

U.S. Department of the Interior
U.S. Geological Survey

The National Map Catalog Technical Discussion Paper

**The University of Minnesota's MapServer and Other
Open Source Software in *The National Map***

Louis Driber

Mid-Continent Mapping Center
January, 2005

This document is for internal use by USGS Geography Discipline personnel. It has not been reviewed for conformance with USGS editorial standards and has not been approved for formal publication. Any use of trade names is for descriptive purposes only and does not constitute endorsement by the US Government.

Table of Contents

<i>Abstract</i>	2
1 Introduction.....	2
1.1 Background.....	2
1.2 Hypotheses.....	2
2 Open Source and Public Licenses.....	3
3 Open Standards.....	3
4 Materials and Methods.....	4
4.1 Hardware and Software Configurations.....	4
4.2 Method.....	5
5 Results and Discussion.....	5
5.1 Functionality.....	5
5.2 Cost.....	7
5.3 Documentation.....	8
6 Conclusion.....	8
7 References.....	8
 Attachment A. Creating a MapServer Instance on Linux.....	 9

Abstract

A primary goal of *The National Map* is to ensure availability of nationally consistent and integrated geospatial data. Geographic information services, hosting distributed geospatial databases, are the technical basis for *The National Map*.

MapServer software, developed at the University of Minnesota, is an open source implementation that conforms to the Open Geospatial Consortium's (OGC) Web Map Service (WMS) specification. Initial results of an on-going USGS investigation show MapServer to be a powerful and robust system. MapServer has been demonstrated to be a sound, low-cost, alternative to proprietary systems, and can effectively function to serve geospatial data to *The National Map*.

1 Introduction

Distributed Web Map Servers (WMS) function as the primary vehicle for serving geospatial data to *The National Map*. The MapServer software (<http://mapserver.gis.umn.edu>), developed at the University of Minnesota, is an open source, OGC-compliant, WMS application which can provide a robust and cost-effective alternative to commercial WMS applications. As an open source server application, MapServer does not tie the user to a particular vendor or software developer.

1.1 Background

The USGS is the owner, maintainer, and provider of several critical national geospatial data layers currently available in *The National Map*. Examples of such coverages include elevation, hydrography, transportation, and boundary data layers. These and other coverages hosted by the USGS are being served from ESRI ArcIMS map server implementations. Similarly, most of the USGS data partners contributing geospatial data to *The National Map* are also serving their data from ArcIMS implementations. Collectively, of the 951 data layers¹ currently available in *The National Map*, all but 33 are being served from ArcIMS map server implementations. ArcIMS software is proprietary, that is, the source code associated with the application is not publicly available and the application is tied to a particular vendor.

In September 2004 the USGS Mid-Continent Mapping Center started an investigation to identify effective alternatives to proprietary WMS software applications and evaluate the feasibility of open source WMS implementations from the point of view of an organization that serves, and partners with agencies that serve, geospatial data of broad appeal to the public through *The National Map*.

1.2 Hypotheses

1. Web Map Services implemented with Open Source software, as opposed to proprietary software, are suitable for serving data to *The National Map*.
2. Such systems can be implemented on inexpensive low-end PC hardware, yet be efficient and robust.

¹ As of January, 2005

2 Open Source and Public Licenses

The USGS has publicly expressed its commitment to supporting the development and implementation of open source software and geospatial standards. The development of open source software tools and adherence to geospatial standards is viewed as a step in the right direction for partnering with public and private organizations in support of implementation of *The National Map*.

Open Source Software - Open source software is freely available for download, modification, and redistribution (<http://www.opensource.org/>). The concept, as explained by the Open Source Initiative (OSI), is based on the premise that when programmers can read, redistribute, and modify application source code, the software has the potential to improve very quickly [1]. MapServer is an example of open source software that conforms to the OSI definition. Apache and Mozilla Firefox are other examples of open source software commonly associated with web services.

GNU/General Public License - The GNU General Public License (GPL) copyright was initiated by the Free Software Foundation, Inc. (<http://www.gnu.org/>) in attempt to guarantee one's freedom to share and make modifications to free software. The General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. The "free" in free software refers to freedom, not price. General Public Licenses are designed to ensure that anyone has the freedom to distribute copies of free software (and charge for this service if so desired), that the source code is available, that the software can be modified [2].

