PROVENEANCE TRACKING IN A COMMONS
OF GEOGRAPHIC DATA

By
David B. McCurry
B.Sc. University of New Brunswick, 2006

A THESIS
Submitted in Partial Fulfillment of the
Requirements for the Degree of
Master of Science
(in Spatial Information Science and Engineering)

The Graduate School
University of Maine
December, 2007

Advisory Committee:
Harlan Onsrud, Professor of Spatial Information Science and Engineering, Advisor
Kate Beard-Tisdale, Professor of Spatial Information Science and Engineering
Silvia Nittel, Professor of Spatial Information Science and Engineering
Advancement in digital archiving technologies provides researchers with a multitude of methods for sharing their research and data digitally with others. However, when acquiring data from others directly or indirectly the law often imposes an assumption of copyright in the dataset acquired. This creates a difficult legal situation affecting future use and creation of derivative works from the data. A digital commons may be defined as a shared resource in which creators of contributed materials (data) grant a legal right for all others to use the material under the provisions of an open-access license. This thesis hypothesizes that an approach can be developed that automates the intellectual property rights and licensing management for contributors to a commons of geographic data. In addition, an approach can be developed such that contributors receive credit for their data, and the source of the data can be identified even through generations of alteration and reuse. The technological approach presented centers around embedding both visible and hidden identifiers in contributed data files. The identifiers, which remain intact
through reuse and derivatives of the data, display the open-access licensing provisions to future users of the data. The research also involves using the identifiers to retrieve standards-compliant metadata records for the data and preserve links between different versions of the data. Because contributors of data are more likely to receive credit and recognition for their contributions of data when used by others and legal clarity is increased, this new approach may provide incentives to contributors to more openly share data and thereby provide greater benefits to the community through its availability.
ACKNOWLEDGEMENTS

This thesis would not have been possible if it were not for the guidance and support of my family and friends, my colleagues in the department of Spatial Information Science and Engineering at the University of Maine, and particularly my previous education at the University of New Brunswick.

I would like special acknowledgement to Dr. Harlan Onsrud, his knowledge of the CGD system, and his patience and help with this entire thesis. To the other members of the project team, Jim Campbell, Eeva Hedefine, Marilyn Lutz, Silvia Nittel, and Kenton Williams who all played a part in the implementation of the system as well as a great influence on progress towards open geospatial data.

I would like to finally acknowledge the funding of my thesis, which came through the Institute of Museum and Library Services.
# TABLE OF CONTENTS

ACKNOWLEDGEMENTS ........................................................................................................... ii

LIST OF TABLES ................................................................................................................ viii

LIST OF FIGURES .............................................................................................................. ix

Chapter

1. INTRODUCTION ......................................................................................................... 1
   1.1 Motivation .................................................................................................................. 1
   1.2 Research Goal and Hypothesis ................................................................................ 2
   1.3 Scope of Thesis ......................................................................................................... 4
   1.4 Organization of the Remaining Chapters ................................................................ 5

2. DESIGN AND ARCHITECTURE .............................................................................. 6
   2.1 Introduction ............................................................................................................. 6
   2.2 System Back End .................................................................................................... 6
   2.3 System Front End .................................................................................................... 7
   2.4 The Contribution Process ...................................................................................... 8
      2.4.1 Owner Information ........................................................................................... 9
      2.4.2 License Selection ............................................................................................. 10
      2.4.3 File Upload .................................................................................................... 10
      2.4.4 File Information ............................................................................................. 11
      2.4.5 Location ........................................................................................................ 11
      2.4.6 Data Information ........................................................................................... 11
      2.4.7 Summary ....................................................................................................... 12
   2.5 Conclusion .............................................................................................................. 13
LIST OF TABLES

Table 1: Creative Commons Licenses ................................................................. 10
Table 2: CGD Input Formats ............................................................................... 17
Table 3: Seven CGD Interchange Formats ......................................................... 19
Table 4: Unused Bits in CGD Interchange Formats .............................................. 23
Table 5: Jittering Example .................................................................................. 44
Table 6: Jittering Encoding Scheme ..................................................................... 45
LIST OF FIGURES

Figure 1: CGD System Architecture ......................................................... 7
Figure 2: CGD Contribution Process .......................................................... 9
Figure 3: Sample Dataset Summary ........................................................... 12
Figure 4: Sample Checksums from Copied Images .................................... 22
Figure 5: Creative Commons Text License ............................................... 28
Figure 6: Creative Commons Embedded License ....................................... 29
Figure 7: 24 Bit Image Encoding ............................................................... 32
Figure 8: Binary Identifier Encoding ......................................................... 33
Figure 9: CGD Frequency Hopping Encoding .......................................... 37
Figure 10: Sample Visible Identifier ......................................................... 40
Figure 11: Encoded Line Segment ............................................................. 47
Figure 12: Extraction Process Flow ......................................................... 53
Chapter 1

INTRODUCTION

This chapter introduces the concepts of a Commons of Geographic Data (CGD) and of provenance tracking in a CGD. It provides a motivation for the research presented in the remainder of the thesis. This chapter also states the goal of the research, which is to enable users of the CGD to contribute their data, license it to the public commons or public domain, and to provide incentives to contribute. Finally, this chapter presents an outline of the remainder of the thesis.

1.1 Motivation

The digital library is rapidly expanding, and with the growth of archiving technology and software (e.g., DSpace, Fedora), there exists a need to create more specialized archives to store data as well as its documentation. A major difficulty with creating a repository for data is the assumptions associated with copyright of the data. While individual data items may be considered facts and are not copyrightable, data sets typically meet the requirements for copyright. Unless a digital work is noted to be under specific licensing provisions, the owner or creator of that work holds the copyright. This means that the burden is on the user of the data to gain permission for use to integrate the work with their own, or to create derivatives of the work.

Initiatives such as the Creative Commons and Science Commons have created open-access licenses in order to provide users with a means for future use of digital works for most purposes without the legal need to request case-by-case permission. These open-access licenses allow a creator of a digital work to dedicate
the work to the public domain or to provide open-access licensing provisions that retain copyright but place the work in a legal commons environment in which most uses are allowed without asking for permission. In this way, the assumption of copyright is overridden by the Creative Commons license, as long as the restrictions imposed by the selected open-access license are visibly noted on the work.

1.2 Research Goal and Hypothesis

The goal of this thesis is to focus on the discipline of GIS and geospatial data, and provide an open-access repository for datasets, available to all users. This is based on the conceptual Public Commons for Geospatial Data (Narnindi, 2003). In particular, this thesis focuses on lineage tracking of contributed datasets.

In the context of the CGD Framework, provenance tracking is the process of taking each dataset contributed to the system, and using a combination of both hidden and visible identifiers to mark the file in order to display that it is licensed under a Creative Commons license. The system will then monitor subsequent datasets contributed to the system, search for these identifiers, leave these intact, and retrieve metadata from the repository.
If identifiers are found for one or more files in a contributed work, this means the contributed work is a derivative product. In this case the system automatically perpetuates in the new derivative file any restrictions imposed by the previous license (Onsrud et. al. 2004).

Key questions that need to be addressed include:

• How can we maintain a reliable link between a dataset and the open access license that affirmatively states its allowable uses even after a potential user has downloaded the file?
• How can we embed several types of identifiers in different types of geospatial data while providing standard formats back to the users and while avoiding significant compromises to the quality of the data?
• How can we effectively monitor through generations of reuse and altered derivatives the license on a dataset that has been dedicated to the public domain or public commons?

This thesis addresses the above questions, and the following hypothesis is formed:

_A provenance tracking methodology can be developed and incorporated into a combined technical and legal approach that overcomes the legal rule that requires gaining permission to use the data of others on a case-by-case basis._

The remainder of this thesis describes detailed approaches that address the research questions above and the hypothesis.
1.3 Scope of Thesis

The Commons of Geographic Data has come from a conceptual model explained in (Narnindi, 2003), and has been partially implemented as a proof-of-concept. Likewise, the provenance tracking algorithms and methods further explained in this thesis have been implemented as a proof of concept, and have not been tested in a production environment.

