Geospatial Metadata Community Adaptor
Applying XSLT Technologies to Geographic Metadata to Address
Interoperability and Compatibility Issues

A dissertation submitted in partial fulfillment of the
requirements for the degree of Doctor of Philosophy in
Computer Science and Engineering

by
Qiping Yan

Dr. Sergiu Dascalu/Dissertation Advisor

May, 2014
We recommend that the dissertation
prepared under our supervision by

QIPING YAN

Entitled

Geospatial Metadata Community Adaptor
Applying XSLT Technologies To Geographic Metadata To Address Interoperability
And Compatibility Issues

be accepted in partial fulfillment of the
requirements for the degree of

DOCTOR OF PHILOSOPHY

Sergiu Dascalu, Ph.D., Advisor

Frederick C. Harris, Jr, Ph.D., Committee Member

Mehmet Gunes, Ph.D., Committee Member

Yanto Shen, Ph.D., Committee Member

Janet Usinger, Ph.D., Graduate School Representative

David W. Zeh, Ph.D., Dean, Graduate School

May, 2014
Acknowledgements

I would like to take this opportunity to express my profound gratitude and deep regards to my advisor Dr. Sergiu Dascalu for his exemplary guidance, monitoring and constant encouragement throughout the course of this dissertation. The advice, help, and guidance given by him time over time shall carry me a long way in the journey of life on which I am about to embark. Dr. Dascalu has always been given me invaluable guidance, support, and advices throughout my program of study. I have learnt more than just knowledge from him. This dissertation would not have been possible without him.

I also would like to thank Dr. Fred Harris, Dr. Yantao Shen, Dr. Mehmet Gunes, and Dr. Janet Usinger for serving as committee members for this dissertation and their precious support and advice.

My thanks also go to my colleagues who work in the Software Engineering Lab (SOELA) at UNR and workers in the “Collaborative Research: Cyberinfrastructure Developments for the Western Consortium of Idaho, Nevada, and New Mexico” project. I have received valuable feedbacks, inputs, and suggestions from them during my research of my dissertation. In particular, I would like to thank Dr. Richard Kelley for suggesting the initial ideas from which this work started.

I am grateful to the IGT Knowledge Center of University of Nevada, Reno for providing me the private study carol. That helped me on concentrating on writing the dissertation.

This material is based in part upon work supported by the National Science Foundation under grant number EPS-0919123. I am grateful to the National Science Foundation and the Nevada System of Higher Education for the opportunity provided to me to work on this project.

For my family, I could not simply say only “Thank you”. Nothing is enough to express my gratitude to my parents, my wife, and my son.
Disclaimer

Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author and do not necessarily reflect the views of the National Science Foundation.
Abstract

In today's world, geographic data plays an increasing role in many areas, including academic research, government decision making, and in peoples everyday lives. As the quantity of geographic data gets larger, making full use of the data in a distributed, heterogeneous network environment, like the Internet, becomes a major issue. To better utilize and share those valuable resources, metadata standards have been developed. Metadata makes it easier to discover, explore, and share geographic data, particularly for cataloging geographic data in clearinghouses. But one of the new problems that emerged with metadata is interoperability since multiple metadata standards exist. Important geographic metadata standards include: ISO 19115, Dublin Core, CSDGM (US) and prENV 12657 (Europe). Semantically, metadata standards are distinct but, rather, they overlap and relate to each other in diverse and complex ways. In this dissertation, we propose a method and supporting software toolset, entitled Metadata Community Adaptor (MCA), to perform transformations between different geographic metadata standards, and help solve certain interoperability issues by using the new popular web service and XML/XSLT technologies. The distinguishing characteristic of the proposed approach consists of the unique combination of capabilities for dealing with metadata creation, metadata validation, and metadata transfer between standards. The dissertation provides details of the proposed approach and supporting software tools and includes extended descriptions of software specification and design. All the main capabilities of MCA are illustrated with the aid of four application scenarios. A feature-based comparison with related work is also included, and an outline of possible directions of future work is provided.
Contents

1 Introduction ................................................................. 1
   1.1 The problem ........................................................... 1
   1.2 The proposed approach ............................................... 2
   1.3 NSF EPSCoR collaborative research project ......................... 3
   1.4 Organization of the dissertation .................................... 3

2 Background: metadata exchange .......................................... 5
   2.1 Metadata ............................................................... 8
   2.2 Metadata standards .................................................. 8
   2.3 Geographical metadata standards ................................... 10
   2.4 ISO 19115 ............................................................ 12
   2.5 Tri-State metadata exchange ........................................ 13

3 Background: Related technologies ....................................... 15
   3.1 Web service .......................................................... 15
      3.1.1 What is a web service? .......................................... 16
      3.1.2 Classes of web services ....................................... 16
      3.1.3 Web service architecture ...................................... 16
   3.2 XML .................................................................... 17
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.2.1</td>
<td>What is XML?</td>
<td>17</td>
</tr>
<tr>
<td>3.2.2</td>
<td>What does XML do?</td>
<td>18</td>
</tr>
<tr>
<td>3.2.3</td>
<td>XML-related technologies</td>
<td>20</td>
</tr>
<tr>
<td>3.3</td>
<td>XSLT</td>
<td>20</td>
</tr>
<tr>
<td>3.3.1</td>
<td>What is XSLT ?</td>
<td>20</td>
</tr>
<tr>
<td>3.3.2</td>
<td>Logical components of an XSLT application</td>
<td>21</td>
</tr>
<tr>
<td>3.4</td>
<td>ASP.NET/C#</td>
<td>26</td>
</tr>
<tr>
<td>3.4.1</td>
<td>What is ASP.NET?</td>
<td>26</td>
</tr>
<tr>
<td>3.4.2</td>
<td>ASP.NET fundamentals</td>
<td>27</td>
</tr>
<tr>
<td>3.4.3</td>
<td>Event-driven application</td>
<td>28</td>
</tr>
<tr>
<td>3.4.4</td>
<td>ASP.NET server controls</td>
<td>29</td>
</tr>
<tr>
<td>3.4.5</td>
<td>Web service support in ASP.NET</td>
<td>30</td>
</tr>
<tr>
<td>3.5</td>
<td>SQL Server</td>
<td>30</td>
</tr>
<tr>
<td>4</td>
<td>Overview of the MCA approach</td>
<td>32</td>
</tr>
<tr>
<td>4.1</td>
<td>Motivation</td>
<td>32</td>
</tr>
<tr>
<td>4.2</td>
<td>The MCA approach</td>
<td>34</td>
</tr>
<tr>
<td>4.3</td>
<td>Similar work and discussion</td>
<td>49</td>
</tr>
<tr>
<td>5</td>
<td>Requirements and design</td>
<td>61</td>
</tr>
<tr>
<td>5.1</td>
<td>Use case modeling</td>
<td>61</td>
</tr>
<tr>
<td>5.1.1</td>
<td>Use case diagram</td>
<td>62</td>
</tr>
<tr>
<td>5.1.2</td>
<td>Description of use case</td>
<td>64</td>
</tr>
<tr>
<td>5.1.3</td>
<td>Key use case in details</td>
<td>64</td>
</tr>
<tr>
<td>5.2</td>
<td>Requirements specification</td>
<td>72</td>
</tr>
<tr>
<td>5.2.1</td>
<td>Functional requirements</td>
<td>73</td>
</tr>
<tr>
<td>5.2.2</td>
<td>Non-functional requirements</td>
<td>75</td>
</tr>
</tbody>
</table>
6 User interface aspects

6.1 User interface ........................................ 82
   6.1.1 User interface matters .......................... 83
   6.1.2 User interface design .......................... 83
   6.1.3 Guidelines for user interfaces ................. 84
6.2 MCA user interface .................................. 88
   6.2.1 Design overview ............................... 88
   6.2.2 Features, functions, and technical aspects .... 89

7 Application scenarios .................................. 93

7.1 Application scenario 1: Metadata creation and submission ....................... 93
7.2 Application scenario 2: Translation with MCA’s web services ..................... 97
7.3 Application scenario 3: Validate a geospatial metadata .......................... 98
7.4 Application scenario 4: Translate geospatial resource from one metadata standard to a different standard .......................... 100

8 Comparison with related work ................................ 104

8.1 DataONE ........................................ 104
   8.1.1 DataONE Cyberinfrastructure .................. 105
8.2 Geoportal server .................................... 106
   8.2.1 Key Features .................................. 106
8.3 GeoNetwork Open Source ................................ 107
8.4 ArcGIS/ArcCatalog .................................. 109
8.5 Feature based comparison ................................ 110
8.6 Summary ................................................................. 110

9 Future work and conclusions 112
  9.1 Future works .......................................................... 113
  9.2 Conclusions ........................................................... 115

A Acronyms 117

B Sample metadata documents 121
## List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>The web service Technology Stack</td>
<td>16</td>
</tr>
<tr>
<td>3.2</td>
<td>Browser support for XML and XSLT [56]</td>
<td>23</td>
</tr>
<tr>
<td>3.3</td>
<td>Common events and event handler</td>
<td>29</td>
</tr>
<tr>
<td>5.1</td>
<td>Use case description of MCA application</td>
<td>64</td>
</tr>
<tr>
<td>5.2</td>
<td>Detailed use case template</td>
<td>65</td>
</tr>
<tr>
<td>5.3</td>
<td>Detailed use case of UC12: UploadMetadata</td>
<td>66</td>
</tr>
<tr>
<td>5.4</td>
<td>Detailed use case of UC13: ValidateMetadata</td>
<td>67</td>
</tr>
<tr>
<td>5.5</td>
<td>Detailed use case of UC14: CreateMetadata</td>
<td>68</td>
</tr>
<tr>
<td>5.6</td>
<td>Detailed use case of UC15: TranslateMetadata</td>
<td>69</td>
</tr>
<tr>
<td>5.7</td>
<td>Detailed use case of UC18: WSMetadatupload</td>
<td>70</td>
</tr>
<tr>
<td>5.8</td>
<td>Detailed use case of UC19: WSMetadatavalidation</td>
<td>71</td>
</tr>
<tr>
<td>5.9</td>
<td>Detailed use case of UC20: WSMetadatatranslation</td>
<td>72</td>
</tr>
<tr>
<td>5.10</td>
<td>Software requirement specification levels</td>
<td>73</td>
</tr>
<tr>
<td>5.11</td>
<td>Functional software requirements of MCA</td>
<td>73</td>
</tr>
<tr>
<td>5.12</td>
<td>Non-functional software requirements of MCA</td>
<td>75</td>
</tr>
<tr>
<td>8.1</td>
<td>Related work feature based comparison</td>
<td>111</td>
</tr>
</tbody>
</table>
List of Figures

2.1 Global mean land-ocean temperature change from 1880–2012, relative to the 1951–1980 mean. The black line is the annual mean and the red line is the 5-year running mean. The green bars show uncertainty estimates[2]. Source: NASA GISS ............... 7

3.1 Web service architecture: Web services provide a standard means of interoperating between different software applications, running on a variety of platforms and/or frameworks 17

3.2 A sample XML file ................................................................. 19

3.3 An XSLT processor takes one or more XML documents and one or more XSLT files as input and produces a new XML document ......................................................... 21

3.4 ASP.NET and IIS 7.x architecture ............................................. 28

4.1 Tr-State Geospatial Metadata Exchange ..................................... 35

4.2 MCA architecture ................................................................. 37

4.3 MCA software flowchart ......................................................... 38

4.4 ISO 19115 metadata standard entity UML diagram ...................... 39

4.5 ISO 19115 metadata standard identification UML diagram ........... 40

4.6 The MCA Approach: Top level workflow chart .......................... 42

4.7 The MCA Approach: Create metadata activity .......................... 43

4.8 The MCA Approach: Validate metadata activity ........................ 44
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.9</td>
<td>The MCA Approach: Upload metadata activity</td>
<td>45</td>
</tr>
<tr>
<td>4.10</td>
<td>The MCA Approach: Community contribution activity</td>
<td>46</td>
</tr>
<tr>
<td>4.11</td>
<td>The MCA Approach: Translate metadata activity</td>
<td>47</td>
</tr>
<tr>
<td>4.12</td>
<td>MCA home page</td>
<td>50</td>
</tr>
<tr>
<td>4.13</td>
<td>MCA login page</td>
<td>51</td>
</tr>
<tr>
<td>4.14</td>
<td>MCA about page</td>
<td>52</td>
</tr>
<tr>
<td>4.15</td>
<td>MCA Search page</td>
<td>53</td>
</tr>
<tr>
<td>4.16</td>
<td>MCA search result page</td>
<td>53</td>
</tr>
<tr>
<td>4.17</td>
<td>MCA upload metadata page</td>
<td>54</td>
</tr>
<tr>
<td>4.18</td>
<td>MCA create metadata page</td>
<td>55</td>
</tr>
<tr>
<td>4.19</td>
<td>MCA metadata translation web service page</td>
<td>56</td>
</tr>
<tr>
<td>4.20</td>
<td>MCA test page for metadata translation web service</td>
<td>57</td>
</tr>
<tr>
<td>4.21</td>
<td>MCA map viewer page</td>
<td>58</td>
</tr>
<tr>
<td>4.22</td>
<td>Tri-State Data Hub metadata detail page</td>
<td>59</td>
</tr>
<tr>
<td>5.1</td>
<td>MCA Use Case Diagram</td>
<td>63</td>
</tr>
<tr>
<td>5.2</td>
<td>Software requirement traceability matrix of MCA</td>
<td>77</td>
</tr>
<tr>
<td>5.3</td>
<td>MCA Functionality Flowchart</td>
<td>79</td>
</tr>
<tr>
<td>5.4</td>
<td>MCA create metadata page</td>
<td>80</td>
</tr>
<tr>
<td>5.5</td>
<td>Screen shut of MCA metadata uploading page</td>
<td>81</td>
</tr>
<tr>
<td>5.6</td>
<td>MCA Metadata translation page</td>
<td>81</td>
</tr>
<tr>
<td>6.1</td>
<td>Upload Metadata section function illustration</td>
<td>90</td>
</tr>
<tr>
<td>6.2</td>
<td>Ajax date picker</td>
<td>92</td>
</tr>
<tr>
<td>7.1</td>
<td>Metadata creation and submission</td>
<td>94</td>
</tr>
<tr>
<td>7.2</td>
<td>Metadata creation and submission</td>
<td>95</td>
</tr>
<tr>
<td>7.3</td>
<td>Metadata search</td>
<td>96</td>
</tr>
<tr>
<td>7.4</td>
<td>Metadata creation and submission</td>
<td>96</td>
</tr>
<tr>
<td>7.5</td>
<td>Translation with MCA's web services</td>
<td>98</td>
</tr>
<tr>
<td>7.6</td>
<td>Validate a local resource file</td>
<td>99</td>
</tr>
<tr>
<td>7.7</td>
<td>Validate an online resource file</td>
<td>100</td>
</tr>
<tr>
<td>7.8</td>
<td>Translate a metadata from one standard to a different standard</td>
<td>102</td>
</tr>
<tr>
<td>7.9</td>
<td>Translation result</td>
<td>103</td>
</tr>
</tbody>
</table>
Listings

2.1 Sample ISO 19115 metadata file .................................................. 11
3.1 How to transform between XML formats ................................. 23
3.2 A javascript implementation of the XML translation .................. 24
3.3 A sample XML file ................................................................. 25
3.4 The XSL file to translate cdcatalog.xml ................................. 25
B.1 ISO 19115 dataset metadata sample (excerpt) ...................... 121
B.2 FGDC metadata sample (excerpt) ........................................... 124
Chapter 1

Introduction

This chapter introduces the background for the research of this dissertation, the issues faced by scientists and researchers in the climate change field, and the NFS EPSCoR program, which partially funded this research. First, the chapter presents the problems related to geospatial data sharing. Second, it provides a brief introduction of our proposed solution. Third, this chapter concludes with a short description of the structure of the dissertation and a brief introduction of the contents of remaining chapters.

1.1 The problem

Global warming is a significant issue worldwide. It is a fact that we are living in a warmer world and sea levels are rising. Severe weather like drought, tornadoes, and hurricanes are occurring frequently. Governments and corporations are investing heavily in climate related research and data collection. Complex scientific models and tools have become available. With those tools, scientists and researchers can investigate the phenomena in greater depth. The problem with additional resources is that scientists have to spend the time to find, gather, and convert the data to satisfy the requirements of their computational tools while they could and should spend the time in analyzing results.
“The heterogeneous, distributive, and voluminous nature of many government and corporate data sources impose severe constants on meeting the diverse requirements of users who analyze the data.”[1] Geographic data is being collected and stored at an ever-increasing rate by government agencies like NASA, large corporations like Google, small to medium companies, universities of all sizes, NCCP, and individuals, such as, researchers and scientists. It is a challenge to make those valuable data resources available to a wide range of data consumers. To make data easily discoverable, metadata standards have been established. With metadata, data can be easily indexed and cataloged into clearinghouses and online catalog systems. Data collecting and data producing organizations, especially medium to small size organizations, still face challenges in submitting their data for indexing for the following reasons.