Mandrakelinux, Red Hat, and Debian are popular Linux GNU/General Public License distributions which are available for download at no cost.

3 Open Standards

The National Map business model includes a well defined process for adding OGC-compliant Web Map Service implementations to *The National Map*. These WMS implementations are owned and managed by a distributed network of local, state, and federal partners. The process of adding data to *The National Map* via distributed networks would not be possible without standard interface specifications. Open Geospatial Consortium (OGC) standards (<http://www.opengeospatial.org>) define these spatial interface specifications which enable transfer of geospatial data over the web in a standard and predictable manner. OGC-compliant WMSs are made available across any network, application, or platform and are compliant in the sense that they are not tied to the syntax or conventions of a specific server implementation [3].

MapServer is an OGC-compliant WMS. Regardless of the platform from which MapServer is operating, its behavior, relative to specific web requests, is predictable. Conversely, ArcIMS "out-of-the-box" communicates very well with other ESRI applications but it is not OGC-compliant. This non-compliance issue presents obvious interoperability problems when attempting to transfer geospatial data over the web to non-ESRI implementations. In an attempt to remedy interoperability problem, ESRI supplies OGC-connector software that enables ArcIMS to conform to OGC specifications. Through the connector, ArcIMS will accept requests defined by the OGC WMS specification (`getCapabilities`, `getMap`, and `getFeatureInfo`). Incoming requests are converted to an internal ArcXML format and responses are converted back to a format understood by an OGC WMS client. USGS data partners hosting data from ArcIMS implementations must enable the OGC-connector software before their data can be included in *The National Map*.

4 Materials and Methods

The following is a summary of the hardware and software configurations used in testing the hypotheses of this investigation.

4.1 Hardware and Software Configurations

As an initial proof-of-concept test, Mapserver was compiled on a low-end PC which was retrieved from USGS surplus stock. Costs associated with this implementation of MapServer, in terms of hardware and software, were minimal. IT network support was also minimal since access to this initial implementation of MapServer was limited to the local USGS local area network. Public access was restricted.

Favorable results returned from the proof-of-concept testing moved the investigation into its second phase. This phase of testing involved compiling MapServer on a USGS production server. The intent was to grant public access to this instance of MapServer, thereby providing an opportunity to observe overall system performance in a "real world" production environment.

Proof-of-Concept WMS Configuration:

- Gateway E-3400 "surplus" PC
- Processor - Intel Pentium 3 processor (single)
- Clock Speed - 800 MHz
- Memory - 256 mb RAM
- Operating System - Mandrakelinux 10.0
- Server Application - Apache Web Server
- Web Map Server Application - MapServer 4.2.3

Production WMS Configuration

- Processor (single) - AMD Athlon K-7
- Clock Speed - 2.08 GHz
- Memory - 1GB RAM
- Data Storage - 3ware Raid, 1 boot hard drive, 8 hard drives connected via SCSI-2 interface.
- Operating System - Mandrakelinux 10.0
- Server Application - Apache Web Server
- Web Map Server Application - MapServer 4.4.0

Network Characteristics associated with Production WMS Configuration

- Physical Network connection: 100Base-TX
- Firewall config: Connected to a Public Services Network on a Juniper Netscreen 50
- Router: connected to a 100Base-TX interface on a Cisco 7200 series router
- Wide Area Network: T3 connection into the GeoNet3 WAN
- Internet: The GeoNet3 WAN has at least two different connections to the public Internet over T3 lines

4.2 Method

MapServer can be compiled to run from several operating systems including Windows, Linux, and MacOS. Linux was chosen as the operating system to support this investigation because it is open source software. Additionally, Linux is widely recognized as having strong networking performance, due in large part to excellent multitasking functionality [4]. This multitasking characteristic of Linux has direct application to a production WMS scenario.

A low-end PC was acquired from in-house surplus stock and loaded with the Mandrakelinux 10.0 operating system. To reduce the amount of processing overhead, the Mandrakelinux configuration was customized on the low-end PC to specifically support web server tasking. Word processing, spreadsheets, and other common office applications were not included in the installation. Apache web server, included with the Mandrakelinux distribution, was configured as the httpd server software.