For the purposes of designing a scheme for tracking open-access licenses through different types of geospatial data, this document provides a sufficient explanation and a partial implementation of the methods explained. These methods have not been tested on a large public audience, and may or may not be suitable for a production environment without significant modification.

This thesis explains the legal approach taken towards licensing geospatial datasets to the public commons. The document also provides a detailed technical explanation of the use of hidden and visible identifiers in the monitoring of this license, the design of the system, and the design of a contribution process for expert and non-expert GIS users.
1.4 Organization of the Remaining Chapters

Chapter 2 presents an overview of the architecture of the Commons of Geographic Data System, including a software specification of the system, and the design and implementation of the contribution process. Chapter 3 explains the provenance tracking approach and provides an overview of the particular steps in tracking a dataset in the CGD. Chapter 4 explains the first method being used to track the provenance of a contributed dataset, which is hiding identifiers in the header space of several file types. Chapters 5, 6, and 7 focus on different types of geospatial data and explore the methods used to embed identifiers in each separate category of data. Chapter 8 describes the automated process of discovering the range of hidden identifiers and exposing them to the system. Chapter 9 finally concludes the thesis, assesses the extent to which the hypothesis statement was achieved and provides future suggestions for work in both lineage tracking and the development of the CGD.
Chapter 2
DESIGN AND ARCHITECTURE

2.1 Introduction

This chapter presents an overview of the Commons of Geographic Data (CGD) system. It describes the architecture of the server, the software used to implement the CGD system, and the rationale for choosing the software. This section then explains the process for contributing a dataset into the CGD archive, the rationale for this process, and how it affects lineage tracking. All of this is in line with the objectives of the CGD design process, which is to create a system based on open-source software that will remain open, extendable, and redistributable.

2.2 System Back End

The CGD system is built completely using open-source software. At its core, the development server operates using version 2.6 of the Linux kernel. The server’s operating system is Debian Linux, which is licensed under the GNU General Public License (GPL). The GPL is also the license of the Linux kernel (GNU-GPL, 1991).

The CGD site is served using the second version (2) of Apache’s http server. Apache licenses its software under its own license (Apache 2004).

The final part of the backend is the PostgreSQL database management system, which serves the CGD database. PostgreSQL also has the spatial extension, PostGIS built to work with it. Both PostgreSQL and PostGIS are licensed under the GNU-GPL (Postgres 2007).
2.3 System Front End

The front end of the CGD system is served to clients using PHP (PHP Hypertext Preprocessor) for the scripting language. The output of the site conforms to the W3C XHTML 1.0 Transitional standard and W3C CSS 2.0 standard. The World Wide Web Consortium (W3C) maintains both of these standards openly, and PHP is an open-source language released under the GNU GPL. Figure 1 depicts the architecture of the CGD system.

Figure 1: CGD System Architecture
2.4 The Contribution Process

The process of a user contributing a file to the CGD forms the basis of the file types allowed, the extraction of information about the file, and the generation of metadata both from the users, and from existing files in the database (Campbell, et. al. 2005). The design of the contribution process was based on the following principles:

- Each user should create standards-compliant (ISO 19115) metadata for their dataset
- The entire contribution, from start to finish, should not exceed ten minutes for a user familiar with the system and readily achievable by a novice user
- The interface should be tab-based to clearly indicate progress to the user

The contribution process contains six major steps for each dataset contributed. Figure 2 is a flow chart of the contribution process in the CGD. The six major steps are shaded and are described below. As we can see, on upload, some identifiers are extracted. These are simply the identifiers that are used to check for duplicates of a file, and are discussed in chapter 4. The remainder of the lineage tracking algorithms is run after the contribution of a file because of their time consumption. This allows the users to contribute files in much less time.
Figure 2: CGD Contribution Process

2.4.1 Owner Information

The first of the six-step process is the owner information. This simply asks the user to confirm whether or not they are the owner of the dataset, or if they are contributing on behalf of an organization. The system pulls the owner information from the database, and automatically populates the fields with it. The owner then confirms the information and can proceed to the next step.
2.4.2 License Selection

There are three licenses a user can choose from in the CGD framework, which follow (Creative Commons, 2007):

- **Public Domain**
 任何人可自由使用我正在贡献的数据，用于任何用途。这些数据成为公共领域的部分。

- **Attribution**
 任何人可自由使用我正在贡献的数据，用于任何用途，但需给予我创建该数据的信用。

- **Attribution Share-Alike**
 任何人可自由使用我正在贡献的数据，用于任何用途，但需给予我创建该数据的信用，并且，如果从这些数据创建的衍生作品，这些衍生作品需要在相同的条件下使用。

<table>
<thead>
<tr>
<th>License</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Public Domain</td>
<td>任何人可自由使用我正在贡献的数据，用于任何用途。这些数据成为公共领域的部分。</td>
</tr>
<tr>
<td>Attribution</td>
<td>任何人可自由使用我正在贡献的数据，用于任何用途，但需给予我创建该数据的信用。</td>
</tr>
<tr>
<td>Attribution Share-Alike</td>
<td>任何人可自由使用我正在贡献的数据，用于任何用途，但需给予我创建该数据的信用，并且，如果从这些数据创建的衍生作品，这些衍生作品需要在相同的条件下使用。</td>
</tr>
</tbody>
</table>

### Table 1: Creative Commons Licenses

The above three licenses and statements appear to the user for the license selection in the CGD contribution process. The user then must select one of these licenses for the dataset they are contributing. A user also can select a default license, which is retrieved from the CGD database.

2.4.3 File Upload

Next, the system asks for the dataset to be uploaded into the system. The user simply selects the file that contains their dataset, and it is then uploaded onto the CGD server.
2.4.4 File Information

Most of the metadata is assigned to the file in this step. The user enters a small descriptive name for the file, the spatial representation of the file, and the topics that the file refers to. All of these are derived from the ISO 19115 standard. The user also must confirm the filename and format in this step.

2.4.5 Location

To select a location of the dataset, the user can enter the upper-right and lower-left longitude and latitude of the dataset (in decimal degrees), or they are presented with the more user-friendly option of using a map interface. The map interface allows the user to search and zoom to a location, and they can select a button to draw a box for an approximate bounding box of their dataset. In a more advanced implementation, for future development, each submitted file would be searched, bounding coordinates would be derived, automatically supplied, and the bounding map shown to the user as an option to accept or reject.

2.4.6 Data Information

The final step of the process is to input some remaining fields in the metadata. The first is the dates that the dataset refers to (not the date of upload). Finally, the user adds keywords to make the dataset searchable, and a short narrative summary containing any additional information about the dataset.
2.4.7 Summary

The user is then presented with a summary of their dataset. If something needs correction, they can return to any previous step, correct the information, and return to the summary. A sample summary appears below:

<table>
<thead>
<tr>
<th>Dataset #722</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Dataset title:</strong> aerial photo of orono</td>
</tr>
<tr>
<td><strong>Coordinates:</strong> Top left- Latitude: 44.837613 Longitude: -68.621807</td>
</tr>
<tr>
<td>**Bottom right- Latitude: 44.813262 Longitude: -68.585415</td>
</tr>
<tr>
<td><strong>Date Contributed:</strong> 2007/08/06</td>
</tr>
<tr>
<td><strong>Responsible Party:</strong> Personal Ownership</td>
</tr>
<tr>
<td><strong>Name:</strong> David McCurry</td>
</tr>
<tr>
<td><strong>Email:</strong> <a href="mailto:cmccurry@spatial.maine.edu">cmccurry@spatial.maine.edu</a></td>
</tr>
<tr>
<td><strong>Address:</strong> 5/11 Boardman Hall</td>
</tr>
<tr>
<td>Orono, ME 04469</td>
</tr>
<tr>
<td>USA</td>
</tr>
<tr>
<td><strong>Phone:</strong> (207) 581-2175</td>
</tr>
<tr>
<td><strong>Topic Categories:</strong> imagery/BaseMaps</td>
</tr>
<tr>
<td><strong>Abstract:</strong> This is an aerial photo of Orono, Maine. It is poor resolution, but covers the entire municipality.</td>
</tr>
<tr>
<td><strong>Dates the data refers to:</strong> 2006-03-14 - 2006-03-14</td>
</tr>
<tr>
<td><strong>License Choice:</strong> No restrictions</td>
</tr>
<tr>
<td><strong>Keywords:</strong> photo</td>
</tr>
<tr>
<td>orono</td>
</tr>
<tr>
<td>maine</td>
</tr>
</tbody>
</table>