A. **Metadata missing.** They did not have the tools to generate metadata when data is collected.

B. **Invalid metadata.** Metadata files were created manually when or after data is collected and invalid values are introduced.

C. **Different metadata formats needed.** The metadata files are in one metadata standard and the new tools only support a different metadata standard format.

### 1.2 The proposed approach

We propose a solution to address the above mentioned challenges by creating a geospatial Metadata Community Adaptor (MCA) which is essentially a method supported by web-based software tools. Together with the Tri-State Data Hub(TSDH), which is powered by geoportal server, MCA provides an easy-to-use online platform for data providers to submit, validate, and create metadata files to be uploaded into TSDH server. It also provides a metadata translation function to transform a metadata file from one metadata standard into a different metadata standard in the supported list of popular formats. In addition, MCA provides a set of web services for data providers to submit, validate, and translate
metadata files programmatically instead of doing those time consuming and tedious tasks manually. For data consumers, MCA provides a limited search feature to find and purchase or download available resources.

1.3 NSF EPSCoR collaborative research project

This research is partially funded by the NFS EPSCoR project “Collaborative Research: Cyberinfrastructure Developments for the Western Consortium of Idaho, Nevada, and New Mexico.” The goal of the program is to increase sharing of data and information within and across the three states, which will allow researchers to collaborate seamlessly on climate change issues. Data and information will be make available to scientists, decision makers, students, and the public.

1.4 Organization of the dissertation

The rest of the dissertation is organized as follows: Chapter 2, Background: metadata exchange, provides the background information about metadata exchange. It introduces the concept of metadata, metadata standard, and geographical metadata standard. It also provides details of the most popular ISO 19115 metadata standard. The chapter finishes with the descriptions of the Tri-State Metadata Exchange program. Chapter 3, Background: Related technologies, presents the related technological background. It includes web services, XML and XSLT technology. This chapter also details the reasons for choosing the platform for MCA: ASP.NET/C# and MS SQL Server. Chapter 4, Overview of the proposed approach introduces the proposed methodological approach, which includes: details of our motivation, details of the proposed approach, and details of the main components and capabilities. It concludes with a discussion of some related works. Chapter 5, Requirements and design, explains the software requirements and design of MCA. It covers use case modeling with use case diagram, description of each use case, and more details of the key use cases. This includes the requirement specifications,
which list all the functional and non-functional requirements in a tabular format. A requirement traceability matrix is also presented in this section. Chapter 6, User interface aspects, is about user interface considerations that guided our approach. User interface is a significant element for a software project. It begins with user interface concepts, good and bad user interfaces, and the consequences of bad user interfaces. It is followed by details about MCA’s user interface design. Chapter 7, Application scenarios, describes several application scenarios that are available to the users. It also provides details about the web metadata creating process and the usage of the web service to translate metadata standards. Next, Chapter 8, Comparison with related work, compares MCA to several related projects in the industry including DataONE, Geoportal, GeoNetwork Open Source, and ArcGIS. Chapter 9, Future work and conclusions, concludes this dissertation. It describes a number of tasks for future work that would make the MCA toolset more robust and more functional. It describes how and why we chose the topic, problems we addressed and overall achievements of our work. For reference purpose, an appendix section is added next. Appendix A lists the technical terms and abbreviations with their meanings used in the dissertation. Appendix B shows a sample excerpt of ISO 19115 metadata file. The appendices are followed by the bibliography list.
Chapter 2

Background: metadata exchange

We have all heard the phrase “global warming”. The Wikipedia article states that “Global warming is the rise in the average temperature of Earth’s atmosphere and oceans since the late 19th century and its projected continuation. Since the early 20th century, Earth’s mean surface temperature has increased by about 0.8 °C (1.4 °F), with about two-thirds of the increase occurring since 1980 (See Figure 2.1). Warming of the climate system is unequivocal, and scientists are more than 90% certain that it is primarily caused by increasing concentrations of greenhouse gases produced by human activities such as the burning of fossil fuels and deforestation. These findings are recognized by the national science academies of all major industrialized nations.”[14] There are scientists who do not agree. They believe it is normal to have a higher temperature during “summer” than “winter” of our solar system or even the universe.

On the other hand, decision-makers and policy-makers need more “knowledge” so they can establish rules (laws) to address the issue. Global warming is a world wide issue. It affects each and every one of us on the planet. The new rules will affect people’s life and will cost money. So it must be preceded with cautions. Some questions need to be answered first. Is the increasing temperature caused by human activities? What activities? Is it reversible? Scientists can help develop their knowledge by
building models, analyzing data, proving their models. This leads to the next element, data. Do we have the data? Do we have enough useful and reliable data? It could be argued that we do. A staggering amount of data, especially geographic data, is collected and stored everyday and the increasing rate at which data is being collected is overwhelming. In the meantime, scientists have developed complex models that can analyze and manipulate larger, richer, finely-grained datasets from multiple heterogeneous sources than ever before. Researchers and scientists are able to investigate phenomena in great depth. One of the common problem researchers and scientists face is how to find and integrate heterogeneous domain-specific data into a format that meets their own needs. [41, 42, 43]

Researchers and scientists are highly trained professionals who can analyze data and transform data into knowledge for decision-makers. Based on the knowledge provided by scientists, decision-makers can make sound policies. That would mean that researchers and scientists should spend their valuable time and resources analyzing data. But, in reality, they need to spend a great deal of time locating useful data, converting it from one format into another format to meet their needs while they could use the time more productively in analyzing those data. [44]

Because companies and organizations use different equipment to collect data and use different software to store their data, researchers cannot easily find the data they seek. A great amount of data remains untapped. Another scenario is that researchers invest heavily in equipment and software to collect data for a project. If there is not a system to make those data discoverable, it will be wasted while it could be reused by more data consumers.

A dataset itself is not useful without the data about the dataset. It is like a backup tape, it is a piece of junk without a proper label. Metadata is used to describe data and provides a way to comprehensively describe datasets. It is widely used by data catalogues and clearinghouses. Various metadata standards have been developed by different organizations for different data categories. Due to the coexistence
of various metadata standards, making full use of metadata in different metadata standards becomes a problem. This dissertation addresses the issues pertaining to multiple metadata standards by proposing a new approach and providing a set of online web services to help manipulate metadata created in different geographic metadata standards. [25]

Figure 2.1: Global mean land-ocean temperature change from 1880–2012, relative to the 1951–1980 mean. The black line is the annual mean and the red line is the 5-year running mean. The green bars show uncertainty estimates[2]. Source: NASA GISS
2.1 Metadata

Metadata is commonly defined as “data about data”, “structured data about data”, or “data which describes attributes of a resource”, or even “information about data”. Metadata provides information about one or more aspects of the data, such as purpose of the data, means of creation of the data, owner or author of the data, point of contact of the data, etc.

Metadata is not a new concept. Business cards and library cards are examples of metadata in our everyday lives. Metadata is often used for photographs (IPTC Schema, XMP, Exif, PLUS, etc.), videos (transcript, text description of the scenes), and webpages (keywords, description, software used to create the page, author of the page).

The objective of metadata is to provide more details about a dataset. It has been widely used in dataset cataloguing and clearinghouse activities so data users can locate the data source more easily and accurately.

Metadata can be embedded inside the data file or stored in a separate document. Furthermore, metadata is commonly used by computers and software, rather than humans. Metadata can be stored internally, in the same file as the data, or externally, in a separate file.

2.2 Metadata standards

In the information age, the amount of data, especially geographic data, is exploding. Many companies produce and collect geodata in various formats from different sources with different equipment. The result of this heterogeneous geodata creates a major problem for data sharing. Geographic metadata describes the existing geodata. By reading the geo-metadata, users can get more information about the original dataset, like name, quality, ratio, data structure, etc. [3]. Because most geodata is large in size,
written in various (standard) formats and stored in different file formats, it is hard to directly access the original dataset. Thus, it is necessary to utilize the geodata to not only describe and catalogue data, but also to discover, convert, manage, and use data in a network [4].

The more widely used geographic metadata standards include CSDGM by FGDC [5] and ISO 19115 by ISO/TC211 [6]. There are many differences between metadata standards. For example, the original version of DC (Dublin Core) defined only 15 elements known as the original set of 15 classic metadata set, while the ISO 19115 has more than 300 elements, organized in 86 classes with 282 attributes and 56 relations.

Metadata standards are usually presented using a structural file. This concept is useful for standards with many defined elements, but it is hard to analyze and process. There is also not a single metadata definition language. Therefore, different standards are presented using different notations. For example, the ISO 19115 uses UML while the CSDGM uses a formal file notation [7]. Finding a common way to present different metadata standards will allow computers to automatically recognize, analyze, and share geo-data in different metadata standards. To this end, XML is a popular markup language that can define, present, verify, and index metadata [8].

As described in [24], “metadata standards are requirements which are intended to establish a common understanding of the meaning or semantics of the data, to ensure correct and proper use and interpretation of the data by its owners and users.” A metadata standard is usually established by national and international standard communities like ANSI [10](American National Standards Institute) and ISO [11](International Organization for Standard).
2.3 Geographical metadata standards

In the geographic domain, the most common metadata standards are FGDC’s Content Standard for Digital Geospatial Metadata (CSDGM) and the recently ratified ISO 19115 [7]. The CSDGM standard contains over 300 data and compound elements while the ISO 19115 has over 400 elements (divided into 14 metadata packages) in 86 classes that have 282 attributes and 56 relations. The ISO 19115 was developed by the geospatial community to address specific issues relating to both the description and the curation of spatial data[6]. The ISO 19115’s abstract models are written using the UML (Unified Modeling Language). The accompanying XML schema, ISO/CD TS 19139, enables interoperable XML expression of ISO 19115 compliant metadata. [26, 27]
Listing 2.1: Sample ISO 19115 metadata file
2.4 ISO 19115

The following is a definition of ISO 19115 from the ISO website:

“ISO 19115:2003 defines the schema required for describing geographic information and services. It provides information about the identification, the extent, the quality, the spatial and temporal schema, spatial reference, and distribution of digital geographic data.

ISO 19115:2003 is applicable to:

- the cataloguing of datasets, clearinghouse activities, and the full description of datasets;
- geographic datasets, dataset series, and individual geographic features and feature properties.

ISO 19115:2003 defines:

- mandatory and conditional metadata sections, metadata entities, and metadata elements;
- the minimum set of metadata required to serve the full range of metadata applications (data discovery, determining data fitness for use, data access, data transfer, and use of digital data);
- optional metadata elements - to allow for a more extensive standard description of geographic data, if required;
- a method for extending metadata to fit specialized needs.

Though ISO 19115:2003 is applicable to digital data, its principles can be extended to many other forms of geographic data such as maps, charts, and textual documents as well as non-geographic data.”

The ISO 19115 standard is part of the ISO 19000 series (ISO Geographic Information Suite of Standards). It defines how to describe geographic information and services, including contents, spatial-temporal purchases, data quality and right of use. The objective of ISO 19115 is to provide a clear procedure to describe digital geographic datasets so that users can determine whether the data is useful and how to access it.
2.5 Tri-State metadata exchange

This effort is part of the metadata exchange project, which is, itself, part of Track II of the NSF EPSCoR funded project “Collaborative Research: Cyberinfrastructure Developments for the Western Consortium of Idaho, Nevada, and New Mexico”. On September 2008, NSHE (Nevada System of Higher Education) was awarded $15 million by NSF EPSoR over five years to develop science, education, and outreach infrastructure at UNR, UNLV, DRI (Desert Research Institute), NSL (Nevada Seismological Laboratory), and NSHE community colleges for the study of climate change and its effects on Nevada. The Nevada Climate Change Project now comprises a web portal, the SENSOR data system (software, hardware, and database), and a high-speed TCP/IP network infrastructure.

The NCCP (Nevada Climate Change Portal) website (http://sensor.nevada.edu) provides information to project members, researchers, and the public [23]. Via search interfaces and web services, users can search and download data collected from SENSOR data. The web portal also provides real-time videos and photos from monitoring sites. The SENSOR data collection system comprises numerous Campbell Scientific data loggers (CR1000 and CR3000), each with dozens of physical sensors that collect thousands of measurements per minute. Through a secure virtual private network (VPN) of the Nevada Seismological Laboratory [23], the data loggers transport collected data to the data center via the lossless TCP/IP protocol [20]. Then the data is stored on file servers and imported into a SQL server. By design, the SENSOR system uses a standards-neutral database schema, meaning that the database structure is not modeled after a specific metadata standard like FGDC or ISO 19115 [20] – the system simply collects more information than is needed by any one standard. A key purpose of this work is to provide a system to transform raw data to a chosen metadata standard.

As mentioned before, the NSF EPSCoR-funded project involves three western states: Nevada,
Idaho, and New Mexico. Idaho and New Mexico have their own data centers or data repositories implementing particular metadata standards. This creates a problem: how can we effectively share data between data centers? One solution is to build a central clearinghouse to which each data center submits metadata for cataloguing [12]. Users can then search the clearinghouse for geographic data from all participating data centers. Each participant web portal can query the clearinghouse for data too.

This dissertation proposes a practical mechanism to transform between different metadata standards. With this service, the clearinghouse will be able to handle data submissions accompanied by differing metadata standards, allowing each data center to utilize its own metadata standard, such as ISO 19115. Specifically, in this dissertation we propose to create a metadata community adaptor that will transform geographic metadata in different standards to the ISO 19115 metadata standard format and vice versa. More details about the proposed work are presented in Chapter 4.
Chapter 3

Background: Related technologies

The proposed Metadata Community Adaptor (MCA) will be implemented as a set of web services on a Windows web server. It will utilize some of the newest Internet technologies, such as web service, XML, and XSLT. In terms of development, it will be built using Microsoft Visual Studio with ASP.NET and C# support. The Microsoft SQL Server will be used for database support. This chapter presents background information about these related technologies. [28, 29, 30]

3.1 Web service

Web service is a popular Internet technology. It is a low cost communication method using a standardized protocol to achieve application interoperability. Specifically, it is an interface between application code and the user. Notably, interoperability is one of the key benefits of web services.
Table 3.1: The web service Technology Stack

| Discovery | Description | Packaging | Transport | Network |

3.1.1 What is a web service?

A web service is a communication method between two electronic devices over the Internet. The W3C definition of the web service is: “a software system designed to support interoperable machine-to-machine interaction over a network.” A web service uses WSDL (Web Service Description Language) to describe itself. The WSDL is in a machine-processable format.

3.1.2 Classes of web services

According to W3C, there are 2 major classes of web services, the REST-compliant web services and the arbitrary web services. RESTful web services’ primary purpose is the manipulation of XML representations of web resources using a uniform set of “stateless” operations while the arbitrary web services may expose an arbitrary set of operations.

3.1.3 Web service architecture

Figure 3.1 depicts the typical web service architecture. In general the web service architecture is implemented through the layering of five types of technologies. These are organized into layers and build upon one another, as shown in Table 3.1
3.2 XML

3.2.1 What is XML?

The Extensible Markup Language (XML) is an open standard markup language that defines a set of rules for encoding documents in a format that is readable by both humans and machines. Most importantly, XML is extensible, one can make up one’s own tags and uses tag-based syntax. XML is the foundation for the next generation web technologies like XHTML, RSS, Ajax, and web services. Nevertheless, It is not a replacement for HTML.
3.2.2 What does XML do?

XML is used to structure and describe information; specifically, it provides a way to exchange information between different systems over the Internet. XML is designed to store, retrieve, and transport data. XML files are text files so the notation is platform-independent. That makes it a perfect choice for data transportation between machines or over the Internet. Furthermore, it is meant for small files rather than large files. Many software applications use XML for preference settings, look-up tables, work files etc. For example, a calendar application can use an XML file to store holidays instead of hard-coding them inside the source code. The end users can add new, personal holidays, change or delete existing holidays by simply editing the holiday XML file after the application is compiled. The holiday can be categorized using the XML entity attributes and the application can decide which holidays to include in a calendar.

XML only describes the data, it does not include presentational instructions. There are a number of ways for formatting an XML file. The easiest way is to use CSS (cascading style sheet). CSS works the same way as HTML. The XML file includes a reference to a CSS file. A CSS compliant browser will format the XML according to the instruction stored in the CSS file. CSS is the simplest way of formatting XML, and works similarly to “find” & “replace”.

The XSLT technology provides a powerful and more flexible and versatile technique to format XML files. XSLT transformation is a process that breaks apart a XML file and rebuilds a completely different file format. The XSLT processor takes a XML file and XSL (eXtensible Stylesheet Language) as inputs and transforms the XML file according to the instruction provided in the XSL file and produces a new output file. Together with XPath and Xpointer, XSLT provides more powerful programming features such as search, sort, aggregation (calculation) and so on. XSLT is heavily used in this project.
While XML is very useful, there are some drawbacks as well. While XML offers great power of flexibility, platform independence, and easy transport, it must ensure data integrity. XML files can be corrupted, truncated, incomplete, or mistyped and a corrupted data file can cause unexpected consequences. XML offers different mechanisms to ensure XML document integrity. The first technique is to check for well-formedness, which means that it basically checks for missing tags, malformed tags, illegal characters, and so on. An XML document must be well-formed, meaning that the document adheres to the syntax rules specified by the XML 1.0 specification. Well-formedness requirements include: content be defined, content be defined with a beginning and end tags, and content be properly nested (each XML document must contain one and only one root element, parents within root and children within parents) and so on. The well-formedness is ensured by XML parsers. The XML parser must report any problems to users.