Upon configuration of the Linux operating system, the latest version of MapServer was downloaded from <http://mapserver.gis.umn.edu>. To support MapServer operational requirements, the PROJ4 open source projection library, along with associated open source graphics libraries (GDAL, GD, LIBCURL2), were downloaded and included in the installation. Refer to Appendix A for URLs to these libraries and a summary of the MapServer installation procedures used in this investigation. No problems were encountered compiling MapServer or the associated libraries on the Linux operating system. Successful compilation produced a single mapserv.exe binary which executes from the httpd cgi-bin directory.

The primary objective of this investigation was to assess MapServer's performance as a Web Map Service (as opposed to, say, a Web Feature Service or Web Coverage Service). Initial testing to assess MapServer's WMS functionality began with loading and serving transportation data, in shapefile format, on the surplus Gateway PC. Transportation datasets with file sizes ranging from 50-100mb were served from MapServer in .png format, reprojected, and symbolized (both locally and via Styled Layer Descriptors (SLD)). Testing also included an assessment of MapServer's raster display and reprojection functions using USGS-produced orthophotos in geotiff format. In addition to serving geospatial datasets, the OGC-compliance characteristics of MapServer were evaluated. Responses returned from getCapabilities, getMap, getFeatureInfo requests were analyzed for compliance.

Encouraged by the positive results of the initial proof-of-concept testing, a decision was made to install MapServer on an existing USGS production server with the hope of bringing an instance of MapServer on-line as a public facing WMS.

5 Results and Discussion

MapServer's performance was evaluated on a low-end surplus PC and a USGS production server using a variety of vector, point, polygon, and raster datasets. Results obtained from testing MapServer's performance and related characteristics are summarized below:

5.1 Functionality

General - MapServer functionality was evaluated using datasets in shapefile and GeoTIFF format. These are the most common source file formats found in *The National Map*. No problems were encountered serving vector, point, or polygon data from MapServer. Test datasets used included

national, state, and local coverages. As expected, MapServer image display time was faster on the production server configuration than the Gateway E-3400 surplus PC, however both systems displayed small shapefile datasets (10-75mb) within acceptable response times when compared to other distributed map services currently serving data to *The National Map*. As shapefile datasets approached 100mb, the display times associated with the instance of MapServer executing from the surplus PC increased to the point of being unacceptable for use in a production environment (as a general rule, display times of 5-seconds or more are generally considered unacceptable).

Similarly, no problems were encountered displaying raster data in GeoTIFF format from MapServer. Mosaiced USGS digital orthophoto quarter-quads (DOQQ) and high-resolution urban area orthoimages were used to evaluate MapServer's ability to serve relatively large raster datasets. As expected, image display response times increased as the size of the raster datasets increased, however, in all instances the raster images were successfully rendered from both platforms. No attempt was made to create reduced resolution image datasets as part of this investigation, however, in an actual production scenario this would be required. Additionally, no attempt was made to access raster datasets from any type of database implementation such as Postgres or Oracle. We plan to test MapServer's ability to interface with databases in the near future.

In addition to simple display functions, vector and raster datasets were also successfully reprojected within MapServer. Datasets referenced to local projections were reprojected to the earth centered projections commonly associated with national coverages (e.g., WGS84). Likewise, datum transformations from NAD27 to NAD83 were processed without any problems. Reprojecting datasets increased system processing requirements by about a 20% as compared to those datasets not requiring reprojection.

Worth noting is the fact that MapServer could indeed be successfully configured to reliably function from a low-cost, widely available, PC platform. This configuration would most likely be adequate to meet the demands of small-scale operations. However, due to performance issues associated with large datasets and the average users expectations regarding server response time, this configuration would not be adequate for serving large national coverages to *The National Map*.

Symbology Support – Consistent symbology is an important objective of *The National Map*. For this reason, MapServer's ability to apply symbology was carefully evaluated as part of this investigation. Symbology was applied to sample geospatial datasets in the following two ways:

- locally, via MapServer Configuration File (.map)
- remotely, via dynamic Styled Layer Discriptors (SLD)

Results of applying symbology locally, i.e., from the MapServer configuration file, were successful, however, of greater importance was MapServer's ability to apply symbology by means of dynamic SLDs residing on remote computers. The USGS views this approach to applying symbology as the most robust solution to achieving consistent symbology throughout *The National Map*. Dynamic SLDs were successfully applied to vector datasets served from MapServer. Procedures for effectively symbolizing point features via dynamic SLDs are still being developed.