Figure 3: Sample Dataset Summary
2.5 Conclusion

This section has presented an overview of the design and implementation of the CGD system, and the contribution process. Each step of the contribution process was explained in detail, and the metadata gathered in this process is then archived in the CGD and available for future usage of the dataset. Each dataset is licensed using one of the three Creative Commons licenses described, and will be embedded in the dataset for future users to clearly see.
Chapter 3
PROVENANCE TRACKING

3.1 Introduction

This chapter delineates the process of provenance tracking in the CGD system. For clarity, provenance tracking in the CGD framework refers to the process of embedding identifiers into files, extracting identifiers on subsequent use, and tracking the license provisions on the files through this reuse. This section presents an overview for the handling of files and formats contributed by users as well as the methods for then categorizing those files and embedding identifiers in order to offer them back out to others.

Typically, when embedding identifiers into a variety of media files, there are two systems that are normally discussed. The first method that is commonly employed, particularly in digital multimedia, is Digital Rights Management (DRM) system. The main concept behind DRM systems is to restrict the usage of the file so that a user has restrictions on sharing the file with others, and a user can only make a certain number of copies of a certain file. This approach, which involves embedding additional information into a file, is not appropriate for the CGD framework because there aren't any restrictions placed on the use of the file (OpenGIS Consortium 1997, Royan 2000).

The second approach that is commonly taken is to use a container system. Container systems are software that packages all file formats into a container, then to be edited; the files must be extracted before being packed back into a container for contribution back to the system (Lucas, 2005). This adds another layer of software
to the process that only makes files more difficult to edit and less accessible for the end users.

Rather than using one of the mentioned methods, the CGD has borrowed concepts from both approaches and applies it to all contributed files. This approach has been broken down into four steps: 1) identify the file type, 2) convert the file into common formats, 3) embed the identifiers, and 4) extract the identifier on subsequent use of the file.

The embedding and extraction of the hidden identifiers are discussed in detail in the remainder of the document; however, an overview of the entire process is given here.

3.2 Identification

When a user contributes a file to the CGD system, no restrictions are initially imposed on the types of files that are accepted into the system. Since there are several thousand formats of data, it would be impossible to maintain and support that many different file types. Instead, the approach is taken to separate the data into distinct categories and maintain a subset of formats from each of those categories. The top-level partitioning of the datasets is done using the ISO19115 standard for spatial representation. This metadata field separates the data into one of the following categories:
This top-level enumeration gives the CGD an overview of what to accept.

Video files, although they can have a spatial element to them, are not integral datasets to the CGD system so identifiers will not be developed for video files. Likewise, TIN (Triangulated Irregular Network) files will not have identifiers developed for them, since TINs are commonly contained in vector files. Stereo Model files are not accepted into the CGD system because they lack common, open file formats.

After this separation, the CGD system accepts text / table files, vector files, and raster files. The following formats for each file type to be accepted contributions are:
<table>
<thead>
<tr>
<th>Text / Table</th>
<th>Vector</th>
<th>Raster</th>
</tr>
</thead>
<tbody>
<tr>
<td>txt</td>
<td>shp</td>
<td>bmp</td>
</tr>
<tr>
<td>text</td>
<td>rt</td>
<td>jpg</td>
</tr>
<tr>
<td>tex</td>
<td>dwg</td>
<td>jpeg</td>
</tr>
<tr>
<td>latex</td>
<td>dxf</td>
<td>tif</td>
</tr>
<tr>
<td>ps</td>
<td>fld</td>
<td>tiff</td>
</tr>
<tr>
<td>eps</td>
<td>dgn</td>
<td>gtiff</td>
</tr>
<tr>
<td>doc</td>
<td>gml</td>
<td>jp2</td>
</tr>
<tr>
<td>xls</td>
<td>mif</td>
<td>png</td>
</tr>
<tr>
<td>xml</td>
<td>tab</td>
<td>dem</td>
</tr>
<tr>
<td>htm</td>
<td>eoo</td>
<td>img</td>
</tr>
<tr>
<td>html</td>
<td>svg</td>
<td>mng</td>
</tr>
<tr>
<td>shtml</td>
<td>svf</td>
<td>pcx</td>
</tr>
<tr>
<td>dif</td>
<td>swf</td>
<td>bil</td>
</tr>
<tr>
<td>ppt</td>
<td>ai</td>
<td>bip</td>
</tr>
<tr>
<td>prn</td>
<td>arc</td>
<td>bsq</td>
</tr>
<tr>
<td>wpd</td>
<td>wmf</td>
<td>adrg</td>
</tr>
<tr>
<td>csv</td>
<td>vpf</td>
<td>sid</td>
</tr>
<tr>
<td>asc</td>
<td>vxp</td>
<td>ecw</td>
</tr>
<tr>
<td>dbf</td>
<td>ddf</td>
<td>pict</td>
</tr>
<tr>
<td>dat</td>
<td>dvi</td>
<td>hfa</td>
</tr>
<tr>
<td>pdf</td>
<td>dvips</td>
<td>mex</td>
</tr>
<tr>
<td></td>
<td>dvipdf</td>
<td></td>
</tr>
<tr>
<td></td>
<td>mat</td>
<td></td>
</tr>
<tr>
<td></td>
<td>nco</td>
<td></td>
</tr>
<tr>
<td></td>
<td>nc</td>
<td></td>
</tr>
</tbody>
</table>

Table 2: CGD Input Formats

3.3 Convert

Each category of files will be offered back to the users in a small subset of the formats above. The conversion of the input formats to the outputs will be done using existing open-source software packages. Each of the three categories is done in a similar manner, but using different software, which is explained below.
3.3.1 Text and Tables

Text and tables will be converted from input formats contained in table 1 to two separate output formats that are easiest to display licenses and identifiers in. Most of the formats in table 1 are simple plaintext (txt, html) formats and are straightforward to convert.

The binary files, such as Microsoft Word documents, cannot be parsed in the same method as plain text. These formats each have open source libraries that can convert the files to html format. A detailed listing of these libraries is contained in Appendix A.

Once converted to plain text, all text and table files contributed to the CGD will be offered back to the users in html (Hypertext Markup Language) and plaintext formats. These formats both have open specifications as ASCII text files.

3.3.2 Vector

Vector formats will be converted using the OGR Simple Feature Library. The OGR library supports a variety of vector formats, which are detailed in Appendix B. The OGR library will be applied to the inputs in table 1, and the system will offer vector files back to the users in ESRI Shapefiles (shp), which is an open format. Users can also download the spatial SQL for a vector file as well (from PostGIS).
3.3.3 Raster

Quite similar to the vector formats, raster formats will be converted using the Geospatial Data Abstraction Library (GDAL). GDAL supports all of the formats listed in table 1, as well as several more not supported by the CGD system. A detailed listing of GDALs support is contained in Appendix B. After conversion, the CGD will offer users their files back in JPEG, PNG, and GeoTiff formats. These formats all have open specifications available.

3.4 Embedding and Extracting

After the conversion of the files, each CGD dataset is in one of seven formats, each of which is openly specified for manipulation. This simplifies the process of embedding identifiers into each file, as well as extracting the identifiers of subsequent usage. The processes of embedding identifiers are explained in Chapters 4, 5, 6, and 7. Detection and extraction is explained in Chapter 8. Table 3 shows the seven CGD interchange formats. All files submitted to the CGD are converted to one or more of these formats and identifiers are embedded only within these formats and offered back out to other users.