A well-formed XML only guarantees that it is grammatically correct, not necessarily a valid XML document. Most applications require the validation test. A valid XML document is a well-formed XML document that also passes the validation test. For a validation test, XML authors supply a DTD or schema for the XML document. The XML parser validates the XML file against the DTD or schema. The validation rules can be embedded inside the XML document or stored in a external file and provide the reference link inside the XML. Example of the validation rules include: what entities are allowed, number of children, allowed values for a particular entity. An example of XML file is shown in Figure 3.2.2.

![XML](image)

Figure 3.2: A sample XML file
3.2.3 **XML-related technologies**

As mentioned previously, XML is the foundation of the next generation of the Internet. XML provides a way to structure information. To present, search, and modify the information on the web, other related technologies have been developed. Here is a list of some of the related technologies: [45, 47]

- Xpath – Extensive path language, which uses path-like syntax.
- XSLT – styling language
- XQuery – used to query XML data such as SQL
- Xpointer and Xlink, which are used to create hyperlinks to XML documents, and arbitrary points within XML documents

3.3 **XSLT**

3.3.1 **What is XSLT?**

As we mentioned before, XML is a popular, powerful, and flexible mark-up language used to store and retrieve data. Its flexibility provides the power to develop various vocabularies. This presents the need to be able to transform documents marked up in one XML vocabulary to another.

The XSLT (eXtensible Stylesheet Language Transformation), along with XPath (XML Path Language), meet that need. The XSLT provides a powerful implementation of tree-oriented transformation language that transfers an XML instance using one vocabulary into either another XML document in a different vocabulary or a non-XML document like text file. The XSL (eXtensible Stylesheet Language) is used to specify the instructions for a XSLT processor to generate the desired output from the given input XML document. Notably, XSLT enables and empowers interoperability among XMLs in different domains. [46, 48]
3.3.2 Logical components of an XSLT application

The XSLT transformation is carried out by a XSLT processing software (XSLT Engine). The XSLT Engine takes an XML document and XSL document as inputs and generates the desired output file(s) according to the instructions provided within the XSL input document. Figure 3.3 depicts this process. [49, 50]

![Diagram showing XSLT processor, XML, and XSL documents.](image)

Figure 3.3: An XSLT processor takes one or more XML documents and one or more XSLT files as input and produces a new XML document

Specifically, the first component is the input XML file. The input XML document is a text file that was made of a sequence of data characters and markup tags. Data elements are delimited with start-tags and end-tags. The XSLT engine does not work on the XML input file directly. It builds a tree based on the XML input file and works on the tree. [51, 52, 53]

The second component is the XSLT stylesheet (XSL file). An XSL file is a well-formed XML document. It acts like a computer program that includes all the transformation instructions. It is called
a stylesheet file but it is totally different from the HTML stylesheet (CSS) file. The XSL file does not have “style” information but instead contains “transformation” information, and thus should be called a transformation file. [34, 55] The XSL file contains a series of template rules. Each rule is a sequence of XSLT commands and each command is an XML element with attributes. A rule is executed when it matches some conditions or is called by name. [34, 35]

Another component of an XSLT application is the XSLT engine. There exists several ways to perform a transformation. For stand-alone XSLT engines, users download the application (many free open source XSLT processors) and execute a transformation using a command line like: [36, 37]

```
Qiping > Saxon [input XML] [input XSL] [output file]
```

Popular XSLT processors include:

- Saxon (http://users.iclway.co.uk/mhkay/saxon/)
- C Xalan XSLT (http://xml.apache.org/xalan/index.html)
- C Unicorn XSLT Processor (http://www.unicorn-enterprises.com/)
- C XSLT C library for Gnome (http://xmlsoft.org/XSLT/)

In a web environment, an XSLT transformation can occur in the web server or on the client side. All the major web browsers (recent versions) support XSLT. Or, inside a webpage, Javascript can be used to open and transform a XML file.

All major browsers have support for XML and XSLT, as shown in Table 3.2.
Table 3.2: Browser support for XML and XSLT [56]

<table>
<thead>
<tr>
<th>Browser</th>
<th>Support</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mozilla Firefox</td>
<td>Firefox supports XML, XSLT, and XPath from version 3.</td>
</tr>
<tr>
<td>Internet Explorer</td>
<td>Internet Explorer supports XML, XSLT, and XPath from version 6. Internet Explorer 5 is NOT compatible with the official W3C XSL Recommendation.</td>
</tr>
<tr>
<td>Google Chrome</td>
<td>Chrome supports XML, XSLT, and XPath from version 1.</td>
</tr>
<tr>
<td>Opera</td>
<td>Opera supports XML, XSLT, and XPath from version 9. Opera 8 supports only XML + CSS.</td>
</tr>
<tr>
<td>Apple Safari</td>
<td>Safari supports XML and XSLT from version 3.</td>
</tr>
</tbody>
</table>

Another way to transform XML files is in the server, using frameworks such as ASP.NET. Listings 3.1 to 3.4 present a sample transformation in ASP. [38]

ASP XSLT Transformation

```csharp
<%'
'Load XML
set xml = Server.CreateObject("Microsoft.XMLDOM")
xml.async = false
xml.load(Server.MapPath("cdcatalog.xml"))

'Load XSL
set xsl = Server.CreateObject("Microsoft.XMLDOM")
xsl.async = false
xsl.load(Server.MapPath("cdcatalog.xsl"))

'Transform file
Response.Write(xml.transformNode(xsl))
%>
```
Listing 3.1: How to transform between XML formats

Javascript implementation

```html
<html>
<head>
<script>

function displayResult ()
{
xml=loadXMLDoc("cdcatalog.xml");
xsl=loadXMLDoc("cdcatalog.xsl");
// code for IE
if (window.ActiveXObject)
{
ex=xsl.transformNode (xml);
document.getElementById("example").innerHTML=ex;
}
// code for Mozilla, Firefox, Opera, etc.
else if (document.implementation &
document.implementation.createDocument)
{
  xsltProcessor=new XSLTProcessor();
xsltProcessor.importStylesheet(xsl);
resultDocument = xsltProcessor.transformToFragment (xml, document);
document.getElementById("example").appendChild(resultDocument);
}

</script>
</head>
<body onunload="displayResult ()">
<div id="example" />
</body>
</html>
```
Listing 3.2: A javascript implementation of the XML translation

XML document: cdcatalog.xml

```xml
<?xml version="1.0" encoding="ISO-8859-1"?>
<catalog>
  <cd>
    <title>Empire Burlesque</title>
    <artist>Bob Dylan</artist>
    <country>USA</country>
    <company>Columbia</company>
    <price>10.90</price>
    <year>1985</year>
  </cd>
</catalog>
```

Listing 3.3: A sample XML file

XSL document: cdcatalog.xsl

```xml
<?xml version="1.0" encoding="ISO-8859-1"?>
<xsl:stylesheet version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:template match="/">
    <html>
      <body>
        <h2>My CD Collection</h2>
        <table border="1">
          <tr bgcolor="#9accd3">
            <th align="left">Title</th>
          </tr>
```


<th align="left">Artist</th>
</tr>
<xsl:for-each select="catalog/cd">
<tr>
<td><xsl:value-of select="title" /></td>
<td><xsl:value-of select="artist" /></td>
</tr>
</xsl:for-each>
</table>
</body>
</html>
</xsl:template>
</xsl:stylesheet>

Listing 3.4: The XSL file to translate cdcatalog.xml

The last component of the XSLT application is the output file. An XSLT transformation can produce one or more output files in the format of XML, HTML, or text (untagged files).

XSLT is a popular technology, many businesses are using it in some degree because it is easy, cheap, fast. However, it is not good at everything, for example, it cannot convert something else into an XML file. Also, it is not good with non-XML data like Word documents, it is not as good as programming languages for computation, and it is not suited for large data files. Furthermore, it is weak on string manipulation. [39, 40]

3.4 ASP.NET/C#

3.4.1 What is ASP.NET?

ASP.NET is a server-side web application development framework from Microsoft. It is a popular framework on the Windows platform to build dynamic websites, web applications, and web services.
Microsoft released the 1.0 .NET framework in January 2002. It is the successor of Microsoft’s Active Server Pages (ASP) technology. ASP.NET is built on the Common Language Runtime (CLR) to allow programmers to write ASP.NET code in any supported languages like Visual Basic, C#. One of the fundamental advances in ASP.NET over ASP is the separation of presentation from control. With ASP, a dynamic web page is achieved by including control code (Visual Basic, Javascript) inside the HTML page. For ASP.NET, Microsoft introduced the “code-behind” model to separate a webpage’s formatting from its logical control. Each webpage (.aspx) has a corresponding “code-behind” page (.aspx.cs) to handle programming logic to control the page behavior.

### 3.4.2 ASP.NET fundamentals

ASP.NET introduced an entirely new object-oriented execution model. ASP.NET execution centers around CLR classes that implement an interface named IHttpHandler.

Upon arrival on an IIS webserver, the HTTP request/response is routed through many server side objects for processing. When a user hits the Return key after entering a URL, the web browser sends a GET request to the target web server. The request travels through various routers and reaches the web server on port 80. The IIS on the server handles the request. The IIS intercepts the request and maps it to the worker process and the request follows a very specific path through the pipeline. The path is different depending on the IIS version. For IIS 7.X, it works as follows:

A. The browser makes a request for a resource on the Web server.
B. HTTP.SYS picks up the request on the server.
C. HTTP.SYS uses the WAS to find configuration information to pass on to the WWW Service.
D. WAS passes the configuration information to the WWW Service, which configures HTTP.SYS.
E. WAS starts a worker process in the application pool for which the request was destined.
F. The worker process processes the request and returns the response to HTTP.SYS.
G. HTTP.SYS sends the response to the client.
The ASP.NET and IIS 7.x architecture is shown in Figure 3.4.

![ASP.NET and IIS 7.x Architecture Diagram](image)

**Figure 3.4: ASP.NET and IIS 7.x architecture**

### 3.4.3 Event-driven application

Most of the modern popular web development platforms, like PHP, ASP, and Cold Fusion, support the OOP (Object Oriented Programming) concept. OOP is effective in keeping web application code well organized, modular, and scalable, especially for large, complex projects. [32, 33]

EDP (Event Driven Programming), on the other hand, is less common among web development platforms. ASP.NET is the one that supports both OOP and EDP. It provides an unmatched level of developer productivity on complex projects. Before EDP, most programs were procedural. A procedural program is like a list of instructions. The computer executes those instructions stored in a program one by one with some conditional branches dependent upon user input or the result of previous action. A procedural program works fine for simple console applications, but it is not enough for the modern GUI.
(Graphical User Interface) type of applications. Instead of executing instructions, an EDP program does not do anything unless “something” happened. That “something” is called an event. An event driven program responds to events. Events can be generated or fired by a user action or by the system (like file is loaded). In ASP.NET, objects may raise events and may have assigned event handlers. The event handlers respond to events by executing the code inside the event handler. ASP.NET names event handlers by prepending the work “On” to the event name. For example, “OnClick” for the Click event.

Table 3.3 shows a list of common events and event handler names.

<table>
<thead>
<tr>
<th>Event name</th>
<th>Event handler name</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td>BubbleEvent</td>
<td>OnBubbleEvent</td>
<td>All controls</td>
</tr>
<tr>
<td>CheckedChanged</td>
<td>OnCheckedChanged</td>
<td>CheckBox</td>
</tr>
<tr>
<td>Click</td>
<td>OnClick</td>
<td>Button, LinkButton, ImageButton</td>
</tr>
<tr>
<td>DataBinding</td>
<td>OnDataBinding</td>
<td>All controls</td>
</tr>
<tr>
<td>Init</td>
<td>OnInit</td>
<td>All controls</td>
</tr>
<tr>
<td>ItemCreated</td>
<td>OnItemCreated</td>
<td>Repeater</td>
</tr>
<tr>
<td>ItemDataBound</td>
<td>OnItemDataBound</td>
<td>Repeater</td>
</tr>
<tr>
<td>Load</td>
<td>OnLoad</td>
<td>All controls</td>
</tr>
<tr>
<td>PreRender</td>
<td>OnPreRender</td>
<td>All controls</td>
</tr>
<tr>
<td>SelectedIndexChanged</td>
<td>OnSelectedIndexChanged</td>
<td>DataGrid, DataList, CheckBoxList, DropDownList, ListBox, RadioButtonList</td>
</tr>
<tr>
<td>TextChanged</td>
<td>OnTextChanged</td>
<td>TextBox</td>
</tr>
<tr>
<td>Unload</td>
<td>OnUnload</td>
<td>All controls</td>
</tr>
</tbody>
</table>

### 3.4.4 ASP.NET server controls

In order for the ASP.NET code behind to communicate with the HTML controls in an ASP.NET page, the controls must be enclosed inside the `<form>` tag. ASP.NET `<form>` tag is different from the regular HTML `<form>` tag. Any HTML controls can be placed inside the ASP.NET `<form>` tag. It is like a regular HTML web page for the web browser to format and display the page. In order for the behind code to retrieve and set the controls’ attributes or value, the “RunAt” attribute must be set to “Server.” Then the code behind can access the controls by referencing to controls by ID.
3.4.5 Web service support in ASP.NET

As mentioned in Section 3.1, a web service includes providing and consuming web service. By providing web service, I mean that when a user sends a web service request to a web site or a web application in a web server, the web application processes the request (for example, fetch data from the database, invoke another application, do the calculation, format the result and send the result back to user). By consuming web service, we mean that we construct a web service request, manually or automatically by a program, send the request to a different web server or to the same web server, including maybe in the same website. When the reply arrives the application processes or parses the SOAP message. With the information, the application formats it as a new web service response or as a web page and sends it to the user. It is logically easier to “provide web service” than to “consume web service” since we have to deal with various message structures when consuming web services.

The concept of web service is fairly simple and easy to understand. It is an application that can send data as XML conforming to the SOAP specification. It is straightforward to build a web application capable of receiving SOAP messages over the HTTP and delivering SOAP messages out of it. The ASP.NET just makes it easier for web application developers to provide/use web services by providing the framework to do the “plumbing” work. The developers can focus on the application logic, instead of figuring out how HTTP, SOAP, WSDL or any of the technologies that form the basis for web service work. The Microsoft Visual Studio .NET provides a rich feature set for programmers to easily develop well-documented, scalable web service applications.

3.5 SQL Server

The Microsoft SQL Server is an RDMS (Relational Database Management System) developed by Microsoft. It is one of the popular RDMS in the market (Other popular RDMS's include Oracle, IBM's DB2, and MySQL). The newest version is SQL Server 2012 that is cloud-ready. [57] Microsoft SQL
Server is used by small, medium, and large enterprises. One of its advantages is that the MS SQL Server encompasses more than just a relational database engine, it includes business intelligence, a complex event processing engine, highly scalable data warehousing solutions, and a SQL Server edition running in the cloud. [38]

Microsoft offers different editions for different clients. SQL Server editions include: Enterprise, Business Intelligence, Standard, Web, Express, Compact, and SQL Azure. All the SQL Server editions can only run in a Microsoft Windows system. The Enterprise, Business Intelligence, and Web editions only can be installed on a Windows Server platform while the rest of the editions can also be installed on Windows systems like Windows 8, 7, and Vista. [39, 60]

Microsoft breaks down its SQL Server editions in almost the same way as Oracle, with Express, Standard, and Enterprise editions being the main products. But Microsoft does not offer additional options that can be purchased separately as Oracle does, such as Partitioning, Spatial, OLAP, and Data Mining. Instead, it bundles all its features into the base product. The available features depend on the editions purchased.
Chapter 4

Overview of the MCA approach

4.1 Motivation

As shown in Chapter 2, working with metadata is a complex and demanding task. This task is further complicated by the co-existence of various metadata standards which, in the geospatial domain, slows down data exchange and utilization. [31] The work presented in this dissertation is aimed at addressing issues pertaining to using metadata and metadata standards. In terms of related areas of research, the following can be considered interesting directions of exploration (these have resulted from a discussion with Dr. Richard Kelley, who is a member of the Nevada cyberinfrastructure group involved in metadata exchange with New Mexico and Idaho):

Generating metadata

One problem that scientists and data managers have right now is that they tend to have a great deal of data stored, but not used; they may not have much (or any) metadata that describes their data. Although tools can be developed that extract descriptions from data directly, explicitly represented metadata makes it easier to share data and use data provided by others. When researchers begin to collect data without any thought for collecting and/or curating metadata, the result is that after several years it
becomes difficult and costly to generate the metadata for a data collection. The cost of generating such metadata then becomes an impediment to further collaboration with the broader scientific community. If there were a tool that researchers and data managers could use to easily generate large volumes of metadata, that would solve a problem that many people have.