OGC Compliance - MapServer's response to standard getCapabilities, getMap, and getFeatureInfo requests were evaluated to assess the system's conformance to OGC WMS standards. Response to the getCapabilities request produced a valid xml response document consistent with the OpenGIS Map Server Capabilities predefined Capabilities Document Type Definition (DTD). Since the getCapabilities response serves as the primary vehicle for reporting a WMS's configuration and data holdings, it is a requirement that any map server supplying data to *The National Map* be able to produce a valid getCapabilities response document. OGC-compliant getMap and getFeatureInfo requests sent to MapServer via IE Web Browser and *The National Map* viewer applications successfully yielded responses that resulted in the display of map images and feature attribute information in a predictable manner, respectively.

Performance - An important aspect of this investigation was to evaluate MapServer's overall performance compared to existing USGS ArcIMS implementations. Having both MapServer and ArcIMS executing from production servers allowed some basis for a generalized performance evaluation in a real-world production scenario. Although no quantitative benchmark tests were conducted, general consensus supported the observation that MapServer, functioning as a WMS, is capable of serving geospatial data as efficiently as existing USGS ArcIMS implementations.

When comparing MapServer to ArcIMS, it must be understood that ArcIMS includes functionality that MapServer does not include. This includes html and applet based viewers for the client as well as several components on the server. The components on the server include the application server, manager components, and the spatial servers. The spatial server is what render the images, handles spatial queries, extracts features, etc. MapServer is only analogous to the ArcIMS spatial server, though the other two features can be added to MapServer using other tools. [5].

Many users of both systems in the GIS community would say MapServer is faster than ArcIMS, however, formal testing associated with this investigation cannot support or refute that claim. To date, no attempts have been made to optimize MapServer performance. Additionally, the MapServer configuration currently residing on the USGS production server is not utilizing any geodatabase implementation. Additional MapServer performance tests are planned. Results from this investigation to date suggest that MapServer, executing from a production server running Linux, can support the needs of *The National Map* as effectively as commercial ArcIMS implementations.

5.2 Cost

Start-up costs associated with the MapServer implementation executing from the Gateway E-3400 PC were minimal in terms of hardware, software, and personnel expenditures. The PC hardware cost is estimated to be about \$300. The Linux operating system and MapServer open source software were acquired at no charge. Operating system and MapServer configuration required about 12 hours of Computer Tech time. The MapServer proof-of-concept implementation was restricted from public access by a router firewall so IT personnel costs to ensure network security were negligible.

As expected, the overall cost of the MapServer implementation residing on the USGS production server was considerably higher than the initial proof-of-concept configuration due to server hardware costs and IT personnel resources required to implement network security measures

consistent with agency policy. The estimated value of the production server hardware configuration (2002 technology) is estimated to be about \$1500. IT personnel costs were estimated at 160 hours. The cost of the existing USGS network infrastructure is not included in this analysis. Obviously any implementation of a public facing web service will require an adequate supporting network.

5.3 Documentation

The available MapServer published documentation (<http://mapserver.gis.umn.edu/doc.html>) proved to be adequate in providing basic information pertaining to system configuration and the application of map objects in the MapServer configuration file. Equally helpful were the canned tutorials, also available for download from the MapServer website, which provided numerous examples of the application's functionality. As our specific requirements for using MapServer in a production environment evolved in complexity and specificity, gaps in the published documentation became evident. Although the documentation fell short in describing solutions to many of our specific intended applications, the MapServer discussion board, listserv archives, and Wiki proved to adequately fill in the gaps. This form of web accessed technical support proved to be a key characteristic of open source development that enabled this investigation to continue to move forward.

6 Conclusion

This investigation into MapServer has focused on a narrow piece of its overall capability, namely, its ability to serve geospatial data over the web. Testing is planned over the next several months to investigate additional MapServer functions and interfaces. Of particular interest is MapServer's ability to apply point feature symbology via application of dynamic SLDs. Additionally, investigations into performance optimization through interfacing MapServer with datasets stored in geodatabases is of great interest to the USGS. Results to date indicate the MapServer WMS is a sound and low-cost alternative to commercial map servers, and can effectively function to serve geospatial data to *The National Map*.