<table>
<thead>
<tr>
<th>File Types</th>
<th>File Formats</th>
</tr>
</thead>
<tbody>
<tr>
<td>Text and Tables</td>
<td>html, plaintext</td>
</tr>
<tr>
<td>Raster</td>
<td>jpg, png, tiff</td>
</tr>
<tr>
<td>Vector</td>
<td>shp, sql</td>
</tr>
</tbody>
</table>

Table 3: Seven CGD Interchange Formats
The identifiers created for each dataset will be explained further in the sections on embedding and extracting the identifier. These identifiers will be in the form of integers, which are unique to the CGD system. Each dataset contributed to the system is assigned a unique identifier from the database (in ascending order), and this is the identifier that will be embedded into the dataset.

3.5 Discussion

This chapter has discussed the process of lineage tracking in the CGD system. Although the approach is relatively simple, its major shortcoming is quite easy to notice upon contribution of files. The system only supports a small subset of files, but attempts to support the most commonly contributed geospatial datasets. In the future, the CGD aims to refine these formats in order to best serve the end users of the repository, both on the input and output format side. The system does aim to maintain open formats to the end-users, as these formats are supported in most GIS application software.
Chapter 4

PREVENTING DIRECT COPYING OF DATASETS

4.1 Introduction

This chapter discusses the method that the Commons of Geographic Data (CGD) framework uses to discourage direct copying of contributed datasets. When a dataset is contributed to the CGD, it is licensed to the public commons, and thus is simple for others to copy and claim ownership of.

Because of this result, the CGD has incorporated methods in order to detect and prevent direct copying without traversing through the entirety of the lineage tracking algorithms that are discussed further. This problem is addressed using the openly specified formats that the CGD is offering back to users.

4.2 File Formats

As discussed in the previous section, the CGD is providing users of datasets with a standard set of formats. The advantage of this is that direct copies will keep the format of the file intact. Using this knowledge, we can detect exact copies of works using two different methods- file checksums and embedding an id in the header space of the file.

4.3 File Checksums

A popular method of verifying that a file has downloaded completely and properly is to use cryptographic hash functions to generate a string that condenses the input of the file into a much smaller string. One of the most popular hash programs is md5, which generates a 32-character string based on any input. The design of md5 is to make it collision resistant, such that there is a very small chance
that two input strings will produce the same output. This implementation makes it useful for comparing two files to one another.

Using md5, we can compare the strings that are generated by two files. If they come out the same, there is a very large chance that the files are exactly the same.

\[
\text{MD5 (1.jpg) } = \text{9304a2e0913f2c64e0809cadca762e06} \\
\text{MD5 (2.jpg) } = \text{9304a2e0913f2c64e0809cadca762e06}
\]

**Figure 4: Sample Checksums from Copied Images**

4.4 File Headers

After comparing the checksums of the two files, if they are indeed the same, we can check the file header to see if an identifier exists. Before we discuss the identifier, a brief digression into file headers is necessary.

Each file contributed to the CGD will reside in a particular directory reserved for datasets on the main server. After the conversion to standard formats and the embedding of identifiers, these files will not be overwritten except under extenuating circumstances. These files will be offered back to the users “as-is” after this transformation has been performed. The formats of the files have already been discussed, but the importance is to note that these are all open formats, so the specifications are available and well documented. Each file contains the actual data of the file (for example, each pixel of an image is defined within the file), but each file also has to have a specific amount of metadata that travels along with it. This includes the format of the file, the size of it, and any other information specific to the
particular format. All of the formats used by the CGD have a number of unused bits in the file headers, which can be manipulated to store an identifier. Details of the unused bits for each file type are below:

<table>
<thead>
<tr>
<th>Format</th>
<th>Unused Bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>ESRI Shapefile (.shp)</td>
<td>Bytes 4-20</td>
</tr>
<tr>
<td>JPEG (.jpg, .jpeg)</td>
<td>Comment field: FFEEnn + id</td>
</tr>
<tr>
<td>Portable Network Graphics (.png)</td>
<td>IEND field: 73 69 78 68 + id</td>
</tr>
<tr>
<td>GeoTiff (.gtiff, .tif, .tiff)</td>
<td>GeoKey field: 34737 + id</td>
</tr>
<tr>
<td>Plaintext</td>
<td>N/A</td>
</tr>
<tr>
<td>Hypertext (.htm, .html)</td>
<td>N/A</td>
</tr>
</tbody>
</table>

Table 4: Unused Bits in CGD Interchange Formats

The above table has the details of the unused bits in the header file. As we can see, in shapefiles, we have bytes 4-20 that are unused (ESRI 1999). The raster formats are a bit different, as they usually have markers to identify fields. The JPEG uses a comment marker, followed by an identifier (JPG 1992), PNG can package the identifier with an IEND marker (World Wide Web Consortium 2003), and finally GeoTiff has a GeoKey field that can contain the identifier (GeoTiff 2000).
4.5 Discussion

This section discussed the method that the CGD is employing to prevent direct copies of datasets that already exist in the archive. When a duplicate dataset is submitted to the commons, the metadata record for the dataset will be retrieved and displayed to the user. If the dataset belongs to the same owner as the duplicate, the metadata can be edited. If the dataset belongs to a different creator or owner, then the metadata is displayed, but no edits are permitted.

While the provenance tracking would address this issue in a different method, this “near-instant” detection of copies adds some functionality to the CGD that is useful to end-users, for example, if someone at a particular organization has contributed data, this algorithm would detect identical sets immediately, and results in less redundancy in the system.

4.6 Results

When the above methodology is applied to a limited set of input files for testing, it was found that by using md5 checksums, the detection of directly copied files was accurate. This was simply due to the small chance of discovering a collision in the md5 algorithm, which was very unlikely for a small test case.

When addressing the identifiers hidden in the spare bits, the test results were more mixed. After embedding, if the file remained unchanged, bits were able to be detected, however, if a file was opened by an external program (such as an image or shapefile editing utility), the bits were often changed and unable to be recovered.
These limited test cases verify both the speed and shortcomings of the above algorithms. For the purposes described above, immediate detection of directly copied files, the methods were verified by testing.
5.1 Introduction

This chapter begins discussing the methods of embedding hidden and visible identifiers into the files in the Commons of Geographic Data. The first category of files is text and tabular data, which differs from the binary raster and vector formats simply because the plaintext of the data can be extracted.

Historically, text steganography is much different from binary formats because of the limitations imposed by text-based data. Several techniques exist for hiding messages in text, including shifting characters, and actually inserting characters within the data (Johnson 1998). The CGD does not attempt to hide data, as this would create a compromise of the quality of the data, which is one of the important components of geospatial data. Instead, the CGD will approach the problem only by using visible identifiers.

5.2 Text Extraction

Chapter 3 presented an overview of the major types of formats in the CGD framework. This included two different types of output, both of which are plaintext formats. Although these are the only two formats offered back to the users of the system, several different types of binary and plain text files are enabled for contribution to the system. The numbers of formats that are accepted into the public commons are determined by 1) open-source libraries to extract the text from the binary format, and 2) the needs of the end-users of the system. If a specific format is common and not implemented in the CGD framework, an extraction algorithm for
the formula will be investigated and eventually included in the system. The first iteration of the system has attempted to identify common formats that would be contributed, and the formats are as follows:

- txt
- text
- tex
- latex
- ps
- eps
- doc
- xls
- xml
- htm
- html
- shtml
- dif
- ppt
- prn
- wpd
- csv
- asc
- dbf
- dat
- pdf

5.3 Visible Identifiers

Each text file, after its conversion into plain text and html, will have a visible Creative Commons license identifier appended to the file. These identifiers are available openly through the Creative Commons, and contain an html link to the license on their server, along with a logo, and an XML license. An example license follows:
Figure 5: Creative Commons Text License

There are two parts to the above license- the first is the HTML portion, which simply displays a link and image to the Creative Commons site. The second portion, which is a little more obfuscated, is the XML portion. This allows crawlers and search engines to index the license of the document, along with the conditions of the license, which are enclosed in the <permits> and <requires> XML tags (Creative Commons 2007). On an actual document, this identifier will appear as follows:
5.4 Invisible Identifiers

Plain text and HTML files cannot be treated as raster and vector datasets simply because of the specifics of the formats. While raster and vector are binary formats, these two formats are simpler, containing only ASCII text. While it would be possible to hide identifiers within the actual text, this would result in losing a significant precision or loss of quality of the data.

