The class “ui-btn-active” is added so that “Home” is set active upon initialization of navigation bar. Additionally a class “ui-state-persist” is added to make the framework restore the active state each time the page is shown while it exists in the DOM.

```xml
<xsl:template match="dri:body">
  <div data-role="navbar" data-id="nav_bar" data-position="fixed" data-transition="slide">
    <ul>
      <li>
        <a>
          <xsl:attribute name="href">
            <xsl:value-of select="$mobile-url"/>
          </xsl:attribute>
          <xsl:attribute name="class">
            <xsl:text>ui-btn-active ui-state-persist</xsl:text>
          </xsl:attribute>
          <xsl:attribute name="data-icon">
            <xsl:text>home</xsl:text>
          </xsl:attribute>
          <xsl:attribute name="rel">
            <xsl:text>external</xsl:text>
          </xsl:attribute>
          Home
        </a>
      </li>
    </ul>
  </div>
</xsl:template>
```

5.6.3 The Footer

The buildFooter creates the container for footer. The data-role is “footer” for this container. The buildTemplate is shown below.

```xml
<xsl:template name="buildFooter">
</xsl:template>
```
5.7 PASSWORD LOGIN

On clicking the “Login” link in the home page the user is redirected to /password-login link. The user is prompted to enter the user name and password. There can be two types of logging: (1) Eperson and (2) Admin.

When logged in as eperson the facets for eperson are shown. This includes viewing their submissions. An admin can view and access the administrative panels. Admin panel includes a number of features like managing the eperson, groups, authorizations, metadata etc. The mobile theme for each of the features will be discussed below. Figure 5.6. shows the screenshot of Password Login page and logged in page.

![Password Login Screenshot](image)

Figure 5.6. Screenshot of password login page and logged in page.

5.8 ADMIN PANEL

An admin on logging in can view the following:
1. Context
2. Administrative panel

The following are the administrative options.
Figure 5. 7. shows the admin panel top page and Admin panel bottom page.

5.8.1 Create Community

This link lets the admin create a new community. The mandatory field is name of the new community. Other optional fields include description, introductory text and copyright text for the community. A Screenshot of create community page is shown in Figure 5.8.

Figure 5.8. Screenshot of create community page.
5.8.2 EPeople

The admin can view the list of eperson by clicking on “EPeople” link in administrative panel. The admin can manage the eperson by creating a new eperson or browsing eperson from the list of epeople. Figure 5.9 shows the screenshot of EPeople list and EPerson management.

![Figure 5.9. Screenshot of EPeople list and EPerson management.](image)

5.8.3 Authorizations

Each collection can have a number of policies. The admin can view these policies, alter and create a new policy. The authorization policy specifies who can view the collection and who can submit an item to this particular collection. Figure 5.10 shows the screenshot of Authorizations screen.

5.8.4 Metadata

Admin can view the schema and the metadata associated with that particular schema. Admin can also add new schema or new metadata. Figure 5.11 show the screenshot of list of metadata on a mobile.
Figure 5.10. Screenshot of authorizations screen.

Figure 5.11. Screenshot of metadata listing when admin clicks on “Metadata” link in admin panel.
5.8.5 Format

Admin can view formats in DSpace and add or edit a format. Figure 5.12 shows he bitstream format screen when admin clicks on “Format” link in admin panel. The list consists of all the bitstream formats that are supported.

![Format Screen](image)

Figure 5.12. Screenshot of format screen when admin clicks on “Format” link in admin panel.

5.8.6 Statistics

Admin can view the statistics. The third navigation bar lets the administrator view statistics. In DSpace 3.1 onwards the following statistics view are supported. These include (a) Usage statistics (b) Search Statistics (c) Workflow statistics. Figure 5.13 shows the screenshot of the statistics and usage statistics.

5.8.7 Submission and Workflow

Administrator and Eperson can view their submissions by clicking the link “Submissions” in home page. This redirects the user to /submissions page. The user can view their unfinished submissions, Archived submissions. An administrator can view their submissions in workflow too as in Figure 5.14. They can take a task, approve or reject as in Figure 5.15.
Figure 5.13. Screenshot of the statistics navigation and usage statistics.

Figure 5.14. Screenshot of unfinished items and workflow tasks seen on clicking the “Submissions” link in home page of mobile theme.
An administrator can take a task, approve or reject an item or send it back to the task pool. Once an item is approved it goes to next step in workflow which is configured in workflow.xml in the deployment config directory. An example of workflow step in SDSU workflow.xml is as follows: The below step is configured to send an item after submission to “reviewer”. Once the step is completed it us sent to next step called “editstep”. There the item is sent for review to “editors” and then to “final editor”. Once the final editor approves the item is archived.

```xml
<step id="reviewstep" role="reviewer" userSelectionMethod="claimaction">
  <outcomes>
    <step status="0">editstep</step>
  </outcomes>
  <actions>
    <action id="reviewaction"/>
  </actions>
</step>
```

### 5.9 Search and Advanced Search

The mobile theme also supports search and advanced search. SDSU is using discovery search instead of lucene search. Hence in the template dri:body a new container is
added with data-role="content". The search option is in the home page. A form is submitted when the “Search” button with an input is given. The Figure 5.16 and Figure 5.17 show the search feature and results. Figure 5.18 shows screenshot of a filter applied and the resulting search result.