7 References

- [1] "Open Source Initiative", <http://www.opensource.org>
- [2] "Preamble, GNU General Public License", <http://www.gnu.org/copyleft/gpl.html>
- [3] "Vision and Mission, Open Geospatial Consortium", <http://www.opengeospatial.org/>
- [4] Welsh, Dalheimer, Dawson, Kaufman, "Running Linux." O'Reilly, 2003.
- [5] "MapServer vs ArcIMS", <http://mapserver.gis.umn.edu/cgi-bin/wiki.pl>

Attachment A. Creating a MapServer Instance on Linux

Download Source

Download current version of MapServer source code from <http://mapserver.gis.umn.edu> or type:

```
wget http://cvs.gis.umn.edu/dist/mapserver-4.2.5.tar.gz
```

At this point, the directory that the MapServer source is downloaded to is not critical, just create a temporary working directory such as **home/temp/source_download/** to use to hold the source for the present.

Unpack the MapServer source code tar-ball

```
tar -zxvf mapserver-4.2.5.tar.gz
```

This will unpack the MapServer source into a directory named “mapserver” which is automatically created when the file is unzipped.

Before proceeding with the MapServer configuration process, download and configure the following libraries if applicable to intended use:

Suggested MapServer Libraries

GDAL, raster format translator library. Download to temporary working directory and unzip tar file:

- Download from <http://www.gdal.org/download.html> to /home/temp/gdal_source_tar/, which will automatically create a subdirectory called gdal-1.2.3 when unzipped.
- configure gdal:

```
./configure --with-png --with-geotiff --with-jpg --with-gif --with-gdal
```
- compile and install gdal: **make install**

PROJ4, projections library. Download to a temporary working directory and unzip tar file:

- wget <ftp://ftp.remotesensing.org/pub/proj/proj-4.4.8.tar.z>
copy to /home/temp/proj4_tar/, which will automatically create a subdirectory named **proj-4.4.8** and a subdirectory called **nad**. Before building the PROJ4 library you need to copy the nad27 datum transformation lookup tables to PROJ4 as follows:
- wget <ftp://ftp.remotesensing.org/pub/proj/proj-nad27-1.1.tar.gz>
Copy the proj-nad27-1.1.1.tar.gz file to the /proj-4.4.8/nad/ subdirectory.
Unzip tar file.
- After the nad27 tables have been copied into PROJ4, cd to proj-4.4.8 and run the following:

```
./configure  
make install
```

GD, mandatory MapServer graphics library. GD can be installed directly from Disk 3 of Mandrakelinux 10.0 distribution set or downloaded from <http://www.boutell.com/gd/> . Install both the development and static GD libraries.

LIBCURL2, file transfer library required for MapServer WMS/WFS support. Can be directly installed from Disk 4 of Mandrakelinux 10.0 distribution set.

Configuring and Compiling MapServer

Assuming the above referenced libraries have been successfully built, MapServer can now be configured and compiled.

- To configure MapServer, type:
./configure --with-freetype --with-png --with-jpg --with-gd --with-proj --with-ogr --with-gdal --without-tiff --with-postgis --with-wfs --with-wfsclient --with-httpd=/usr/sbin/httpd2

- To create the MapServer executable, type **make**
Upon completion, your directory should contain the mapserv executable binary.
-rwxr-xr-x 1 user user 351177 Oct 20 11:38 mapserv

- As a simple test to determine if MapServer will run:

```
[user@host mapserver]$ export LD_LIBRARY_PATH  
[user@host mapserver]$ ./mapserv
```

Response:

“This script can only be used to decode form results and should be initiated as a CGI process via a httpd server.”

The message above is perfectly normal, and means the mapserv program is operating properly.

- The mapserv binary executable is a CGI executable, meant to be called and run by the installed web server. Move the compiled “mapserv” executable from the current directory to the cgi-bin directory:
/var/www/cgi-bin/mapserv

To further test if the MapServer CGI program is working, open the following URL on a web browser:

<http://localhost/cgi-bin/mapserv?>

If everything is working properly, you should get the following message:

“No query information to decode. QUERRRY_STRING is set, but empty”