Because of this formatting requirement, there will only be visible identifiers in these text files. The users of the system will be responsible for following the licenses displayed on the file. Potential violation will instead be tracked by employing a pattern matching process as described in Chapter 8. The methods for tracking these licenses also will be discussed in Chapter 8.
5.5 Summary

This section discussed the identifiers used in the text formats in the CGD framework. These formats are unique because they will only contain a single type of identifier, which will be visible. The possibility of a hidden identifier was discussed, however, this is decided against in order to preserve the quality and accuracy of the data contained in these text files. The visible license will be kept intact by users choosing to edit and redistribute the file. The CGD has also developed methods to track whether or not the license is being followed, and these are discussed in Chapter 8.
Chapter 6
WATERMARKING RASTER DATA

6.1 Introduction

This chapter presents the techniques employed by the Commons of Geographic Data to track the provenance of raster data contributed to the system. As mentioned before, the system will accept a subset of the formats of raster data, convert each raster file to the interchange formats, and then identifiers will be embedded within the interchange format files before they are offered back to the users.

Each image will have a visible identifier appended to it, much like text files. This is the final step to marking each image. Each raster dataset will also have two types of hidden identifiers embedded as well. These hidden identifiers originate from traditional digital multimedia watermarking. The first is least-significant bit (LSB) embedding, and the second is using frequency hopping watermarks.

These two methods will be discussed in detail, from the least computationally intensive (LSB) to the more complex process of frequency hopping.

6.2 Rationale

The primary reason for using two distinct methods for processing the raster dataset is the nature of digital watermarks. In order to be considered an effective watermark, first, the mark must make a permanent alteration to the file, that is, it must survive on subsequent uses of the file. Second, the watermark should not make significant alterations to the quality of the data (Hernandez 1999, Ruanaidh et. al.)
1996). For instance, a watermark should not be overwhelming to the actual image, and should preserve the information and quality of the raster data. Finally, the watermark should be able to withstand attack. A more detailed analysis of such attacks is summarized in (Craver et. al. 1998), but in particular, datasets in the CGD would be vulnerable to “print-edit-scan” attacks. A “print-edit-scan” attack is simply the process of printing a hard copy of the image (on paper) and rescanning the image using a digital scanner. Some steganographic methods are susceptible to this type of attack, and because of its simple nature, the CGD aims to overcome it.

6.3 Identifiers using Least Significant Bit

Least Significant Bit (LSB), or noise-addition watermarking is available in several commercial software products, and is proven to work with a variety of raster formats. The CGD incorporates LSB watermarking on all of its supported raster file formats, which are JPEG, TIFF, and PNG images. These images all have open specifications and are easy to work with on a bitwise level using imaging toolkits. A more detailed specification of the LSB technique is available in (Petitcolas et. al. 1999). The portions of the algorithm used for the CGD are below.

6.3.1 Embedding

The main idea behind LSB watermarking is simple. Each image contains so many pixels, each of which is represented by a binary code, which indicates the color of the pixel. For example, a 24-bit image has the following code for a single pixel:

<table>
<thead>
<tr>
<th>R</th>
<th>1</th>
<th>1</th>
<th>1</th>
<th>1</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>G</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>B</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Figure 7: 24 Bit Image Encoding
The figure depicts the separate colors of the image, each of which has an 8-bit representation. Depending on the format of the image, there may be more values, but the CGD encoding does not alter anything other than the pixel color values.

6.3.2 CGD LSB Encoding

The CGD algorithm is based on the above schemes for using LSB encoding. In order for the watermark to be preserved and extracted successfully, an input image has to have the following properties

- The width of the image has to be at least 256 pixels wide
- The height of the image must be at least 64 pixels

These two properties are necessary for the embedding of the identifier. Each identifier is composed of a bitwise ASCII representation of the string “CGD,” followed by a 32-bit integer containing the numerical representation of the dataset from the CGD archive. Each of these identifiers is unique. In total, each identifier will be 80 bits. There will be 64 pixels between identifiers both vertically and horizontally in the image. This will mean that for a minimum image size of 256 x 64 pixels, there will be a minimum of four identifiers in the image. This will be sufficient to locate and extract the identifier on subsequent contribution. A detailed description of the identifier is depicted in the figure below.

<table>
<thead>
<tr>
<th>ASCII “c”</th>
<th>ASCII “g”</th>
<th>ASCII “d”</th>
<th>Identifier 124</th>
</tr>
</thead>
<tbody>
<tr>
<td>0110 0011</td>
<td>0110 0111</td>
<td>0110 0100</td>
<td>0000 0000 0000 0000 0000 0111 1100</td>
</tr>
</tbody>
</table>

Figure 8: Binary Identifier Encoding
6.3.3 Extraction

Similar to encoding the image, extraction is quite a simple process. The image will be scanned for the encoding of the ASCII string “CGD,” and then scan ahead 64 pixels in both directions from the first located identifier. If another identifier is intact, then the image indeed came from the CGD system and the identifier will be looked up. If not, the system will continue to search for additional identifiers. If a significant portion of any image is left intact, the identifier will be extracted.

6.3.4 Discussion

From the above description, we can see that this is a relatively simple watermark, but is sufficient for many image types. However, LSB watermarking may not survive small changes to the image, and may not survive “print-edit-scan” attacks. The LSB watermark also has a minor affect on the end result of the dataset. That is, after LSB watermarking is applied to an image, there is a small increase in the size of the file, and a small decrease in the quality of the image due to the changed pixel values. This change in the file size and quality, however, is negligible due to the benefits of LSB watermarking. Both the embedding and extraction of the watermarks take place quickly in comparison to more robust methods, and the watermark can survive through several generations of the image.

In the CGD implementation, LSB watermarking does not work for either very small images, or derivatives and mash-ups of small images. From a technological perspective, it becomes very complex to track images when the file size is so small that embedding an identifier is nearly impossible. However, from a
legal perspective, a derivative containing such a small part of an image such that there is no identifier may not have the modicum of creativity to preserve the ownership and license of the image in the first place. This legal perspective solves the problem from a technological point of view, and simplifies the resulting implementation (Onsrud 2007).

6.3.5 Results

In the CGD framework, LSB testing was done using a small amount of regular and geospatial image sets. There was no significant difference in the result between the two types, so these results are independent of the type. Images that were of the proper size (256x64 pixels) were able to successfully retain a watermark through subsequent upload of the same file. When a large enough portion of an image was cropped to bear at least two watermarks, the image was able to be identified as a derivative work.

When images were converted to other formats, the watermark was lost for all changes of format. Portions of watermarks were unable to be detected.

6.4 Identifiers using Frequency Hopping

Frequency hopping, or spread spectrum watermarking addresses the shortcomings of LSB extraction by surviving these discussed attacks. Frequency hopping has been proven to survive “print-edit-scan” attacks (DigiMarc Technologies 2002). Another advantage of frequency hopping is that it can also survive compression into other formats, such as lossy JPEG compression (Hartung and Kutter 1999).
The main premise behind frequency hopping watermarking is the conversion of the image from the spatial to the frequency domain, then the application of the watermark, then the conversion back to the spatial domain. These watermarks are close to impossible to see without advanced analysis of the image (Cox et. al. 1995). A detailed explanation of the algorithm is available in (Hartung and Kutter 1999) as well as additional methods of embedding identifiers. The CGD has chosen a specific implementation which is detailed below.

6.4.1 Embedding

The first step to embed an identifier in the frequency domain of the image is to convert it to the frequency domain. The two traditional methods of this are the Discrete Cosine Transform (DCT), and the Fast Fourier Transform (FFT). In the context of the CGD framework, all of these transforms are calculated for only two dimensions.

The CGD implementation uses a DCT rather than an FFT in order to avoid some of the overhead caused by complex numbers. After the transformation into the frequency domain, the objective is to find a series of values that can be replaced by a watermark, apply the watermark, and then reverse the transform back to the spatial image domain.
As we can see, for each series of values, v, this series is combined with the watermark, denoted x, which creates a new series v'. v' is then inserted back into the original image.