**Repairing metadata**

Lacking metadata isn't the worst case. In some situations, researchers have made an attempt to generate and track metadata associated with their data, but may have succeeded only partially. Due to a variety of factors, a data repository may contain metadata files that are incomplete or incorrect. Sometimes, it may be that metadata errors can be detected by automated tools. An example class of such errors deals with XML validation - if an XML file containing metadata doesn't validate against a known schema (such as ISO 19115), validation errors can be caught automatically. More subtle semantic errors require human involvement. If there were a tool that eased the process of repairing valid but incorrect XML metadata, that would be useful for researchers and data managers.

**Translation across standards**

One of the most difficult things about metadata is that there are many different metadata standards to choose from. Many of these standards are old or obsolete, and as our understanding of metadata management improves it would be nice to have a tool that accepts as input a file containing metadata conforming to one standard (say, the Dublin Core standard) and produces as output a file containing metadata conforming to a different standard (e.g., ISO 19115). Of course, the problem is that this is a \(O(n^2)\) development task - developing a custom translator for every pair of metadata standards would be time-consuming. A possible solution is to automate that process by generating a program that generates programs that do the conversion. Ideally, this converter generator would exploit compositionality and transitivity to reduce the complexity of the problem of generating converters. However, this likely is itself a complicated topic to tackle, so we need to look for other solutions that could facilitate translat-
ing from one metadata standard to another.

Based on the above, we propose to address some of the issues pertaining to using metadata and metadata standards, in particular to metadata generation, metadata validation (not full repair), and metadata conversion across standards. Because the standard ISO 19115 2 is a modern, popular and most comprehensive geospatial metadata standard, most of the presented approach is centered on this standard, without losing track of applicability to other standards as well.

4.2 The MCA approach

The high-level description of the approach is as follows: In Figure 4.1, the Tri-State Data Hub is an online geospatial catalog system built from the open source Geoportal Server package with some customization to better suit for the NCCP, New Mexico, and Idaho climate change portal to share data. It also serves as the central portal for geospatial data users to search for datasets from all the three state data portals. The geoportal server is implemented in Java. Its main function is geospatial cataloging.

The MCA (Metadata Community Adaptor), which is at the core of the work, serves as a bridge layer between Tri-State Data Hub and the three data portals from each state, Nevada, New Mexico, and Idaho. Its main functions include new metadata creation, pre-submitted resources management, and metadata standard translation. MCA will be able to run as a separate website so each state portal performs the above functions using the online web form or web services provided by MCA. Alternatively, each state portal can install MCA functions as a sharing web component in its own server and perform functions within the portal and submit metadata directly to the TSDH(Tri-State Data Hub).

The state portals only submit metadata to TSDH so users can find the data they seek. To access an actual dataset, the user will be instructed to the download URL or purchase method. The dataset will be kept in the state portals. The system architecture is depicted in Figure 4.2 and the software
Figure 4.1: Tri-State Geospatial Metadata Exchange
flowchart is depicted in Figure 4.3.

As shown in Figure 4.2, the architecture of MCA is multi-tiered, having several important layers, specifically: Web Interface, Metadata Tool Component, Web Service Component, and Database Component. In addition, the XSLT Component provides the connection between the Metadata Tool Component and the Web Service Component.

Figure 4.3 presents the generic software workflow chart for the translation part of MCA. The request can be a web form request or a script-generated web service request. The response is always an XML document in a metadata standard format.

Because ISO 19115 is the main standard used in MCA, Figures 4.4 and 4.5 are included to provide a glimpse of the standard's details and complexity. The figures show only part of the ISO 19115-2 UML model.

At a lower level description of the approach, MCA will provide web service-based facilities and resources for metadata search, upload, creation (generation), validation, and translation.

A workflow describing the way users can follow the proposed methodological approach is shown in Figures 4.6. This figure and the following supporting figures (Figures 4.7 - Figures 4.11) reveal that the major capabilities are as follows:

A. Login
B. Select public info activity
C. Select member-based info activity
D. Upload metadata
E. Create metadata
F. Validate metadata
G. Translate metadata
H. Metadata management
Figure 4.2: MCA architecture
MCA Software Workflow

Figure 4.3: MCA software flowchart
Figure 4.4: ISO 19115 metadata standard entity UML diagram
Identification information

Figure 4.5: ISO 19115 metadata standard identification UML diagram
I. User account management
J. Community contribution

The novelty of the proposed approach consists of combining processing capabilities for metadata that, to the best of our knowledge, none of the existing metadata-focused applications provide at the same time. In particular, very few related efforts exist that are focused on metadata translation. Specifically, MCA will address the metadata generation issue described in Section 4.1, provides capabilities for metadata validation, and will offer flexible translation capabilities across geospatial metadata standards. Notably, all these will be built using modern web technologies, and will be available for the community of users who will be able to contribute their own work to the MCA environment.

Regarding translation, one of the important aspects of metadata standards is that there are so many standards to choose. That makes it particularly difficult to share valuable related resources. So, the availability of a web service that can translate between different metadata standards (automatically) would be useful. But geospatial metadata is generally complicated in nature. That makes developing a fully automatic geospatial metadata translator extremely challenging. During research for the work, although I read papers that mentioned this topic, I never encountered an application or service that solves this problem. For example, ESRI claims to offer two programs, Esri2EML and BDP2EML, which translate metadata from ESRI or FGDC and from BDP (Biological Data Profile) to EML (Ecological Metadata Language). That is a limited solution, as it offers only a one-way (to EML) translation and to a single target metadata standard (EML).

Another problem is that new standards are proposed occasionally. That makes having a “general purpose” geospatial metadata translator quite necessary. However, to develop a metadata translator that fully translates between all possible metadata standards seems impossible or near-impossible. Thus, in terms of the translation capability of MCA, we propose a semi-automated metadata translator which will translate between “any” two geospatial metadata standards with some user interactions. The user will provide the source metadata document in one standard and MCA will upload the document and
Figure 4.6: The MCA Approach: Top level workflow chart
Figure 4.7: The MCA Approach: Create metadata activity
Figure 4.8: The MCA Approach: Validate metadata activity
Figure 4.9: The MCA Approach: Upload metadata activity
Figure 4.10: The MCA Approach: Community contribution activity
Figure 4.11: The MCA Approach: Translate metadata activity
scan it. Then MCA will compare the metadata to the ISO 19115 requirements and respond with a web form request for the missing entities and elements. The user will then have to enter those missing pieces of information manually. As a result, MCA will form the complete ISO 19115 metadata document. Conversely, MCA will be able translate the ISO 19115 document into the target metadata standard.

MCA has several metadata processing capabilities available, as indicated next. A brief overview of MCA is provided through the use of snapshots of its user interface.

Specifically, Figure 4.12 displays the home page of MCA. As shown in this figure, the main options and capabilities available to the user are: registration, upload, create, and translate metadata, manage metadata, and search and download meta(data).

In order to cater to a community of users, MCA keeps track of its registered clients. The login page for registration and log in is shown in Figure 4.13. Brief information on MCA is available on the “About” page of the environment, as shown in Figure 4.14. In essence, the user is informed here that MCA attempts to address the interoperability issue among different metadata standards.

Searching for metadata on the Geoportal server is implemented in MCA, as shown in Figure 4.15. Sample results of searching for data are shown in Figure 4.16.

Complementary to searching, MCA supports uploading of metadata, as shown in Figure 4.17. The page displayed in the figure also enables a user to choose a specific metadata file, upload it, and validate it against its standard’s specifications.

Part of the metadata processing capabilities provided by the MCA toolset, creating (or generating) metadata is likely to be very useful to researchers relying on geospatial data. Figure 4.18 shows the interface for creating metadata in ISO 19115 standard. The user has the option to upload an existing
local metadata file or manually create metadata.

The most significant component of MCA, metadata translation, is illustrated in Figure 4.19. While at this time translation between metadata standards is not yet implemented, the figure shows how a user can search for web services that could provide such translations. Associated to this capability, Figure 4.20 presents the MCA test page for metadata translation, which allows the user to select a source and a target metadata standard for the translation.

The MCA toolset seeks to enable a pleasant and effective user experience. For this purpose, attention has been paid to the details of MCA's interface. For illustration purposes, Figure 4.21 presents a map viewer interface that facilitates the user's exploration of available metadata resources. Finally, Figure 4.22 depicts details of a project specific Tri-State Data Hub file.

4.3 Similar work and discussion

For comparison purposes, a short survey of related resources has been performed, indicating that not much has been done in fully integrating metadata processing capabilities, especially in terms of providing support for metadata translation among standards (this survey will be expanded, part of the planned feature-based comparison with related work). Note that in Chapter 8, Comparison with related work, a more detailed survey of other approaches and feature-based comparison are included.

Among the resources surveyed, the GeoNetwork Opensource is a standards-based and decentralized spatial information management system designed to enable access to geo-referenced databases and cartographic products from a variety of data providers through descriptive metadata, enhancing the spatial information exchange and sharing between organizations and their audiences using the capabilities of the Internet[9]. The GeoNetwork Opensource is an open source Java application developed following the principles of Free and Open Source Software (FOSS) and based on the International and
Introduction to MCA

Metadata Community Adaptor is ASP.NET web application that serves both geographical information consumers and producers by providing geospatial metadata standard translation. Users can submit their own metadata files or create metadata files using our online metadata create form. Registered users can use our REST webservice to batch submit metadata, or update previously submitted metadata of their own.

1. **Register**
   Users don’t need register to search for data. But to upload or create metadata, manage metadata files, users must register. It takes just seconds to register. Click here to register now.

2. **Upload or create geographical metadata and translate metadata standards**
   Users can upload, create, and translate metadata in the Upload Metadata area.

3. **Manage metadata**
   Users can manage their own metadata using our online metadata management system. Registration is required.

4. **Search and download geographical (meta) data**
   The main objective of the application is for geographical information users and developers to convert between different metadata standards. But we do manage some metadata on our server so users can search for data. It is not included in the main navigation menu. But users can access this services from here.

---

**Weather: Reno, NV**

**Thursday, July 11, 2013**

**Tonight**

L: 56°F
Breezy
Mostly clear, with a low around 56. Breezy, with a west wind 15 to 20 mph decreasing to 5 to 10 mph after midnight. Winds could gust as high as 30 mph.

**Friday**

H: 88°F
Sunny
Sunny, with a high near 88. West wind 5 to 15 mph.

**Friday Night**

L: 57°F
Clear
Clear, with a low around 57. West wind 20 to 15 mph becoming light in the evening. Winds could gust as high as 25 mph.

**Saturday**

H: 90°F
Sunny
Sunny, with a high near 90. West wind around 5 mph becoming east in the morning.

**Saturday Night**

L: 59°F
Mostly Clear
Mostly clear, with a low around 59. West wind 5 to 10 mph.

**Sunday**

H: 94°F
Sunny
Sunny, with a high near 94.

---

![Global Land-Ocean Temperature Index](https://via.placeholder.com/150)

Photo courtesy of Wikipedia.org

---

**Figure 4.12: MCA home page**
Open Standards for Services and Protocols. While it contains a set of useful capabilities for managing metadata, it has no provisions for metadata translation across standards.

The IDACT Transformation Manager project was part of a group research on the Arctic Pollution Issues conducted by AMAP (Arctic Monitoring and Assessment Programme). It leverages the knowledge of a group of users, ensuring that once a method for a data transformation has been defined (either through automated processes or by a single member of the group) it can be automatically applied to similar future datasets by all the members of the group. However, this prototype was primarily focused on the dataset itself instead of the metadata[1, 8].

The Data Observation Network for Earth (DataONE) is “the foundation of new innovative environmental science through a distributed framework and sustainable cyberinfrastructure that meets the needs of science and society for open, persistent, robust, and secure access to well-described and
Metadata Community Adaptor by Qiping Yan

Version: 1.0.1

In today’s world, geographic data plays an increasing role in many areas, including academic research, government decision making, and in peoples everyday lives. As the quantity of geographic data gets larger, making full use of the data in a distributed, heterogeneous network environment, like the Internet, becomes a major issue. To better utilize and share those valuable resources, metadata standards have been developed.

Metadata makes it easier to discover, explore, and share geographic data, particularly for cataloguing geographic data in data warehouses. But one of the new problems that came about with metadata is interoperability since multiple metadata standards exist. Important geographic metadata standards include: ISO 19115, Dublin Core, CDGM (US) and prENV 12657 (Europe). Semantically, metadata standards are distinct but, rather, they overlap and relate to each other in diverse and complex ways. In this research, we propose a mechanism to transform different geographic metadata standards to solve the interoperability issue using the new popular web service and XML/XSLT technologies.

This is the topic I have chosen for my Ph.D dissertation. Its objective is to address the metadata issues faced by many geospatial researchers and institutions -- interoperability between different metadata standards.

Weather for Reno/Sparks (Thursday, July 11, 2013)

°F
Clear
Wind: 16 mph (W)

Red Flag Warning for Washoe, NV
Figure 4.15: MCA Search page

Figure 4.16: MCA search result page
easily discovered Earth observational data” [15]. DataOne consists of three coordination nodes that are located at University of New Mexico, the University of California Santa Barbara and, the University of Tennessee. These coordinating nodes provide catalog, indexing, and replication services for all the member nodes. Member nodes are a distributed network of data centers, networks or organizations. To join the network of DataONE member nodes, a data center has to implement the required DataONE member node service interface. Furthermore, member nodes must have valid metadata to join DataONE (and one of the initial purposes of the MCA was to achieve this). However, in terms of metadata translation, the DataONE website mentions only two limited solutions available, i.e. two programs, EsriEML and BDP2EML, which can only uni-directionally translate metadata from ESRI or FGDC to EML and from BDP to EML.

Another popular resource, the open source Geoportal Server[16], offers a valuable set of metadata processing resources, including online catalog services and capabilities for metadata searching, mapping,
Figure 4.18: MCA create metadata page
Figure 4.19: MCA metadata translation web service page
Figure 4.20: MCA test page for metadata translation web service
Figure 4.21: MCA map viewer page
Figure 4.22: Tri-State Data Hub metadata detail page
generation, and validation. However, there is no support for translations between metadata standards.

From the above, it can be seen that the existing efforts in processing metadata are especially limited in relation to metadata translation (the few that exist are either unidirectional or “hard coded”).

Thus, the proposed MCA approach and web-based environment will not only answer several specific needs of the NSF EPSCoR-funded “Cyberinfrastructure Developments for the Western Consortium of Idaho, Nevada and New Mexico,” but, more generally, will address the needs of researchers and institutions that rely on geospatial metadata and data. MCA seeks to deal with several significant challenges in using metadata, in particular in generating metadata, validating metadata, and translating metadata across standards.

Furthermore, MCA will provide an environment and set of cohesive resources that, due to the modern technologies used in their implementation (in particular the .NET framework), will integrate seamlessly with the current Nevada Climate Change Portal (http://sensor.nevada.edu/). Thus, it will fulfill the requirements of the project that initiated and supported the work presented in this dissertation. Moreover, MCA will offer methodological guidance for processing metadata and will be expandable through contributions from the community of users, which will be significant assets in dealing with the key challenge faced by researchers in environmental sciences, that of data interoperability and compatibility.
Chapter 5

Requirements and design

In this chapter, we discuss requirements and design of MCA, which ensures the completeness of the application and a good user experience. The use case and requirement specifications of MCA is presented using the UML (Unified Modeling Language). The use case modeling will capture the essential interactions between users and the features and functions provided by MCA. This is followed by the system requirement specifications, which include the details of the behaviors and related operational constraints of MCA. Last, the requirement traceability matrix is presented. The traceability matrix depicts the mapping between use case and the functional requirements.

5.1 Use case modeling

Use case is a good starting point for most software projects. It provides a systematic way to analyze and capture the interactions between users and the system features and functions. This section presents use case diagrams for MCA and descriptions of the use case.
5.1.1 Use case diagram

MCA uses use case modeling to visually represent the interaction between users and the application. A use case diagram outlines interactions between a system and the external entities that interact directly with it; in UML terminology the external entities are called actors. An actor can be a person or another system that interacts directly with our system; or an actor can be time, if temporal constraints apply directly on the system's behavior. The interactions between the system and actors are identified as use cases, where each use case describes how an actor uses the system, and how the system behaves as seen from an external point of view. In the following use case diagrams, using UML notations, the system boundary is drawn as a box with a label for the system's name, actors are drawn using “stick” figures outside the boundary with labels that identify their roles, and use cases are drawn as ovals with their names inside. Furthermore, associations between actors and use cases are indicated by arrows. Notably, all use cases are always started by actors, operating from outside the system.[17]

Figure 5.1 displays the use case diagram of MCA with 2 actors, a data consumer and a data provider user. A data consumer can access MCA from the website. They can use only the services for regular users. A data consumer does not have to register and login. They can search for resources, update contact information if registered, login, and logout. A data providing user can use all the services for data consuming users and the services for data providers. A data provider must register and login to access the services specifically for data providers. They can upload metadata, create metadata, and translate metadata after login.
Figure 5.1: MCA Use Case Diagram
5.1.2 Description of use case

Table 5.1 provides brief descriptions for each use case. The use case marked with * are key use cases, which are provided more details in next section.