Figure 5.16. Screenshot of search for “catalog” and the search results produced.

Figure 5.17. Screenshot of advanced search filter and sort options.
Figure 5.18. Screenshot of a filter applied and the resulting search result.

<xsl:template match="dri:body">
<xsl:choose>
    <h3>
      <i18n:text>xmlui.mobile.search_all</i18n:text>
    </h3>
  </xsl:when>
</xsl:choose>
<form xmlns:i18n="http://apache.org/cocoon/i18n/2.1"
  xmlns="http://di.tamu.edu/DRI/1.0/"
  id="aspect_discovery_SiteViewer_div_front-page-search"
  class="ds-interactive-div primary"
  action="/discover"
  method="get"
  onsubmit="javascript:tSubmit(this);">
  <p class="ds-paragraph">
    <p> <label class="ds-form-label" for="aspect_discovery_SiteViewer_field_query">Enter some text in the box below to search DSpace.</label> </p>
  <input xmlns:i18n="http://apache.org/cocoon/i18n/2.1"
    xmlns="http://di.tamu.edu/DRI/1.0/"
    id="aspect_discovery_SiteViewer_field_query"
    class="ds-text-field" name="query" type="text" value=""/>
</form>
<input xmlns:i18n="http://apache.org/cocoon/i18n/2.1"
id="aspect_discovery_SiteViewer_field_submit" class="ds-button-field"
name="submit" type="submit" value="Go" />

</form>
</div>
</xsl:template>

5.10 REGISTER

The user is given the option to register too. A “Register” button which has a link to /register is displayed at the bottom. The container for the register is as follows:

<div id="register-front-page" data-role="controlgroup">
  <a>
    <xsl:attribute name="href">
      <xsl:text>/register</xsl:text>
    </xsl:attribute>
    <xsl:attribute name="data-role">
      <xsl:text>button</xsl:text>
    </xsl:attribute>
    <xsl:attribute name="data-icon">
      <xsl:text>arrow-r</xsl:text>
    </xsl:attribute>
    <xsl:attribute name="data-iconpos">
      <xsl:text>right</xsl:text>
    </xsl:attribute>
    <i18n:text>xmlui.mobile.register</i18n:text>
  </a>
</div>

5.11 RECENT SUBMISSIONS

The recent submissions are shown on home page and item page in mobile theme.
A screenshot of the recent submission is given in Figure 5.19. The number of recent submissions shown can be controlled by configuring the entry “recent.submissions.count”. For SDSU it is set as 5. Also the sort-option for recent submission is specified in dspace.cfg.

![Figure 5.19. Screenshot of recent submission in home page (left) and item view page (right).](image)

### 5.12 Item View

Item page shows the details about an item and also the bitstream for that item. A user may download the file and view on the mobile device. Figure 5.20 shows the screenshot of an item and its file in mobile theme.
Figure 5.20. Screenshot of Item view and bitstream attached.
CHAPTER 6

CONCLUSION

The DSpace mobile theme for San Diego State University was built to be simple and user friendly that would allow users and administrators to access SDSU digital collection on smartphones, tablets or low end phones. In DSpace mobile theme for SDSU, as search feature was used by most of the users, advanced search feature was included in this theme. Also administrators can access all admin features via this mobile theme and have a good user experience. The mobile theme supports other features like recent submissions, submissions and workflow. With the widespread use of mobile telephony, providing a mobile theme for SDSU DSpace will reach out to faculty and other interested parties.
CHAPTER 7

FUTURE WORK

Since SDSU DSpace is on DSpace 1.8 which is an older version, the mobile theme which is added in DSpace 3.0 was used as the base and customizations were done. In future when SDSU DSpace will be upgraded to newer version, the current mobile theme customizations can be migrated easily and deployed.

Future work also includes testing mobile theme on different mobile and tablets and on different browsers.

Because of the rapidly growing number of new devices and screen sizes, it has become impractical to make a mobile friendly theme that is compatible with the majority of mobile devices and screen sizes currently in market. The discussion seems to suggest that the solution relies on the concept of “Responsive Web Design” which is an approach where the site is created to provide optimal user experience regardless of the end-user’s device [2]. Thus in future a responsive user design for DSpace would be a future work and ideal solution to above issue.
REFERENCES