The equation for v' is:

\[ v'_i = v_i (1 + ax_i) \]

Where a is a scaling factor. For the purposes of this proof-of-concept, a is just a constant at a value of 1.

The important property of these equations is that they are all reversible for values of v not equal to 0. This is an assumption that has been proven to work in practice, so this is not an issue with the CGD algorithm (Cox et. al. 1995).
6.4.2 CGD Frequency Hopping Encoding

To create the series for x, the CGD uses an encoding similar to the LSB encoding described. Each series for x will have the bitwise ASCII representation of the string “CGD” followed by the 32-bit numerical representation of the dataset. This will be then applied to each series v in the frequency spectrum. The selection of v will be based on the number of repetitions of v in the image. This is difficult because midrange frequency elements must be selected. The algorithm finds the highest frequency element and the lowest frequency element of the given length, and then finds the closest match to the midrange of those values.

The identifier is then applied to the values, and the reverse transformation occurs.

6.4.3 Extraction

Similar to the embedding of the identifier, the midrange frequency element must be identified. In this case, since the identifier is already intact in the image we can’t just pick an arbitrary frequency element. The midrange must first be identified, then all frequencies above and below the midrange must be scanned and the inverse equations applied to find the “CGD” ASCII string. Once this is found, the other watermarks can then be located easily by matching it to the original. This can then be extracted and the identifier confirmed.

6.4.4 Discussion

Unlike LSB watermarks, frequency hopping has been shown to survive “print-edit-scan” attacks, and the watermarks are spread at intervals throughout the
image. These watermarks not only survive the attacks, but remain intact for usage of only a significant portion of the image, and the license will thus be enforced for such derivative works.

The trade-off for this better survival is a much more computationally intensive algorithm. This simply means that images in the CGD will first be checked for information in the file header, then for LSB watermarks, and finally for Frequency Hopping watermarks. This way, only new datasets and ones without any LSB watermarks will be processed intensely.

6.4.5 Results

Frequency hopping watermarks were testing in the same manner as the LSB watermarks. First, the difference was noted between formats and similarly disregarded, as no discernible difference was found in testing.

For the limited test, the watermarks were successfully embedded in several image sizes, the least of which being the smallest LSB size (256 x 64 pixels). From the original images, all watermarks were extracted successfully.

Using crop operations, the chances of detecting a watermark significantly decreased for images sizes below the minimum mentioned size. If an image crop was larger than this size, almost all instances detected a watermark.

The final test for segment encoding was compression into both PNG and JPG formats. Because of the probability of detecting a watermark in a small image size
was very low, this was tested on a variety of large images (>500x500) pixels. The watermark was successfully detected after both types of image compression.

6.5 Visible Identifiers in Raster Datasets

Very similar to the plaintext datasets, each raster dataset will contain a visible identifier that references the applicable Creative Commons License. Since quality of the dataset is a concern, the license icon will not overlap with any of the image. Instead, rows will be inserted at the bottom of the image containing the license icon and a URL, which shows the full text of the license. An example is shown below (NASA, 2007):

![Sample Visible Identifier](http://creativecommons.org/licenses/by/3.0/)

*Figure 10: Sample Visible Identifier*
6.6 Summary

This section discussed the two methods employed by the CGD for embedding and extracting hidden identifiers in raster datasets. The advantages and disadvantages of both methods were discussed, and the reasoning for using more than one distinct method was discussed.

If we refer to the test results of both algorithms, it has been shown for a small subset of images that the algorithms work as intended. LSB was able to survive through small alterations to the files, namely cropping re-contributing the file, while the frequency hopping watermarks were able to survive more complex transformations, such as JPEG and PNG compression.
Chapter 7

WATERMARKING VECTOR DATA

7.1 Introduction

Vector watermarking is dissimilar to raster data because of the nature of the files. Pixel-based data can have a watermark embedded sequentially in the pixels. Vector data, however, differs in the sense that it is not discrete so the embedding must take place in the data itself. This makes hiding an identifier more difficult, however, the methods presented here are similar in structure to the raster watermarks.

Like the raster watermarks, vector data will have three different types of identifiers embedded in it- the first is a visible identifier, followed by two types of invisible watermarks: jittering watermarks and line segment encoding.

7.2 Rationale

Vector steganography has been rapidly expanding due to the success of commercial GIS applications. There are several schemes that exist that make small modifications to the topologies of the map, small geometric transformations, and modifications to the actual features in the map (Lopez 2002, Huber 2002, Ohbuchi et. al. 1997).
The difficulty with many of these approaches is that they rely on certain features to be present. Some methods identify four-sided polygons and split into two triangles or leave them intact in order to watermark them. This heavily relies on the dataset having a certain number of squares in order to bear a specific sized watermark.

The CGD approach to watermarking vector datasets is to use the least number of features to create this watermarking. This results in the ability to watermark many more datasets, as well as less impact on the accuracy and quality of the data.

7.3 Identifiers using Jittering

The first type of watermarking in vector datasets is known as jittering. This, similar to the LSB encoding discussed in Chapter 6, is a faster algorithm for both embedding and extracting, however, will not survive most transformations that would be performed on the dataset.

The concept of jittering takes advantage of the coordinates used for displaying vector data. Each vector dataset either uses a global or local coordinate system to create its topology, which is then displayed to the user. Jittering takes all of the coordinates and modifies them slightly in order to hide a watermark in the numerical values. The table illustrates the result of jittering:
Before Jittering | After Jittering
---|---
1.141 | 1.1412351235
2.235 | 2.23512351235
-3.412 | -3.41212351235
1.352 | 1.35212351235
4.235 | 4.23512351235

Table 5: Jittering Example

As we can see, after the addition of the identifier to each point in the table, the value of the point is modified slightly. However, the value each point is modified by less than 0.001 units, and does not affect the quality of the data in a significant manner.

A more detailed explanation of jittering is available in (Ohbuchi et. al. 1997), with more specifications and applications. The CGD implementation details are below.

7.3.1 Embedding

Embedding the identifier into a vector dataset is a short process. As mentioned in chapter 3, each vector dataset will be converted into spatial SQL before being converted back into shapefiles that will be available for download. During this transformation, each element of the shapefile will have an attribute for its geometry, which will contain all of the coordinate reference for each point, line, and polygon in
the shapefile. The embedding algorithm appends an identifier into half of these coordinates, while appending near-zero values to the remaining coordinates.

7.3.2 Encoding

The encoding of the jittering watermarks is very similar to the binary encoding scheme used in the previous section; however, integers are used rather than binary to encode the number. An example of how this is encoded is below:

<table>
<thead>
<tr>
<th>ASCII “c”</th>
<th>ASCII “g”</th>
<th>ASCII “d”</th>
<th>Identifier 124</th>
</tr>
</thead>
<tbody>
<tr>
<td>0110 0011</td>
<td>0110 0111</td>
<td>0110 0100</td>
<td>0000 0000 0000 0000 0000 0111 1100</td>
</tr>
<tr>
<td>99</td>
<td>103</td>
<td>100</td>
<td>124</td>
</tr>
</tbody>
</table>

Table 6: Jittering Encoding Scheme

7.3.3 Extracting

Extracting the watermark has a similar simplicity to the embedding process. When a shapefile is contributed to the CGD archive, it will be first converted into spatial SQL. The file is then scanned for identifiers by first looking for the zero-value appends, which would be unique and usually truncated by most applications. If these are intact, the archive then parses the non-zero appended points in order to find an identifier. Once the identifier has been matched, it is extracted and the process for enforcing the license is instantiated.
7.3.4 Discussion

At first glance, we can see that this algorithm has similar drawbacks to the LSB algorithm for raster datasets. Most transformations result in the point data being rounded, and the jittering watermark will be lost. Once again, the simplicity and runtime of the algorithm make it useful for a lot of purposes. This algorithm, if able to extract an identifier successfully, would allow us not to traverse the segment encoding algorithm, which would save time for the system.