Table 5.1: Use case description of MCA application

<table>
<thead>
<tr>
<th>ID/Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>UC01 Register</td>
<td>The user registers an account with MCA.</td>
</tr>
<tr>
<td>UC02 Login</td>
<td>The user login to MCA.</td>
</tr>
<tr>
<td>UC03 LoginFacebook</td>
<td>The user logs in to MCA through his Facebook account.</td>
</tr>
<tr>
<td>UC04 LoginGoogle</td>
<td>The user logs in to MCA through his Google account.</td>
</tr>
<tr>
<td>UC05 LoginTwitter</td>
<td>The user logs in to MCA through his Twitter account.</td>
</tr>
<tr>
<td>UC06 LoginMicrosoft</td>
<td>The user logs in to MCA through his Microsoft account.</td>
</tr>
<tr>
<td>UC07 SearchResource</td>
<td>The user searches for resources.</td>
</tr>
<tr>
<td>UC08 AccountManagement</td>
<td>The user manages his account such as changing email address and password.</td>
</tr>
<tr>
<td>UC09 Logout</td>
<td>The user logs out from MCA.</td>
</tr>
<tr>
<td>UC10 Feedback</td>
<td>The user submits feedback to MCA through the online form and email.</td>
</tr>
<tr>
<td>UC11 Contact</td>
<td>The user contacts MCA through the online form or email.</td>
</tr>
<tr>
<td>UC12 UploadMetadata*</td>
<td>The user submits his metadata file to the Tri-State Data Hub.</td>
</tr>
<tr>
<td>UC13 ValidateMetadata*</td>
<td>The user submits a metadata file for validation. The system returns the validation result. Valid or Invalid.</td>
</tr>
<tr>
<td>UC14 CreateMetadata*</td>
<td>The user creates a metadata file and submits it to the Tri-State Data Hub after logging to MCA. MCA provides the online metadata creator.</td>
</tr>
<tr>
<td>UC15 TranslateMetadata*</td>
<td>The user provides a metadata file in one metadata standard and MCA returns a metadata file in the target metadata standard format.</td>
</tr>
<tr>
<td>UC16 MetadataUpdate</td>
<td>The user updates one of the previously uploaded resources.</td>
</tr>
<tr>
<td>UC17 MetadataDeletion</td>
<td>The user deletes one of the previously uploaded resources.</td>
</tr>
<tr>
<td>UC18 WSMetadataUpload*</td>
<td>The user uploads a metadata file to Tri-State Data Hub using MCA's web services.</td>
</tr>
<tr>
<td>UC19 WSMetadataValidation*</td>
<td>The user validates a metadata file to Tri-State Data Hub using MCA's web services.</td>
</tr>
<tr>
<td>UC20 WSMetadataTranslation*</td>
<td>The user translates a metadata file to Tri-State Data Hub using MCA's web services.</td>
</tr>
</tbody>
</table>

5.1.3 Key use case in details

In this section, we present several key sample use cases in detail to show the artifacts produced during the use case modeling. We use the Table 5.2 as the template for all the key use cases.
Table 5.2: Detailed use case template

<table>
<thead>
<tr>
<th>Section</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Use case number and name</td>
</tr>
<tr>
<td>Summary</td>
<td>Brief description of the use case</td>
</tr>
<tr>
<td>Rationale</td>
<td>Description of the reason that the use case is needed</td>
</tr>
<tr>
<td>Users</td>
<td>A list of all the categories of uses that interact with this use case</td>
</tr>
<tr>
<td>Preconditions</td>
<td>The state of the software before the use case begins</td>
</tr>
<tr>
<td>Basic course of events</td>
<td>A numbered list of interactions between the system and one or more users</td>
</tr>
<tr>
<td>Alternative paths</td>
<td>Conditions under which the basic course of events could change</td>
</tr>
<tr>
<td>Postconditions</td>
<td>The state of the software after the use case ends</td>
</tr>
</tbody>
</table>

Table 5.3 to Table 5.9 show the key use case in detail to demonstrate system behaviors and the interaction sequences between the users and MCA system.
<table>
<thead>
<tr>
<th>Name</th>
<th>UC12: UploadMetadata</th>
</tr>
</thead>
<tbody>
<tr>
<td>Summary</td>
<td>The user submits his metadata file to the Tri-State Data Hub.</td>
</tr>
<tr>
<td>Rationale</td>
<td>As a data provider, a user or organization has data resources. It is expensive to</td>
</tr>
<tr>
<td></td>
<td>collect those valuable data. In order to fully utilize the resources, they need to</td>
</tr>
<tr>
<td></td>
<td>be re-used and shared by other researchers or users. One way to achieve that goal is</td>
</tr>
<tr>
<td></td>
<td>to upload the metadata file to an online cataloging service so they will be</td>
</tr>
<tr>
<td></td>
<td>discoverable and therefore other users can access the resources. Tri-state data hub</td>
</tr>
<tr>
<td></td>
<td>provides cataloging service to data providers.</td>
</tr>
<tr>
<td>Users</td>
<td>Data Providers</td>
</tr>
<tr>
<td>Preconditions</td>
<td>The user has registered for an account and has logged in, and has metadata file in</td>
</tr>
<tr>
<td></td>
<td>a local system.</td>
</tr>
<tr>
<td>Basic course of events</td>
<td>A. The user points their web browser to MCA web site.</td>
</tr>
<tr>
<td></td>
<td>B. The user logs in with his user id and password.</td>
</tr>
<tr>
<td></td>
<td>C. The user clicks on &quot;Upload Metadata&quot;.</td>
</tr>
<tr>
<td></td>
<td>D. The user chooses &quot;upload metadata&quot; in the action drop down list in the page. A</td>
</tr>
<tr>
<td></td>
<td>file upload form is displayed.</td>
</tr>
<tr>
<td></td>
<td>E. The user clicks &quot;Choose File&quot; and picks the metadata file from the local file</td>
</tr>
<tr>
<td></td>
<td>system.</td>
</tr>
<tr>
<td></td>
<td>F. The user clicks &quot;Upload&quot;</td>
</tr>
<tr>
<td>Alternative paths</td>
<td>The user can submit a metadata file using MCA's web service.</td>
</tr>
<tr>
<td>Postconditions</td>
<td>The metadata file is uploaded to the Tri-State Data Hub and is indexed. The resource</td>
</tr>
<tr>
<td></td>
<td>shows up in the search result list.</td>
</tr>
</tbody>
</table>
Table 5.4: Detailed use case of UC13: ValidateMetadata

<table>
<thead>
<tr>
<th>Name</th>
<th>UC13: ValidateMetadata</th>
</tr>
</thead>
<tbody>
<tr>
<td>Summary</td>
<td>The user submits a metadata file for validation. The system returns the validation result. Valid or Invalid.</td>
</tr>
<tr>
<td>Rationale</td>
<td>Many data collecting organizations created metadata for their data when the data was originally collected. Others may have created the metadata after the data was collected. When metadata was created, errors are injected. Therefore, the metadata files could be invalid and unusable. MCA can check metadata files for validation.</td>
</tr>
<tr>
<td>Users</td>
<td>Data Providers</td>
</tr>
<tr>
<td>Preconditions</td>
<td>The user has registered for an account and has logged in, and has metadata file in a local system.</td>
</tr>
<tr>
<td>Basic course of events</td>
<td></td>
</tr>
<tr>
<td></td>
<td>A. The user points their web browser to MCA web site.</td>
</tr>
<tr>
<td></td>
<td>B. The user logs in with his user id and password.</td>
</tr>
<tr>
<td></td>
<td>C. The user clicks on &quot;Upload Metadata&quot;.</td>
</tr>
<tr>
<td></td>
<td>D. The user chooses &quot;upload metadata&quot; in the action drop down list in the page. A file upload form is displayed.</td>
</tr>
<tr>
<td></td>
<td>E. The user clicks &quot;Choose File&quot; and picks the metadata file from the local file system.</td>
</tr>
<tr>
<td></td>
<td>F. The user clicks &quot;Validate&quot;</td>
</tr>
<tr>
<td>Alternative paths</td>
<td>The user can submit a metadata file using MCA's web service.</td>
</tr>
<tr>
<td>Postconditions</td>
<td>The metadata file is checked. The user can correct errors if it is invalid. Valid metadata files can be submitted to Tri-State DAha Hub or other online clearinghouse later or used by applications in the local system.</td>
</tr>
</tbody>
</table>
### Table 5.5: Detailed use case of UC14: CreateMetadata

<table>
<thead>
<tr>
<th>Name</th>
<th>UC14: CreateMetadata</th>
</tr>
</thead>
<tbody>
<tr>
<td>Summary</td>
<td>The user creates a metadata file and submit it to the Tri-State Data Hub after logging to MCA. MCA provides the online metadata creator.</td>
</tr>
<tr>
<td>Rationale</td>
<td>Some organizations did not create the metadata when they collected their data. Without the metadata, it is very difficult to share and re-use the data. MCA provide an online metadata creator so users can create metadata for their data.</td>
</tr>
<tr>
<td>Users</td>
<td>Data Providers</td>
</tr>
<tr>
<td>Preconditions</td>
<td>The user has registered for an account and has logged in, and has metadata file in a local system. The user has all the required information about the resource.</td>
</tr>
<tr>
<td>Basic course of events</td>
<td></td>
</tr>
<tr>
<td></td>
<td>A. The user points their web browser to MCA web site.</td>
</tr>
<tr>
<td></td>
<td>B. The user logs in with his user id and password.</td>
</tr>
<tr>
<td></td>
<td>C. The user clicks on &quot;Upload Metadata&quot;.</td>
</tr>
<tr>
<td></td>
<td>D. The user chooses &quot;Create Metadata&quot; in the action drop down list in the page.</td>
</tr>
<tr>
<td></td>
<td>E. The creation sub-menu appears. The user chooses the appropriate metadata standard in the drop down list.</td>
</tr>
<tr>
<td></td>
<td>F. The metadata creator window appears.</td>
</tr>
<tr>
<td></td>
<td>G. The user clicks each section on the left side of the metadata creator window. The corresponding form fields appears on the right side of the metadata creator window.</td>
</tr>
<tr>
<td></td>
<td>H. The user fills all the required fields.</td>
</tr>
<tr>
<td></td>
<td>I. The user clicks the submit button on the bottom of the page.</td>
</tr>
<tr>
<td>Alternative paths</td>
<td>None</td>
</tr>
<tr>
<td>Postconditions</td>
<td>The user created a new metadata file for a resource and submitted it to Tri-State Data Hub. The resource show up in search result lists.</td>
</tr>
</tbody>
</table>
Table 5.6: Detailed use case of UC15: TranslateMetadata

<table>
<thead>
<tr>
<th>Name</th>
<th>UC15: TranslateMetadata</th>
</tr>
</thead>
<tbody>
<tr>
<td>Summary</td>
<td>The user provides a metadata file in one metadata standard and MCA returns a metadata file in the target metadata standard format.</td>
</tr>
<tr>
<td>Rationale</td>
<td>When an organization creates metadata files for their data, it chooses a metadata standard for the files. There are many geospatial metadata standards existing in the market and different software applications support different metadata standards. There are situations when a user needs a metadata in a format and the data owner only provides it in a different format. MCA’s metadata translation service addresses this issue.</td>
</tr>
<tr>
<td>Users</td>
<td>Data Providers</td>
</tr>
<tr>
<td>Preconditions</td>
<td>The user has registered for an account and has logged in, and has metadata file in a local system.</td>
</tr>
<tr>
<td>Basic course of events</td>
<td>A. The user points their web browser to MCA web site.</td>
</tr>
<tr>
<td></td>
<td>B. The user login with his user id and password.</td>
</tr>
<tr>
<td></td>
<td>C. The user click on &quot;Upload Metadata&quot;.</td>
</tr>
<tr>
<td></td>
<td>D. The user choose &quot;Convert&quot; in the action drop down list in the page.</td>
</tr>
<tr>
<td></td>
<td>The translate form appears.</td>
</tr>
<tr>
<td></td>
<td>E. The user clicks on &quot;Choose File&quot; and picks the metadata file from the local file system.</td>
</tr>
<tr>
<td></td>
<td>F. The user chooses the from standard and target standard.</td>
</tr>
<tr>
<td></td>
<td>G. The user clicks on the &quot;Translate&quot; button.</td>
</tr>
<tr>
<td>Alternative paths</td>
<td>The user can translate a metadata from one metadata standard into another standard format using MCA’s translation web service.</td>
</tr>
<tr>
<td>Postconditions</td>
<td>The user gets a metadata file in the target metadata standard format. It can be submitted to Tri-State Data Hub or other online clearinghouse services later or used by a local system.</td>
</tr>
<tr>
<td>Name</td>
<td>UC18: WSMetadataUpload</td>
</tr>
<tr>
<td>--------------</td>
<td>----------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Summary</td>
<td>The user upload a metadata file to Tri-State Data Hub using MCA's web services.</td>
</tr>
<tr>
<td>Rationale</td>
<td>While a data provider could use the online metadata upload service to submit metadata to Tri-State Data Hub for indexing and cataloging, MCA provides a web service UI for data producers to submit their metadata in a batch mode, or automatically on the fly.</td>
</tr>
<tr>
<td>Users</td>
<td>Data Providers</td>
</tr>
<tr>
<td>Preconditions</td>
<td>The user has a metadata file in local system. The user has registered with MCA.</td>
</tr>
</tbody>
</table>
| Basic course of events | A. The user forms web service request either manually or with a software.  
|               | B. The user attaches the metadata file and user id to the request.                   |
|               | C. The user sends the web service request to MCA.                                      |
| Alternative paths | The user can also use MCA's online form to submit a metadata file.                  |
| Postconditions| The user gets a response indicating whether the submission is successful or not.      |
Table 5.8: Detailed use case of UC19: WSMetadataValidation

<table>
<thead>
<tr>
<th>Name</th>
<th>UC19: WSMetadataValidation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Summary</td>
<td>The user validate a metadata file to Tri-State Data Hub using MCA’s web services.</td>
</tr>
<tr>
<td>Rationale</td>
<td>It is the same rationale as the UC18. With this web service, metadata validation can be done in a batch mode programmatically, instead of manually.</td>
</tr>
<tr>
<td>Users</td>
<td>Data Providers</td>
</tr>
<tr>
<td>Preconditions</td>
<td>The user has a metadata file in local system. The user has registered with MCA.</td>
</tr>
<tr>
<td>Basic course of events</td>
<td>A. The user forms web service request either manually or with software.</td>
</tr>
<tr>
<td></td>
<td>B. The user attaches the metadata file and user id to the request.</td>
</tr>
<tr>
<td></td>
<td>C. The user sends the web service request to MCA.</td>
</tr>
<tr>
<td>Alternative paths</td>
<td>The user can also use MCA’s online form to submit a metadata file.</td>
</tr>
<tr>
<td>Postconditions</td>
<td>The user gets a response from MCA indicating whether the file is a valid metadata file or not.</td>
</tr>
</tbody>
</table>
Table 5.9: Detailed use case of UC20: WSMetadataTranslation

<table>
<thead>
<tr>
<th>Name</th>
<th>UC20: WSMetadataTranslation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Summary</td>
<td>The user translates a metadata file to Tri-State Data Hub using MCA's web services.</td>
</tr>
<tr>
<td>Rationale</td>
<td>The metadata translation web service can be used programmatically to accomplish the metadata translation tasks. It can also be integrated into a local system to translate metadata on the fly.</td>
</tr>
<tr>
<td>Users</td>
<td>Data Providers</td>
</tr>
<tr>
<td>Preconditions</td>
<td>The user has a metadata file in local system. The user has registered with MCA.</td>
</tr>
<tr>
<td>Basic course of events</td>
<td></td>
</tr>
<tr>
<td></td>
<td>A. The user form web service request either manually or with a software.</td>
</tr>
<tr>
<td></td>
<td>B. The user attach the metadata file, from metadata standard, target metadata standard, and user id to the request.</td>
</tr>
<tr>
<td></td>
<td>C. The user send the web service request to MCA.</td>
</tr>
<tr>
<td>Alternative paths</td>
<td>The user can also use MCA's online form to submit a metadata file.</td>
</tr>
<tr>
<td>Postconditions</td>
<td>The user get a response which contains the metadata file in the target format or a error code if the translation failed.</td>
</tr>
</tbody>
</table>

5.2 Requirements specification

In this section, we discuss the software requirements of MCA. The software requirement specifications are developed based on the use cases we presented in last section. The software requirements are ideal engineering tools for software projects like MCA. It not only serves as a guideline for overall software development, but also is good for unit testing, application deployment, and even for contractual agreement between the developing firms and their customers. Software requirement specifications include both functional and non-functional requirements. Each requirement is categorized into different levels,
which represent the priority levels of the associated requirement. For MCA, we have 3 priority levels for each requirement that are shown in Table 5.10.