7.3.5 Results

Jittering was tested on a small set of sample data. A generous amount of coordinate data was included in each starting file in order to preserve the most data. Using Jittering, watermarks were successfully applied to the data and able to be extracted without modifications.

Cropping data and creating mashups of different input files preserved Jittering watermarks to the extent that at least two original coordinates from the source dataset were mashed together.

Jittering watermarks were not successfully detected for small amounts of data, coordinate projections, or significant changes to the topology of the file.

7.4 Identifiers using Segment Encoding

Segment encoding has a higher degree of complexity than the jittering watermark, but is able to survive much higher degrees of transformation. Segment encoding uses a binary string much like the strings used in the raster datasets to encode a watermark, which is then embedded into line segments in the file. The line
segments can then be read back into the system to extract the watermark. The CGD implementation of this encoding is described below.

7.4.1 Embedding

To embed the segment encoding, the system must first compile a list of the line segments that have a long enough length to be encoded. After compiling the list, each segment is broken up into much smaller segments that have one of two sizes, one that represents 0 and another that represents 1. These smaller segments then form a binary string containing the ASCII representation for CGD, followed by a 32-bit integer that is the identifier for the dataset.

7.4.2 Encoding

The lines will be encoded using the binary scheme describe above, with a fixed space between each 0 and 1 in the segment. An example is shown below for the same encoding given in the table in section 7.3 for the string “CGD”. The encoded segment is on top of the original line.

![Encoded Line Segment](image)

Figure 11: Encoded Line Segment

7.4.3 Extracting

In order to extract the watermark, the system must search for the “CGD” ASCII watermark to check for an identifier. The dataset is parsed for a series of small line segments that have a fixed length between each small segment. The lengths of each segment are used to compute a binary sequence that is compared against the ASCII representation. Once a CGD watermark is found, the identifier
can be extracted, and then a second identifier is needed to match the values, which can then be used to enforce the license of the dataset.

7.4.4 Discussion

Segment encoding will survive most types of geometric transformations to a vector dataset because it uses its own topology. The simplest way to overcome this watermark is to reconnect all the line segments broken by this encoding, which would be difficult to do manually and time consuming. The algorithm to search for the watermarks and then extract is much more complex than jittering, however, this is justified by its ability to survive through generations of reuse of the dataset.

7.4.5 Results

Segment encoding was tested using the same methods as jittering. A sample of test files (with generous amounts of segments) were prepared and used for the testing. The size of each gap is predetermined based on the size of the segment, and worked for encoding each segment inputted, which should create a sample of reasonably small segments.

When the watermarks were embedded successfully, they could immediately be extracted without issue. The watermarks were also able to survive cropping if two complete watermarks remained intact in the resulting file (just the portions of the segments bearing the watermark, which for testing were the leftmost and bottom portions of the segment).

The segment-encoding algorithm also demonstrated survivability after coordinate projections. Segment encoding could be overcome in the test
environment only by deleting small portions of the segment that contained the watermark. In these instances, altering this data was very tedious and would be very difficult for large datasets, however, it was still possible.

7.5 Visible Identifiers

After hidden identifiers are embedded into the vector datasets, a visible identifier is also embedded in order to show the license of the dataset. These are simpler than the raster visible identifiers, and will just contain the text of the license. This text will be inserted into the dataset in the bottom corner like the raster by using a spatial SQL statement. An example is below:

```
INSERT into FEATURES (label, geometry) VALUES
('http://creativecommons.org/licenses/by/3.0 Creative Commons Attribution 3.0' #btmcorner);
```

This will result in a user able to refer to the license by the URL in the statement.

7.6 Summary

This chapter discussed the concept of watermarking vector datasets with hidden and visible identifiers. The two methods that the CGD uses for invisible identifiers were discussed, along with their algorithms for embedding the identifiers, encoding the identifiers, and extracting the identifiers on subsequent contributions of the dataset. The rationale for choosing these algorithms was discussed, and results in less transformation in the data and a higher quality to the resulting dataset. The chapter then concluded by discussing the simple method for displaying a visible license identifier in each vector dataset.
The results were discussed based on a small number of tests administered within the CGD system. The results were near expected; jittering was quick to apply and extract, and worked for large sets of unaltered data. Segment encoding was more hit or miss for cropping segments. Since these can be very long in many cases, getting a portion of the line did not work in many cases (since each segment only contained a single watermark). In future iterations, this issue will be addressed by spreading the watermark through an entire segment to ensure a better survivability.
Chapter 8

DISCOVERY AND EXTRACTION OF IDENTIFIERS

8.1 Introduction

This section delineates the processes by the Commons of Geographic Data (CGD) framework for discovering and extracting identifiers. This is important for all types of data to discover if the dataset is a derivative of one or several other datasets. The results of this process will also be discussed.

8.2 Text File Comparison

Text files were a special case of identifiers, and were discussed in Chapter 5 of this thesis. The difficulty with text files arises from the formats being ASCII and not binary, so that embedding a hidden identifier would result in significant changes to the content of the dataset, which compromises both the quality and accuracy of the dataset. It was decided that, when a text file is contributed to the public commons, the text would only have a visible license appended to the data to maintain the quality.

This makes getting rid of an identifier very easy, so that the dataset cannot survive attack if the user simply removes the license from the file. Although this is not the goal of the CGD, it is understood that some users may have malicious intent and this is a possibility.

In order to alleviate this difficulty, the CGD will incorporate a pattern-based comparison based on blocks of text in the file. This, like the other methods, will not be 100% accurate, but will prevent many cases of direct infringement, and track a
number of derivatives. The approach works simply: an algorithm chooses arbitrary blocks of text from the file of a fixed size (based on the size of the input text.) These blocks are then compared to same-size blocks in other text datasets in the CGD. For each match, more blocks will be checked, until a specified threshold is reached. For this proof of concept, the threshold for similarity has been established as 50%.

8.3 Extraction in the CGD Archive

When a file is contributed to the CGD archive, it will not immediately be available from the CGD for several hours. The reason for this is that the file must be converted and the identifiers embedded before additional users can download the file. During the process of this embedding, the files are also checked for identifiers that have previously been embedded. As discussed, the files will already be checked for direct copying upon contribution, so none of these methods are employed upon upload.
Figure 12: Extraction Process Flow
The figure shows an overview of the discovery process for the identifiers. The files are checked for direct copying during the contribution process, so the methods explained in Chapter 4 are not included in this diagram. As we can see, the input files are partitioned into the three categories, which then have each of their own lineage tracking algorithms applied. The text files have the most simple two algorithms, first the text is checked for an existing Creative Commons license, which could be extracted and allow the traversal to stop. If not, the text will be checked for similarity against the other datasets in the system. If it has such a similarity, the license of the parent dataset will be enforced.

The raster and vector have similar processes to the text. The two simpler identifiers are extracted first, and if found, the algorithm terminates after this extraction. This will save time for files that have not undergone transformations that destroy these watermarks.

After these simpler watermarks are checked, the more complex algorithms will be traversed to extract identifiers.

**8.3.1 Enforcing the Licenses**

If an identifier is extracted by any of the mentioned methods, the license of the originating dataset or datasets will be tracked. Since a dataset cannot be licensed under lesser license provisions, the strictest license will be enforced on the dataset. The dataset will have its license updated if necessary, receive a new identifier, and will reference the original dataset(s) as being a derivative work of those datasets.
8.4 Creative Commons License Identification Application

The CGD has also developed a desktop application for identifying if a dataset is uploaded into the Commons. This allows users to have a fast method of retrieving metadata for a file that exists in the public commons, checking the license of that dataset, and identifying the source of the data.

The CGD License Identification Application allows a user to drag-and-drop a dataset to the application. The application then uses the same algorithms mentioned above to check the file for an identifier, which will then be sent to the CGD archive. If the identifier exists in the archive, the server will respond with all of the metadata for the dataset, including the license. This allows the users to quickly retrieve information on a file without searching the archive or re-contributing a dataset.

8.5 Conclusion

This section has discussed the methods used in the CGD and a stand-alone application to discover hidden identifiers in datasets, extract those identifiers, and use them to retrieve metadata, enforce licenses, and inform users if the files exist in the CGD. The chapter also discussed the methods for checking ASCII text files for similar information, and identifying possible sources for text files by matching patterns in the text.
Chapter 9
RESULTS AND FUTURE WORK

9.1 Introduction

This section discusses the results of using lineage tracking in the Commons of Geographic Data and future work in lineage tracking, and in the CGD as a whole. Finally, the document will be concluded.

9.2 Provenance Tracking

This thesis has presented a method for tracking the provenance of geospatial datasets in the CGD framework. The details of each type of file are discussed, and the algorithms are outlined according to the current implementation in the CGD. The results of this implementation show a proof-of-concept, that in using the above algorithms in conjunction Creative Commons licenses, we can address the problem of monitoring access to datasets, tracking derivative works, and provide users incentive to contribute.

The hypothesis is restated here for clarity:

A provenance tracking methodology can be developed and incorporated into a combined technical and legal approach that overcomes the legal rule that requires gaining permission to use the data of others on a case-by-case basis.

The methodology as developed and implemented within the prototype CGD illustrates that embedding lineage tracking to support an open access legal approach is achievable. The CGD uses Creative Commons licenses to overcome the legal requirement that requires users to gain permission to use the data of others. We have shown that we can use a variety of identifiers, both hidden and visible to travel with the datasets to carry the license information. The design of the identifiers has been
described in detail, along with the shortcomings of how each of them work, and how they perpetuate through reuse of the dataset.

9.3 Derivative Tracking

An interesting result of this tracking is that the derivative works of the datasets are readily visible to all users of the data. That is, if a user contributes a particularly useful dataset to the CGD, that user can then track all users who have downloaded this dataset. After each user makes additions or changes to the dataset, it will be linked to the previous dataset on upload to the system. Each CGD dataset could also have an XML record that advanced Google and Yahoo searches can pick up the Creative Commons license and identify the datasets as open access files.

This means that a dataset can be tracked through its generations of reuse, and allow the users to see where and who has used their data, identify sources of quality data, and promote interdisciplinary cooperation on geospatial data.

9.4 Future Work

With respect to both lineage tracking and the CGD framework, there still remain many tasks. The CGD could be deployed on a distributed network among many library servers that report datasets back to a central controlling server. This server could then apply the identifier embedding and detection to a larger amount of datasets.

The CGD also could have more functionality such as peer review and better search, which already have specifications being developed.
Provenance tracking could be refined significantly to support more file formats, additional types of identifiers, and even more categories of files if need arises to include video and other forms of media in the CGD. Future work also includes tracking of Creative Commons licenses to propagate changes through the system based on work in the legal framework.

9.5 Conclusion

This thesis has discussed provenance tracking in a public commons for sharing geospatial data, the Commons of Geographic Data. The CGD framework provides users with incentives to contribute a dataset into an open archive, along with assurance of a monitored open-access license to travel with their data, once contributed.

The hypothesis formed was supported through a discussion of the framework along with a detailed discussion of algorithms used to embed both hidden and visible identifiers into contributed datasets. Future work was then discussed and the thesis concluded.
REFERENCES

Apache Foundation (2004). *Apache License, Version 2.0*
http://www.apache.org/licenses/LICENSE-2.0.html


http://geodatacommons.umaine.edu/about.php?about=wpapers

http://geodatacommons.umaine.edu/about.php?about=wpapers


Creative Commons (2007). *Choosing a License.*
http://creativecommons.org/about/licenses/


http://remotesensing.org/geotiff/spec/geotiffhome.html

59
http://www.gnu.org/licenses/old-licenses/gpl-2.0.txt

Proceedings of the IEEE 87(7): 1079-1107


# APPENDIX A: TEXT EXTRACTION LIBRARIES

<table>
<thead>
<tr>
<th>Format</th>
<th>Application</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>DOC</td>
<td>XtoXML</td>
<td><a href="http://sourceforge.net/projects/xtoxml/">http://sourceforge.net/projects/xtoxml/</a></td>
</tr>
<tr>
<td>XLS</td>
<td>XtoXML</td>
<td><a href="http://sourceforge.net/projects/xtoxml/">http://sourceforge.net/projects/xtoxml/</a></td>
</tr>
<tr>
<td>PPT</td>
<td>XtoXML</td>
<td><a href="http://sourceforge.net/projects/xtoxml/">http://sourceforge.net/projects/xtoxml/</a></td>
</tr>
<tr>
<td>RTF</td>
<td>vPDFc</td>
<td><a href="http://sourceforge.net/projects/vpdfc/">http://sourceforge.net/projects/vpdfc/</a></td>
</tr>
</tbody>
</table>
APPENDIX B: VECTOR AND RASTER FORMATS SUPPORTED

OGR Library

"ESRI Shapefile" (read/write)
"UK .NTF" (readonly)
"SDTS" (readonly)
"TIGER" (read/write)
"S57" (read/write)
"MapInfo File" (read/write)
"DGN" (read/write)
"VRT" (readonly)
"AVCBin" (readonly)
"REC" (readonly)
"Memory" (read/write)
"CSV" (read/write)
"GML" (read/write)
"PostgreSQL" (read/write)

GDAL Library

VRT (rw+): Virtual Raster
GTiff (rw+): GeoTIFF
NITF (rw+): National Imagery Transmission Format
HFA (rw+): Erdas Imagine Images (.img)
SAR_CEOS (ro): CEOS SAR Image
CEOS (ro): CEOS Image
ELAS (rw+): ELAS
AIG (ro): Arc/Info Binary Grid
AAIGrid (rw): Arc/Info ASCII Grid
SDTS (ro): SDTS Raster
DTED (rw): DTED Elevation Raster
PNG (rw): Portable Network Graphics
JPEG (rw): JPEG JFIF
MEM (rw+): In Memory Raster
JDEM (ro): Japanese DEM (.mem)
GIF (rw): Graphics Interchange Format (.gif)
ESAT (ro): Envisat Image Format
BSB (ro): Maptech BSB Nautical Charts
XPM (rw): Xll PixMap Format
BMP (rw+): MS Windows Device Independent Bitmap
AirSAR (ro): AirSAR Polarimetric Image
RS2 (ro): RadarSat 2 XML Product
PCIDSK (rw+): PCIDSK Database File
PCRaster (rw): PCRaster Raster File
ILWIS (rw+): ILWIS Raster Map
RIK (ro): Swedish Grid RIK (.rik)
PNM (rw+): Portable Pixmap Format (netpbm)
DOQ1 (ro): USGS DOQ (Old Style)
DOQ2 (ro): USGS DOQ (New Style)
ENVI (rw+): ENVI .hdr Labelled
EHdr (rw+): ESRI .hdr Labelled
PAux (rw+): PCI .aux Labelled
MFF (rw+): Atlantis MFF Raster
MFF2 (rw+): Atlantis MFF2 (HKV) Raster
FujiBAS (ro): Fuji BAS Scanner Image
GSC (ro): GSC Geogrid
FAST (ro): EOSAT FAST Format
BT (rw+): VTP .bt (Binary Terrain) 1.3 Format
LAN (ro): Erdas .LAN/.GIS
CPG (ro): Convair PolGASP
IDA (rw+): Image Data and Analysis
NDF (ro): NLAPS Data Format
LIB (ro): NOAA Polar Orbiter Level 1b Data Set
FIT (rw): FIT Image
RMF (rw+): Raster Matrix Format
USGSDEM (rw): USGS Optional ASCII DEM (and CDED)
GXF (ro): GeoSoft Grid Exchange Format
BIOGRAPHY OF THE AUTHOR

David McCurry was born in Lincoln, Maine in 1983. He graduated from Presque Isle High School in 2001. After high school, he moved to Fredericton, New Brunswick, Canada to pursue an undergraduate degree in Software Engineering. He earned his Bachelor of Science in Software Engineering degree in December 2005. David is a candidate for the Master of Science degree in Spatial Information Science and Engineering from The University of Maine in December, 2007.