<table>
<thead>
<tr>
<th>Level</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Functions and features that must be implemented.</td>
</tr>
<tr>
<td>2</td>
<td>Functions and features that might be implemented.</td>
</tr>
<tr>
<td>3</td>
<td>Functions and features that will not be implemented, but make interesting and worthwhile items for future works.</td>
</tr>
</tbody>
</table>

In this section, we list both functional and non-functional requirements of MCA.

### 5.2.1 Functional requirements

Functional requirements are core requirements of the application and are defined based on the use cases discussed in section 5.1. A functional requirement describes the functionality that the system provides in a clear, short, and precise definition. Table 5.11 lists the functional requirements of MCA.

<table>
<thead>
<tr>
<th>ID(Level)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FR01(i)</td>
<td>MCA shall provide a navigation system for main functions and main features.</td>
</tr>
<tr>
<td>FR02(i)</td>
<td>MCA shall provide a registration page for data consumers.</td>
</tr>
<tr>
<td>FR03(i)</td>
<td>MCA shall provide a registration page for data providing users.</td>
</tr>
<tr>
<td>FR04(i)</td>
<td>MCA shall provide a login page for data consumers.</td>
</tr>
<tr>
<td>FR05(i)</td>
<td>MCA shall provide a login page for data providing users.</td>
</tr>
<tr>
<td>FR6(i)</td>
<td>MCA shall provide a search function for users.</td>
</tr>
<tr>
<td>FR7(i)</td>
<td>MCA shall allow users to update contact information such as email address and password.</td>
</tr>
<tr>
<td>FR8(2)</td>
<td>MCA shall allow data providers to update previously uploaded resources.</td>
</tr>
<tr>
<td>FR9(i)</td>
<td>MCA shall allow data providers to delete previously uploaded resources.</td>
</tr>
<tr>
<td>FR10(i)</td>
<td>MCA shall allow data providers to submit metadata files to Tri-State Data Hub.</td>
</tr>
<tr>
<td>FR11(i)</td>
<td>MCA shall allow data providers to validate a metadata file.</td>
</tr>
<tr>
<td>FR12(i)</td>
<td>MCA shall allow users to send feedback to MCA through a web form or email.</td>
</tr>
<tr>
<td>FR13(i)</td>
<td>MCA shall allow users to contact MCA through a web form or email.</td>
</tr>
<tr>
<td>FR14(i)</td>
<td>MCA shall allow users to view resources on a map.</td>
</tr>
<tr>
<td>FR15(i)</td>
<td>MCA shall allow users to view resource details on a web page.</td>
</tr>
<tr>
<td>FR16(i)</td>
<td>MCA shall provide an online metadata creator software.</td>
</tr>
<tr>
<td>FR17(i)</td>
<td>MCA shall allow users to choose a metadata standard when creating a metadata file to submit.</td>
</tr>
<tr>
<td>FR18(i)</td>
<td>MCA shall provide a mechanism for users to prevent mistyping when creating a metadata file.</td>
</tr>
<tr>
<td>FR19(i)</td>
<td>MCA shall allow users to translate a metadata from one standard into a different metadata standard.</td>
</tr>
<tr>
<td>FR20(i)</td>
<td>MCA shall provide a web service for users to submit a metadata file to Tri-State Data Hub.</td>
</tr>
<tr>
<td>FR21(i)</td>
<td>MCA shall provide a web service for users to validate a metadata file.</td>
</tr>
<tr>
<td>FR22(i)</td>
<td>MCA shall provide a web service for users to translate a metadata file from one metadata standard format into a different metadata standard format.</td>
</tr>
<tr>
<td>FR23(3)</td>
<td>MCA shall provide an online geospatial community, so members can contribute to MCA by adding new metadata standards to the supporting metadata list.</td>
</tr>
<tr>
<td>FR24(2)</td>
<td>MCA shall allow users to integrate MCA services into their own system.</td>
</tr>
<tr>
<td>FR25(3)</td>
<td>MCA shall allow data providers to submit a metadata file that is not included in the current supported metadata standard list.</td>
</tr>
<tr>
<td>FR26(1)</td>
<td>MCA shall provide the link to the actual dataset to download or purchase the datasets.</td>
</tr>
<tr>
<td>FR27(1)</td>
<td>MCA shall allow users to save a metadata file to a local system.</td>
</tr>
<tr>
<td>FR28(3)</td>
<td>MCA shall allow users to share a resource to other platforms.</td>
</tr>
<tr>
<td>FR29(1)</td>
<td>MCA shall provide a logout page for data consumers.</td>
</tr>
</tbody>
</table>

### 5.2.2 Non-functional requirements

As the name implies, a non-functional requirement is not about a specific function or feature provided by the system. Rather, it is primarily concerned with the constraints and emergent properties of the system. It is more about how to make the system work efficiently by implementing functional features in a proper way or by integrating a specific framework. Table 5.12 lists the non-functional requirements of MCA.
Table 5.12: Non-functional software requirements of MCA

<table>
<thead>
<tr>
<th>ID(Level)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NR01(i)</td>
<td>MCA shall be implemented in a Windows server.</td>
</tr>
<tr>
<td>NR02(i)</td>
<td>MCA shall be implemented using C# for the server side programming.</td>
</tr>
<tr>
<td>NR03(i)</td>
<td>MCA shall be implemented using Javascript for client side scripting.</td>
</tr>
<tr>
<td>NR04(i)</td>
<td>MCA shall use Microsoft SQL Server for user account management services.</td>
</tr>
<tr>
<td>NR05(i)</td>
<td>MCA shall support HTML 5.</td>
</tr>
<tr>
<td>NR06(i)</td>
<td>MCA shall provide a consistent user interface for all web pages.</td>
</tr>
<tr>
<td>NR07(i)</td>
<td>MCA shall provide a user friendly interface to produce a good user experience.</td>
</tr>
<tr>
<td>NR08(2)</td>
<td>MCA shall provide an easy user interface for users to share resources.</td>
</tr>
<tr>
<td>NR09(3)</td>
<td>MCA shall provide an easy to use interface for users to integrate MCA’s services into other systems.</td>
</tr>
<tr>
<td>NR10(3)</td>
<td>MCA shall provide an easy to use interface for community members to contribute to the system.</td>
</tr>
<tr>
<td>NR11(3)</td>
<td>MCA shall provide an interface to add new metadata standards to the system in the future.</td>
</tr>
</tbody>
</table>

### 5.2.3 Requirements traceability matrix

Requirements traceability matrix is used to check consistency of the specification and to avoid missing use cases in the development of the project. Use case and functional requirements have a many-to-many relationship. One use case may cover more than one functional requirement and every functional requirement should have one or more corresponding use cases. Figure 5.2 shows the requirements trace-
ability matrix of MCA.

### 5.3 Design aspects of MCA

As a research and academic web application, MCA must address the fundamental design aspects of web application development, such as navigation, menu system, information flow, membership management, error handling system, user experience, security issues and so on. There are other design ele-
ments which are vital to commercial web applications that are not important to MCA. Those elements include: user stickiness, conversion rate, SEO (Search Engine Optimization), and merchant transaction.

The majority of the design elements of MCA have been mentioned in previous chapter (Chapter 4 section 4.2). They include: 1) MCA Architecture (Figure 4.2), 2) Tri-State Geospatial Metadata Exchange (Figure 4.1), and 3) MCA Software architecture (Figure 4.3). Therefore, they will not be repeated in this section. For details of those aspects, please refer to chapter 4 section 4.2.

This section will provide details of the design concerns from users’ perspective. As illustrated in Figure 5.3, MCA provide three service components. The first component is the general public area which includes home page, about page, resources page, contact, feedback, and search page. Users can browse those pages without login. The next component is member-based area which includes metadata upload, metadata management, community contribution, and all subsections beneath them. Users must login to access those services. A user can register and a MCA account or use an account from a trusted company. MCA currently supports users from Google, Facebook, Microsoft, and Twitter. The last part is the web services. Web service is for other data providers to submit resource programmati- cally/automatically to TSDP server.
Figure 5.3: MCA Functionality Flowchart
Figure 5.4: MCA create metadata page
Figure 5.5: Screen shot of MCA metadata uploading page

Figure 5.6: MCA Metadata translation page
Chapter 6

User interface aspects

This chapter on the user interface is broken into three sections: 1) the definition of the user interface, why it matters, the cost of a bad interface, and why a software company spends approximately 50% in the user interface design, 2) specific concerns pertaining to web user interface, and 3) the user interface of MCA, which includes the design overview, features and functions, and techniques used during the implementation of MCA.

6.1 User interface

User interface (UI) is an important part of Human-Computer Interaction (HCI) or Human-Machine Interaction (HMI). The user interface actually applies to other domains that are related to computers, such as cars, microwaves, refrigerators, televisions, etc. In this dissertation I focus on software interface, specifically web user interface. This is the space where interaction between human and machine occurs. As it applies to Internet applications, the user interface provides a mechanism for users to interact with the software behind the scene. Therefore users can complete tasks more easily and effectively. User interface design is the most important part of a software system because it is part of the software that a user can see, hear, touch, and feel. Users cannot see the codes which drive the software
application.

### 6.1.1 User interface matters

In our lives, people use an increasing number of computer-based technologies, either directly or indirectly. Directly these could include desk-top and laptop computers, tablet computers, and smart phones. Indirectly we see (smart) TVs, microwaves, refrigerators, cars, etc. When we make a purchase, be it a computer, a smart phone, or a car, we look for the one with the most features within our budget; however, after we spend our money, we never use most of the features available. Look at your TV remote, do you find buttons you never touch? Some remotes provide an additional panel of buttons hidden beneath a cover. To prevent dust, some people wrap their remotes with plastic bags the first day they use the device. They never know that another panel of buttons exists. We see VCRs with a flashing 12:00 after years of use; it has never been set up. While incomplete gadget usage may be due to useless features, most of the time it is due to a bad user interface design.

User interface is the most important part of any computer system, software application, and Internet application. The user interface represents the system to users. Users can see, touch, and hear it. The software codes are invisible and hidden behind the scene, which drive the system and control the user interface. The goal of user interface design is to provide a functional system that is easy, productive, and enjoyable. This should culminate in the purchase of a service or product from an e-commerce website.

### 6.1.2 User interface design

User interface design is the design of computer systems, websites, and software applications with the focus on the user’s experience and interaction. The goal of user interface design is to make the user’s interaction with the system as simple and efficient as possible. The HCI (human-computer interaction) is the study of how humans interact with computer systems. HCI is a cross-discipline study, which includes computer science, psychology, ergonomics, sociology, engineering, and graphic design. There is no one size fits all approach for user interface design. A good user interface design requires a good
understanding of the target user's needs.
Depending on the project, some of the design phases experience more demand than others. Some of the important UI design phases include:

- Gathering functionality requirements – a list of system functionalities and user's needs
- User analysis – analyze the potential users of the system by talking to those users and by studying related researches
- Prototyping – develop wireframes of the user interface
- Usability test – test the prototype on actual users and collect feedback
- Graphic interface design – design the final user interface (GUI). It is a phase of collaborative effort between graphic designers and UI engineers

### 6.1.3 Guidelines for user interfaces

There is no solidified definition of a good or bad user interface, because they are relative and interchangeable. DOS was a good user interface when compared to punch cards, but a weak user interface when compared to Windows. Also, user interfaces change over time. It is easy to understand that a user interface will eventually become obsolete and get replaced by newer user interfaces as technology advances. Sometimes the same user interface can be good in one platform while it is not good in another platform. So Siri is bad, but only for desktop? If so, for example, Siri for iPhone or iPad a a great user interface, while a comparable application has existed unsuccessfully for many years in the desktop platform. It is also true that a good user interface for some users could be bad for others. For instance, a clicking/dragging design is good for beginners and the keyboard/short cut interface is more efficient for professional engineers.

A good user interface, which is usually project based, should include the following attributes. Note that these are listed here as a result of my experience over many years of interactive software development. This experience has also been shaped by reading various HCI books. For detailed HCI guidelines,
principles, and techniques, the reader is referred to textbooks such as [63], [64], and [65]

**Concerning users’ attention.** When a user visits a website, his first and main purpose is to get information by reading the contents in the webpage. Reading on a computer screen or on a tablet can be disruptive as there are many distractions. When designing a web user interface, attention is vital. Do not litter distractible materials around the main content of the page. If you must have ads in the page, place them where users will look when finished reading the main contents. If the designer honors user attention, the user will be happier and the ads will function better.

**Each page should have only one main topic of purpose.** Having more than one primary topic in a single page will be confusing, much like having two performances occur at the same time. Of course, this is acceptable if it takes multiple topics to present a single purpose.

**Keep secondary actions secondary.** A page can have secondary actions, but they need to be secondary. This can be achieved by making secondary actions of lesser consequence visually, or one could show them after the primary action is concluded.

**Provide inline help.** A web site can have a dedicated help system, especially for large sites, but an ideal web application should embed all the help inline in the webpage. The contents should lead users to the next step.

**Prevent possible errors.** The web is getting more interactive and personalized. Many websites feed customized contents to their users. This indicates that companies track user surfing activities using cookie-related technologies. Also, advertising providers (e.g., google adsense) push different ads into the same webpage when a user visits the webpage from different computers or from different locations (home and library or office). To acquire more personal details, websites use online surveys or web forms to collect information. In this case, it is the user interface designers’ job to create a user interface
that can validate and prevent invalid input. Invalid data is hard to detect and even harder and more expensive to fix. Validation can be done with javascripts, while prevention can be achieved by reducing typing through dropdown lists or data picker and such.

**Make your UI invisible.** A great user interface is usually unnoticed by users who use it. If a user interface is successful, users will focus on their own goals instead of the interface. When they finish their tasks, users simply leave the interface. That is why satisfied users are often silent.

There are many categories of bad (web) user interface. Those websites usually lack most of the above good user interface attributes or they just do the opposite.

**Too crowded and distracted.** If a webpage contains too much information, too many flashing buttons, and animations, the user will get distracted and it will be hard for the user to concentrate on reading content. Online users tend to spend very little time on a single website. If a website cannot grab the user’s attention within seconds the user will click away.

**Main action unclear.** A webpage can be divided into primary sections and secondary sections. A user will skip the top and left section because those sections usually contain logos, slogs, and navigation menus. The section in the middle is for primary content and most users will start from there. If the user found the content of the primary section useful, they will look at the contents in the secondary sections for additional details. Therefore, the primary sections should be placed in locations users expect to find primary sections. When a website places its main actions in secondary sections, the search engine will still find the webpage and index the page according to the main contents, while users might not have enough patience to read it before they click away.

**Misleading.** When people look for information they go to search engines like Google. A user enters a term, then Google returns a list of related websites with brief descriptions. Users decide where
to click. There is no misleading in this step since Google is responsible for the accuracy. After a user lands on a webpage it is up to the website interface to provide guidelines and lead users to the proper information page and ensure a pleasant experience. It is very frustrating when a user clicks a link based on the description of the inline guide but lands on a page with wrong content. These misleading links damage the user experience.

**Too much flash.** When working with restaurants we researched restaurant related websites. Most restaurants, especially single family-owned restaurants, believe that a website is a cost-effective way to market their service, which is true if done correctly. Sometimes when a user clicks on a restaurant website, the home page is a beautiful flash with music, photos of dishes, and videos. It takes several minutes before the user can technically enter the website. Some of these flashes do not even offer a skip button or the skip button is non-functional. When users visit a restaurant website, they want information about the restaurant and its services such as address, phone number, business hours, cuisines served, and price information. It is acceptable, and even wise, to include photos and flash videos in a restaurant website, but those features should be placed in a dedicated page instead of the homepage. A nicely done photo and video webpage can help a customer to pick a restaurant.

It is not practical to list all bad web user interface, but many do exist. When you see one you will know it. The cost of a bad user interface is huge.

Great games, software, and website do not just happen. They are the result of the user interface designers’ careful research on target audience, planning, strict attention to detail, instinct, and the occasional flash of inspiration. It is a balance between aesthetics, usability, and functionalities. As is indicated by the many poorly executed user interfaces, the consequence of inadequate design is enormous.
6.2 MCA user interface

The objective of MCA is to create a solution to address the geographic metadata interoperability and compatibility issues. MCA is an internet application with support of metadata creation, translation, and submission. It also provides limited geographic resource indexing, searching, and online cataloging by working with Geoportal. MCA has two major user interface components: web service and web interface. Web service interface is the standard interface for web services that receives requests, processes them, and responds with results. This section will address the web user interface.

6.2.1 Design overview

MCA is a modest internet application for geographic metadata submission, translation, and management. The software logic behind the scene can be quite complex; however, the user interface has to be simple. In return, it makes the software more complicated. As was mentioned in previous sections, the best user interface is one where users use and leave without noticing using it. One successful user interface is Google. There are more than 2.5 billion Google searches a day, but all we remember is the Google logo and a search box. Users go to google.com, search a term, and leave the Google site. In order to return an accurate search result, Google has developed a sophisticated algorithm. To make the simple search experience even better, Google has developed a set of technologies to support the autocomplete feature. When users start to type the search term in the Google search box, Google will show a drop down list of predicted terms based upon the letters already entered. This helps people save time in typing and also helps users who do not know the exact term to search.

Google was the user interface model when we developed MCA, which was predicated upon simplicity. The first principle is simplicity. Colors were limited to three, major sections in a page were limited to three, and there was only one primary section. The second principle is consistency. To make all the pages look and feel the same, we chose to use the ASP.NET master page technology to ensure that each page had the same theme, same navigation, same font size and same color scheme. The last principle is
superior user experience. For example, the date picker in metadata creation section. When users need to enter a date, instead of typing “December, 12, 2013”, users simply click the field, a calendar pops up, and users pick the date by clicking the date in the calendar. The corresponding date will be entered automatically, which saves time for typing and, more importantly, prevents invalid data entry.

6.2.2 Features, functions, and technical aspects

When we designed the user interface for MCA, we studied many related websites and researched different technologies and frameworks to reach a satisfying solution for MCA. Here are some of the major frameworks and technologies we choose for MCA's user interface.

A. Microsoft ASP.NET master page. ASP.NET master page is one of the best solutions in the industry to produce consistent web application user interfaces. The ASP.NET master page system includes a template file with the extension .master and the regular ASP.NET page with the extension .aspx. The master file is identified by a special @master directive instead of the @page directive for ordinary .aspx pages. The master page defines the basic layout with static text, HTML elements, and server controls. The most important server control in the master page is the ContentPlaceHolder control. The content page (.aspx) provides contents for the ContentPlaceHolder controls defined in master page by including the ASP Content control which refers to the specific ContentPlaceHolder controls using the ContentPlaceHolderID attribute.

B. ASP.NET Panel Control. The most important component of MCA is the “Uploading Metadata” component. It provides services such as metadata submission, metadata creation, and metadata translation. Each service includes multiple selections and functions. See Figure 6.1. There are many ways to implement the Upload Metadata feature. We first created a page for each box in Figure 6.1. Then we used hyperlinks to navigate through the maze. There are two major issues
Figure 6.1: Upload Metadata section function illustration
with this approach: 1. we needed a mechanism to store temporary data either using cookies or hidden fields, and 2. we needed a mechanism to handle the situation when a user needs change a previously entered data entry. For example, if a user wants to change data already entered. We tried a “go back” button, but this created additional problems. The required data (fields) in a page depend on the data (choice) the user entered in previous steps. One would have to start from scratch every time the user made a change. The logic for the code behind would be too complicated to be practical.

Another approach we utilized was creating a large page in which we placed all required fields. Theoretically there was nothing wrong with this approach, but it was not functional for our situation. As mentioned, most of the geographic metadata standards are quite sophisticated. The page would have contained hundreds of form fields, potentially scaring user away. If they remained in the site, their user experience would not be pleasant.

We decided to use ASP.NET Panel Control along with MultiView and View Controls for this task. This was the best approach because all tasks can be accomplished within a single page. Most tasks (except the metadata creation) can be finished in less than 4 steps. Even for the metadata creation, users only see one section at a time. While the panel control contains 2 MultiView Controls and 21 View Controls, the page still looks clean and simple because only the panels and views which are involved in the current task are shown, while the rest of the controls are hidden.

C. **MultiView and View Controls.** A MultiView Control contains one or more View controls. A View Control in turn serves as a container and works like a Panel or Canvas that contains other ASP.NET controls, even another MultiView Control. It is very powerful and can get complicated and overwhelming. MultiView and View controls are heavily used in MCA.

D. **AJAX Framework.** Date information exists almost everywhere in geographic metadata standards. Thus, in our Metadata Upload section, we had to implement date fields in many places.
In addition, date field is one of the fields that contains invalid data format and it is hard to detect. In order to prevent users from entering invalid date data and to avoid typing, we utilized the AJAX framework date picker control. Instead of having users manually type a date, the system will popup a calendar each time a user clicks on a date field. After the user clicks on the correct date, the right date will be entered automatically. See Figure 6.2.
Chapter 7

Application scenarios

The objective of this chapter is to present example application scenarios that illustrate in detail the usage of MCA and TSDH. Two application scenarios are illustrated in this chapter. The first scenario was developed to create an ISO 19115 metadata using MCA’s online metadata creator and then upload it into TSDH server. After the new metadata is uploaded, the search feature is used to find the resource, show it on the TSDH map, and reveal the details of the resource. The second application scenario is relatively easy and straightforward. It uses the MCA web service test module to generate a web service request, send it to MCA web service, and show the translated metadata file in the page. The fourth application scenario shows how a geospatial resource can be translated from one metadata standard to another.

7.1 Application scenario 1: Metadata creation and submission

This section describes the first application scenario, which illustrates the MCA web interface for creating and submitting a resource in ISO 19115 standard to the TSDH. Then, we find the resource and display it in the TSDH mapping window. The following is the steps for this scenario.
a) Start MCA in a web browser by entering http://localhost:1324/ in the URL box. See Figure 7.1

b) Login with a MCA account or one of the major websites’ accounts (Google, Facebook, Microsoft, and Twitter). Users can register a MCA account by visiting the registration area located on the top right of any page, or in the login page. See Figure 7.1 and See 7.4

c) Click the "Upload Metadata" button in the site navigation area. The upload metadata section is a member-only section. Users will be directed to the login page if they did not login. See Figure 7.2

d) Choose “Create Metadata” in the drop down list, then choose the “ISO 19115” item in the metadata standard list. The online metadata creator window appears. See Figure 7.2

e) The main sections of the chosen metadata standard are located on the left side of the creator win-
dow. The user clicks each section to bring up all the fields within the section on the right side to complete the form. See Figure 7.2

(f) After all the information is entered, the user clicks the submit button on the bottom of the page to upload the metadata into TSDH. See Figure 7.2

![Image of Metadata creation and submission](image)

**Figure 7.2: Metadata creation and submission**

(g) All users can now find the new resource using the search feature in the home page of MCA. See Figure 7.1

(h) Users can view the resource on a map or view details of the resource by clicking the corresponding buttons or links in the search window. See Figure 7.3
Figure 7.3: Metadata search

Figure 7.4: Metadata creation and submission
7.2 Application scenario 2: Translation with MCA's web services

The web service component of MCA are intended for data providers to submit, validate, and translate metadata files programmatically, instead of manually, for those time-consuming and tedious tasks. In this section, we illustrate the usage of the metadata translation web service. Web services are usually executed behind the scene by stand-alone desk top applications, automation scripts, or web applications running in a web server. For illustration purposes, we use MCA's web service test module for this application scenario.

(a) Manually enter the web address http://localhost:1324/WebServiceTest.aspx into the URL box of a web browser. See Figure 7.1

(b) Click the “Translate Metadata” button. See Figure 7.5

(c) Choose the “Source standard” and “Target standard” from the drop down lists, click the Choose File button to select the source metadata file, then click the Translate button.
(d) The translated metadata file is displayed in the target metadata window. See Figure 7.5

(e) The user clicks the “Save File” button to save the metadata file in a local file system.

7.3 Application scenario 3: Validate a geospatial metadata

When a metadata was created manually, errors are often introduced. An invalid metadata creates problems for later use of the metadata. It is necessary to provide a mechanism to validate a metadata. MCA provides metadata validation services to users. A user can use MCA to validate a local geospatial metadata resource or an online file. Those features are only available to registered members. The login processes is same as steps A to C as described in section 7.1. After login a user can perform a validation as follow: See Figure 7.6

To validate a resource stored in a local file system,

(d) Choose “upload metadata” from the drop down list

(e) Click the "Choose File" button. Then the file explorer will come up. User navigate to the resource file to select the file. Click OK button.
(f) Click the "Validate" button.

Figure 7.6: Validate a local resource file

(g) The validation result will be displayed in the status area in the center or the page.

MCA can also validate a metadata resource located online. Instead of choosing a file from the local file system, a user provides the URL for the metadata file to validate. To validate an online resource, after step (d), do the following: See Figure 7.7

(e) Click [submit or validate an online resource]. The online resource submission form will appear.

(f) Enter the URL into the URL field starting with “http://”
(g) Click the “Validate” button.

![ISO 19115 Community Adaptor - Upload Metadata](image)

Figure 7.7: Validate an online resource file

(h) The validation result will be displayed in the status area in the center or the page.

7.4 Application scenario 4: Translate geospatial resource from one metadata standard to a different standard

There are numerous legacy geospatial applications available in the industry and they will coexist for a foreseeable period of time. Different systems support different metadata standards. To fully utilize the available resources, a translation system is needed to convert geospatial metadata from one standard
into a different metadata standard. MCA provides a metadata translation function to fulfill this need. A user can use MCA to translate a local geospatial metadata resource from one metadata standard to a different metadata standard. This feature is only available to registered members. The login processes is same as steps A to C as described in section 7.1. After login a user can perform a translation as follow: See Figure 7.8

(d) Choose “convert metadata” from the drop down list.

(e) Choose the source format from the drop down list.

(f) Choose the target standard from the target format drop down. The available format list will update accordingly each time the source format changes.

(g) Click the "Choose File" button. Then the file explorer will come up. User navigate to the resource file to select the file. Click OK button.

(h) Click the Translate button. The validation result will be displayed in the status area in the center or the page. See Figure 7.9.
Figure 7.8: Translate a metadata from one standard to a different standard

After translation process finished, the result will be displayed in the status area in the center of the page. Both the source and translated metadata will be displayed inside the content windows below the form. See Figure 7.9.

(i) To save the translated metadata, double click inside the translated window to select. Then user can copy and paste into a new text file to save to the local file system.
Figure 7.9: Translation result
Chapter 8

Comparison with related work

As GPS enabled mobile devices are becoming more popular, location-based apps and GIS services are sprouting up rapidly. The GIS industry and the international GIS community are growing exponentially. The geospatial market is growing at an annual rate of almost 35 percent, with the commercial subsection of the market expanding at the rate of 100 percent each year. In this chapter, we present three online software applications that have made great progress in this area: DataONE, Geoportal Server, and GeoNetwork Opensource.

8.1 DataONE

“Data Observation Network for Earth (DataONE) is the foundation of new innovative environmental science through a distributed framework and sustainable cyberinfrastructure that meets the needs of science and society for open, persistent, robust, and secure access to well-described and easily discovered Earth observational data.”[18]

“Supported by the U.S. National Science Foundation as one of the initial DataNets, DataONE will ensure the preservation, access, use and reuse of multi-scale, multi-discipline, and multi-national science
data via three primary cyberinfrastructure elements and a broad education and outreach program.\[18\]

### 8.1.1 DataONE Cyberinfrastructure

DataONE Cyberinfrastructure includes: Coordinating Nodes, Member Nodes, and Investigator Toolkit.

DataONE currently hosts three Coordinating Nodes that are located at the University of New Mexico, the University of California Santa Barbara, and at the University of Tennessee. They provide network-wide services to enhance interoperability of the Member Nodes and support indexing and replication services. Coordinating Nodes also provide a replicated catalog of Member Node holdings and make it easy for scientists to discover data wherever they reside, also enabling data repositories to make their data and services more broadly available to the international community.\[18\]

DataONE consists of a distributed network of data centers, science networks or organizations. These Member Nodes expose their data within the DataONE network through the implementation of the DataONE Member Node service interface. In addition to scientific data, Member Nodes can provide computing resources, or services such as data replication to the DataONE community.\[18\]

The DataONE Investigator Toolkit enables access to customized tools that are familiar to scientists and that can support them in all aspects of the data life cycle.\[18\]

The purpose of DataONE is to provide an underlying infrastructure that facilitates data preservation and re-use for research. DateONE is built upon existing data centers leveraging the global investment in scientific data prevention. It creates a global federated data network focusing on interoperability solution and provides tools and services to enable new science and knowledge creation. It also facilitates an evolving community of practices enabled by DataONE cyberinfrastructure (CI) and is informed by best practices, exemplary data management plans, and tools that support all aspects of
the data life cycle.[19]

One of the most important principal components is the Member Nodes. Member Nodes are existing or new data repositories that install the DataONE Member Node application programming interfaces (API). A Member Node can be located anywhere on the earth. Together they provide a wide range of data and functionality. Member nodes are selected based on evaluation criteria that include factors such as diversity of data holdings, readiness to participate, community leadership, and resource availability. Participation in DataONE simplifies user access to those valuable resources through a set of service interfaces. It also helps to ensure long-term access to the information through the use of a persistent unique identifier for all data and metadata. [19]

8.2 Geoportal server

Geoportal Server is a free, open source software product from ESRI. Their server provides geospatial services such as discovery and use of geospatial resources, including datasets, rasters, and web services. It helps companies and organizations manage and publish metadata for their geospatial resources so users can discover and connect to them. The Geoportal Server supports standard-based clearinghouse and metadata discovery applications. Their server was released under the Apache license so third-party companies and developers can customize and redistribute the software.

8.2.1 Key Features

Organizations can manage a geospatial website with the Geoportal Server. Data producing users can register their resources so data consuming users can discover and consume. The website does not duplicate the registered resources. Instead, it stores and catalogs the metadata of the resources and the information to access those resources.
• **Data cataloging**

Geoportal Server uses the OGC (Open Geospatial Consortium) compliant CS-W 2.0.2 service for online cataloging. Their server inventories all the metadata of the registered resources in a geoportal catalog service.

• **Data Publishing**

Data producers publish their resources to Geoportal Server by registering the resources’ metadata with the server’s catalog service. Their server also include simple tools for generating and registering metadata.

• **Data Discovery**

The Geoportal Server provides an easy-to-use interface for users to discover and access resources registered with the portal to use in their own projects. The web application ensures that users always access to the most up-to-date and best quality resources available.

8.3 **GeoNetwork Open Source**

GeoNetwork Open Source is an open source Java web application that provides spatial information management and catalog services. The current version is 2.1.0. As stated in GeoNetwork portal:

> "GeoNetwork opensource is a standardized and decentralized spatial information management environment, designed to enable access to geo-referenced databases, cartographic products and related metadata from a variety of sources, enhancing the spatial information exchange and sharing between organizations and their audience, using the capacities of the internet. This approach of geographic information management aims at facilitating a wide community of spatial information users to have easy and timely access to available spatial data and to existing thematic maps that might support informed decision making... The main goal of the GeoNetwork opensource software is to improve the accessibility of a wide variety of data, together with the associated information, at different scale and
from multidisciplinary sources, organized and documented in a standard and consistent way."

The project was originally developed by the FAO (Food and Agriculture Organization) of UN (United Nations) in 2001 to archive and publish the geographic data produced within FAO. It used metadata content available from the legacy system that are transformed into the ISO 19115 metadata standard, which was only a draft metadata standard then. Later, the WFP (World Food Programme) of UN joined the project and, with its contribution, the first version of the software was released in 2003. The catalog system was established in FAO and WFP. The system was based on ISO 19115:DIS metadata standard and embedded the Web Map Client Intermap that supported OGC compliant web services. Distributed searches were possible using the Z39.50 standard catalog protocol. At that time it was decided to make the project a FOSS (Free and Open Source Software) to allow the geospatial community to benefit from the development results and to contribute to the further advancement of the software.[9]

The second version was released in 2004 after UNEP (UN Environment Programme) joined the group. The new version allowed users to work with multiple metadata standards (ISO 19115, FGDC and Dublin Core) in a transparent manner. It also allowed metadata to be shared between catalogues through a caching mechanism, improving reliability when searching in multiple catalogues.

GeoNetwork opensource is the result of the collaborative development of many contributors. These include the Food and Agriculture organisation (FAO), the UN Office for the Coordination of Humanitarian Affairs (UNOCHA), the Consultative Group on International Agricultural Research (CSI-CGIAR), The UN Environmental Programme (UNEP), The European Space Agency (ESA), and many others. Support for the metadata standard ISO19115:2003 has been added by using the ISO19139:2007 implementation specification schema published in May 2007. The release also serves as the open source reference implementation of the OGC Catalogue Service for the Web (CSW 2.0.2) specification. Im-
provements to give users a more responsive and interactive experience have been substantial and include a new Web map viewer and a complete revision of search interface.[9]

GeoNetwork Open Source is a Java web application which can installed and operated in a MS-Windows, Linus, or Mac system with support of Java and Apache Tomcat platform. It consists two web applications. The first one is the portal, which let users search the metadata catalog and manage the resources. The second one is a web map client to view and represent geospatial data.

8.4 ArcGIS/ArcCatalog

ArcGIS is a powerful desktop GIS system produced by ESRI. It is used for computerized mapping and spatial analysis. It provides functionality to capture, store, query, analyze, display and output geographic information.

ArcGIS consists of two major components: ArcMap and ArcCatalog. ArcMap is the main mapping application which allows users to create maps, query attributes, analyze spatial relationship and layout final projects. ArcCatalog is the application for users to organize spatial data in a local system. Users can search, preview, add spatial data to ArcMap, manage metadata, and set up geocoding.

ESRI provides three licensing levels: ArcView, ArcEdit, and ArcInfo. Each level provides increasing levels of functionality. ArcView comes with basic mapping and analysis tools. ArcEditor provides all ArcView capability plus additional processing and advanced editing. ArcInfo provides all the ArcEditor capability plus advanced analysis and processing.

UNR IGT Knowledge Center has a license from ESRI. UNR students and faculty members can access the ArcGIS system using Citrix Receiver software which can be downloaded from UNR website.
ArcCatalog can organize resources created by ArcGIS or stored in another format. Users can create general metadata with ArcCatalog. ArcCatalog also provides limited metadata validation. It converts a metadata file into XML format and validate the XML file. It does not provide metadata translation.

### 8.5 Feature based comparison

We have compared four related services and application in the geospatial metadata field to MCA in this chapter. Table 8.1 show the feature based comparison of them.

From this table it can be seen that our proposed MCA approach not only covers a lot of functionality, which is not addressed in totality by any of the other approaches, but also fills existing capability gaps in a couple of areas. Specifically, MCA deals with metadata translations and translations with web services.

Note that the information in Table 8.1 is not meant to be comprehensive evaluation of the other approaches and tools. They have significant other strong capabilities and are more developed (by teams) than MCA. In Table 8.1 only certain features related to some extent to MCA are included. Further more, our knowledge of those other approaches and tools is limited. We only position MCA in the landscape of metadata-related approaches.

### 8.6 Summary

The main purpose or function of all other systems is online geospatial metadata cataloging. They all address the geospatial data sharing issue by providing searching and cataloging services so users can easily discover high quality, accurate, and relevant data for their own projects. They also help data producing
Table 8.1: Related work feature based comparison

<table>
<thead>
<tr>
<th>Features</th>
<th>Application Name</th>
<th>MCA</th>
<th>DataONE</th>
<th>Geoportal Server</th>
<th>GeoNetwork Open Source</th>
<th>ArcGIS/ArcCatalog</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mapping</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Data View</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Data Analysis</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Search</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Metadata Creator</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Metadata Validation</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Metadata Translation</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Metadata Submission</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Validation with WS</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Translation with WS</td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Big Data Support</td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Online Resource Validation</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Resource Management</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>OAuth 2.0 Support</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>User Acc Management</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>User Registration</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Community Support</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Multiple Standard Support</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

organizations make their valuable resources available to data consuming customers without having to develop sophisticated software to administrate and manage large datasets. Those systems all provide services such as metadata submission and creation in some degree, but users have to login to their system and manually create metadata. When the quality of the datasets is large, in most cases, it is not very practical to submit metadata that way. Also, if data needs to be used in different local systems which support different metadata standards, those services still do not provide much help. MCA will bridge those gaps by providing metadata translation and offering easy integration to local systems to generate desired metadata standards on the fly.
Chapter 9

Future work and conclusions

In this dissertation, we have proposed and implemented a geographical metadata community adaptor (MCA) using XSLT technologies to address geographic metadata interoperability and compatibility issues. The MCA is a more methodological approach that fills existing gaps in these areas. It is supported by a fully functional Internet application, which helps users to search for available geographic resources for their own projects, to view detailed metadata information, and to connect to the actual data providers to download or purchase data. It also provides a set of features and functions for data providing users to submit existing metadata to the Tri-State Data Hub portal for online cataloging and indexing, to validate metadata files, to create metadata files for their data, and to translate existing metadata files among popular standard formats so the metadata can be available to variants of software systems. Geographic datasets are large in quantity by nature. That makes it necessary to automate some of the processes such as metadata validation, submission, and translation. [61, 62, 21] The web service component of MCA aimed to solve the automation issue, by including features such as metadata submission, validation, and translation. With the web services, data providers can accomplish those time-consuming and tedious tasks programmatically instead of manually to save time and to prevent data entering errors.
9.1 Future works

Due to the complexity of the proposed method and its supporting tools, MCA is still an ongoing project. More work should be done to make its software more robust, more user-friendly, and to serve more users. Here is a list of important features and functions that should be added in the future:

A. User Interface. As we mentioned in the User Interface chapter, a good UI is important. It represents the software and it could be the factor to determine the success or failure of a software product. A recent example illustrates this point. Apple introduced its flagship product iPhone 5s in September 20, 2013. The iPhone 5s come in 3 colors: silver, gold and space gray. Within hours of launch, the gold color iPhone 5s was sold out worldwide while other 2 colors were available. We all know that iPhone 5s has the same configuration and same features, but the color of the UI design made all the difference. In addition, UI design is an evolving process. A good UI today could and will become obsolete tomorrow, while a bad UI today might gain popularity tomorrow. So, for MCA, we need keep improving the UI as new tools and new technologies become available.

B. Hardware support. MCA is currently hosted in a Windows 7 system, which resides inside a Macbook pro with Virtual Box. The computing power and RAM are very limited. To make MCA more robust, we need to implement it in a dedicated Windows server with a faster processor and more RAM support. Also, for security concerns, the MS SQL server should be installed in a separate machine.

C. Migration tools. For MCA’s web services, a software tool should be developed to assist data providers in integrating MCA into their own system. Tasks in this area include both web service request generating and a software installer.

D. A system to support future metadata standards. MCA currently supports five of the most
popular geospatial metadata standards (ISO 19115, ISO 19115-2, FGDC, Dublin Core, and GEMINI(Data)). We should have a module to add future metadata standards to the supported standard list when available.

E. Development for other platforms such as Linux. MCA is developed with MS Visual Studio in a Windows platform, so it only works on a Windows server with ASP.NET. It should have a version for other platforms as well, as the LAMP servers are growing in quantity due to the fact that it is open source and it is free.

F. DataONE support and integration. DataONE is a fast-growing organization in the industry. It is a worldwide project with huge number of clients and user base. MCA should integrate the DataONE GUI into the system to make it easier to join DataONE network.

G. Community platform. In order for MCA to support future metadata standards and improve over time, it should be an open source to community. This way, community members can contribute to MCA by coding for the project and by providing help and consulting services to other members.

H. Metadata repairing. Invalid metadata is a key issue for resource sharing. Addition of metadata repairing will further enhance the usefulness of MCA and will benefit more metadata resource providers and consumers.

I. Translation validation. MCA provides a community contribution platform for its members. Members can submit new translations for new metadata standards. To verify the correctness of submitted translation, MCA should add intelligent translation validation capabilities.
9.2 Conclusions

Global warming is a fact. We are living in a warmer world. Sea levels are rising. These phenomena are accelerating. Scientists across the globe are researching the issues to find solutions to such problems. To support their research, geospatial data must be collected and provided. While more and more organizations are collecting valuable data constantly using various equipment and software, as well as storing data in different formats and into different medias, data interoperability and compatibility has become a vital issue. Therefore, many geographic metadata standards have been developed and used to represent the data. Those metadata are, then, used to catalog and index the data into clearinghouses and online cataloging systems. Problems data providers face include missing metadata, invalid metadata, and metadata compatibility for local systems.

In this dissertation, we have directly addressed those issues and have proposed and implemented the MCA approach. MCA will help data providers with services to submit, create, validate metadata files, and translate metadata files from one metadata standard into another standard. It also provides users with web services to automate some of the related time-consuming and tedious tasks. In particular, the web service component of MCA includes metadata submission, validation, and translation.
Appendix A

Acronyms

**ANSI** American National Standards Institute.
**API** Application Programming Interface.

**BBOX** Bounding BOX.

**CA** Certificate Authorization.
**CAS** Community Authorization Service.
**CSDGM** Content Standard for Digital Geospatial Metadata.
**CSISS** Center for Spatial Information Science and Systems.
**CSW** Catalogue service—Web Profile.

**DAAC** Distributed Active Archive Center.
**DC** Dublin Core.
**DNS** Domain Name Server.
**DPSS** Distributed Parallel Storage System.
**DRS** Data Replication Service.
**DTD** Document Type Definition.

**ebRJM** ebXML Registry Information Model.
**ECHO** EOS Clearing HOuse.
**ECS** EOSDIS Core System.
**EGA** Enterprise Grid Alliance.
**EOS** Earth Observing System.
**EOSDIS** EOS Data and Information System.
**ESRI** Environmental Systems Research Institute, one of several developers of GIS software.
**ETM** Enhanced Thematic Mapper.
FGDC  U.S. Federal Geographic Committee.
FTP  File Transfer Protocol.

GCMD  Global Change Master Directory.
GGF  Global Grid Forum.
GIS  Geographic Information System.
GRAM  Grid Resource Allocation and Management.
GRASS  Geographic Resource Analysis Support System.
GSI  Grid Security Infrastructure.
GT  Globus Toolkit.
GUI  Graphical User Interface.

HPSS  High Performance Storage System.
HTTP  Hypertext Transfer Protocol.

ISO  International Organization for Standardization.

KVP  Key Value Pair.

LFN  Logical File Name.
LRC  Local Replica Catalog.

MDS  Monitoring and Discovery Service.
MODIS  Moderate-resolution Imaging Spectroradiometer.
MPI  Message Passing Interface.

NASA  National Aeronautics and Space Administration.
NCCP  Nevada Client Change Portal.
NDVI  Normalized Difference Vegetation Index.
NFS  Network File System.

OASIS  Organization for the Advancement of Structured Information Standards.
OGC  Open Geospatial Consoaium.
OGF  Open Grid Forum.
OGSA  Open Grid Service Architecture.
OGSA.DAI  OGSA Data Access and Integration.
OGSI  Open Grid Service Infrastructure.
ORSS  Optimized Resource Selection Service.
OSPF  Open Shortest Path First.
OWL   Ontology Web Language.
OWS   OGC Web Services Initiative.

PFN   Physical File Name.

RAID  Redundant Array of Independent/Inexpensive Disks.
RDMS  Relational database management system.
REST  Representational State Transfer.
RFT   Reliable File Transfer.
RLI   Replica Location Index.
RLS   Replica Location Service.
RSVP  Resource ReSerVation Protocol.

Schema A document that describes, in a formal way, the syntax elements and parameters of a web language.

SDK   Software Development Kit.
SOA   Service Oriented Architecture.
SOS   Sensor Observation Service.
SQL   Structured Query Language.
SRB   Storage Resource Broker.

TCP   Transmission Control Protocol.
THREDDS Thematic Realtime Environmental Distributed Data Services.
TLS   Transport Layer Security.

UML   Unified Medical Language.
URI   Uniform Resource Identifier.
URL   Uniform Resource Locator.
UUID  Universally Unique Identifier.

VWCS  Virtual Web Coverage Service.

WCS   Web Coverage Service.
**WCTS**  Web Coordinate Transformation Service.
**WFS**  Web Feature Service.
**WICS**  Web Image Classification Service.
**WMS**  Web Map Service.
**WMT**  Web Mapping Testbed.
**WSDL**  Web Services Description Language.
**WSRF**  Web Service Resources Framework.

**XML**  eXtensible Markup Language.
**XPath**  XML Path Language.
**XQuery**  XQuery is a query and functional programming language that is designed to query collections of XML data.
**Xsan**  Mac OS X Storage Area Network.
**XSL**  Extensible Stylesheet Language.
**XSLT**  Extensible Stylesheet Language Transformations.
Appendix B

Sample metadata documents

A sample ISO 19115 dataset metadata (excerpt only, the actual document is 8 pages long) is shown in Listing B.1

```xml
<?xml version='1.0' encoding='UTF-8'?>
  <gmd:fileIdentifier>
    <gco:CharacterString>5E508CBA-C3CC-47D9-B862-F54CCD42BD3F</gco:CharacterString>
  </gmd:fileIdentifier>
  <gmd:language>
    <gco:CharacterString>en</gco:CharacterString>
  </gmd:language>
  <gmd:hierarchyLevel>
    <gmd:MD_ScopeCode codeSpace='ISO TC211/19115' codeList='http://www.isotc211.org/2005/resources/Codelist/gmxCodelists.xml#MD_ScopeCode' codeListValue='dataset'>
      dataset</gmd:MD_ScopeCode>
  </gmd:hierarchyLevel>
  <gmd:contact>
    <gmd:CI_ResponsibleParty>
```

```xml
```
<gmd:organisationName>
   <gco:CharacterString>UNR</gco:CharacterString>
</gmd:organisationName>

<gmd:contactInfo>
   <gmd:CI_Contact>
      <gmd:address>
         <gmd:CI_Address>
            <gmd:electronicMailAddress>
               <gco:CharacterString>qiping@aol.com</gco:CharacterString>
            </gmd:electronicMailAddress>
         </gmd:CI_Address>
      </gmd:address>
   </gmd:CI_Contact>
</gmd:contactInfo>

<gmd:CI_Contact>
   <gmd:role>
      <gmd:CI_RoleCode codeSpace="ISOTC211/19115" codeList="http://www.isotc211.org/2005/resources/Codelist/gmxCodeslists.xml#CI_RoleCode" codeListValue="pointOfContact">pointOfContact</gmd:CI_RoleCode>
   </gmd:role>
</gmd:CI_Contact>

</gmd:contact>

<gmd:CI_ResponsibleParty>
   ... ...
</gmd:CI_ResponsibleParty>

<gmd:role>
</gmd:role>

</gmd:CI_ResponsibleParty>

</gmd:contact>
... ...  
</gmd:MD_DataIdentification>  
</gmd:identificationInfo>  
<gmd:distributionInfo>  
</gmd:MD_Distribution>  
</gmd:distributionFormat>  
</gmd:MD_Format>  
</gmd:name>  
</geo:CharacterString>download </geo:CharacterString>  
</gmd:name>  
</gmd:version>  
</geo:CharacterString>1.1 </geo:CharacterString>  
</gmd:version>  
</gmd:MD_Format>  
... ...  
</gmd:distributionInfo>  
</gmd:dataQualityInfo>  
</gmd:DQ_DataQuality>  
</gmd:scope>  
</gmd:DQ_Scope>  
</gmd:level>  
</gmd:MD_ScopeCode codeSpace="ISOTC211/19115" codeList="http://www.isotc211.org/2005/resources/Codelist/gmxCodeLists.xml#MD_ScopeCode" codeListValue="dataset">dataset </gmd:MD_ScopeCode>  
</gmd:level>  
</gmd:DQ_Scope>  
</gmd:scope>  
</gmd:lineage>  
</gmd:LL_Lineage>  
</gmd:statement>  
</geo:CharacterString>lineage required. </geo:CharacterString>  
</gmd:statement>
Listing B.1: ISO 19115 dataset metadata sample (excerpt)

A FGDC metadata sample (excerpt only, the actual document is 6 pages long) is presented in Listing B.2.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<metadata>
  <idinfo>
    <citation>
      <citeinfo>
        ... ...
        <pubplace>Reno, NV USA</pubplace>
        <publish>University of Nevada, Reno</publish>
      </citeinfo>
      <pubinfo>
        <onlink>http://cse.unr.edu</onlink>
      </pubinfo>
    </citation>
    <descript>
      <abstract>This FGDC metadata document is generated for the Tri-State Data Portal debugging purpose only.</abstract>
      <purpose>Web application debug and testing.</purpose>
    </descript>
    <timeinfo>
      <sngdate>
        <caldate>20130710</caldate>
        <time>Unknown</time>
      </sngdate>
    </timeinfo>
  </idinfo>
</metadata>
```
<timeinfo>
  <current>publication date</current>
</timeinfo>
<timeperiod>
  <status>
    <progress>In work</progress>
    <update>As needed</update>
  </status>
</timeperiod>
<bounding>
  <westbc>−119.8192</westbc>
  <eastbc>−119.8117</eastbc>
  <northbc>39.5333</northbc>
  <southbc>39.528</southbc>
</bounding>
<keywords>
  <theme>
    <themekey>economy</themekey>
    <themekey>structure</themekey>
    <themekey>transportation</themekey>
  </theme>
  <place>
    <placekey>Geographic Names Information System</placekey>
    <placekey>Reno Nevada</placekey>
  </place>
  <proctest>
    <procdesc>Manually generated data</procdesc>
    <procdate>Not complete</procdate>
  </proctest>
Listing B.2: FGDC metadata sample (excerpt)
Bibliography


[2] Global Warming, Source:


http://www.dcc.ac.uk/resources/briefing-papers/standards-watch-papers/what-are-metadata-standards


[16] Geoportal Server, Retrieved July 13, 2013 from ESRI web site:
http://www.esri.com/software/arcgis/geoportal

[17] Arlow, J. and Neustadt, I., UML 2 and the Unified Process: Practical Object-Oriented Analysis and

[18] DateONE website <http://www.DataONE.org>


Workshop on Sensor Network Applications, 2011

Proceedings of the 21st ISCA International Conference on Software Engineering and Data Engineering
(SEDE 2012), 2012, pp. 173-178


Change Data Infrastructure for Nevada.” Advanced Information Systems Engineering Workshops


http://sourceforge.net/projects/xsltsl/


http://www.quepublishing.com/articles/article.asp?p=27800&rl=1


Web site: http://www.w3.org/XML/


http://xml.apache.org/xalan-c/

[33] *Xalan-Java version 2.7.0*, (n.d.). Retrieved May 15, 2013 from the Apache XML Project Web site:

http://xml.apache.org/xalan-j/


df59714e92fa1b31ae


Press, New York, NY, 11-16.


