Design Features and Software Architectures of Web-based Choropleth Map Services

Haixia Zhao
Department of Computer Science
University of Maryland, College Park, 20742
Email: haixia@cs.umd.edu
September 6th, 2001

Abstract

Web-based choropleth map services have become the trend in publishing geo-spatially referenced statistical data for public access. A client-server architecture is considered to be a necessity to fulfill the growing demands for a sophisticated interaction, and the support of dynamically changing maps. This paper first summarizes the features of three current client-server architectures, and analyzes their pros and cons. Second, sending vector map data to the client is the common approach that current Internet projects use to support local map manipulation and implement dynamically changing maps. However, they usually require massive data transfer across the Internet, and relatively long development periods. Based on the regional feature of choropleth map, we propose to use color-region mapping to keep geographic knowledge in a raster base map image instead of vector file, so the client can locally change the map display without server-side support. This alleviates the burden on network bandwidth and client computing power, while being easy to adopt. Our approach is proposed to put Dynamaps on web, but we believe it could benefit many similar choropleth map service systems.

Keywords: choropleth map, web-based map service architecture, Internet, Java applet, dynamic query, information visualization.

1 Introduction

Federal government agencies generate massive volumes of statistical data. Choropleth maps are one of the most commonly used visualization methods in geospatial-referencing data exploration in GIS [PopViz], to reveal geographical patterns, to show phenomena with distinctive distributions, and to discover outliers (extreme values of variables with otherwise unremarkable distribution), etc.

Static choropleth maps are not enough for analysis purposes. It is more promising to support dynamically changing maps that allow users to interactively manipulate geo-object classification and immediately observe the resulting changes in the map to seek interesting spatial patterns. A significant amount of work has been done in this direction, from proposing methods, to developing integrated environments. Examples can be found in [DNS01] [Mon89] [VisEarth] [GisVis] [MBH98] [MBH99] [CMS96]. Whether they are designed as general analysis tools or to support special applications, they all try to put into an integrated environment multiple visualization approaches (such as scatterplot matrices) linked to each other, and make the visualization user manipulable, interactive, and dynamic.

However, these non-web-based systems have essential disadvantages when it comes to serving the public, especially occasional users. First, they are not accessible unless users obtain and install the analysis software as well as massive data, usually distributed on media such as CD-ROMs. This prevents occasional users from accessing the information, due to various reasons such as license issues and education level. Also, it is hard to obtain up-to-date data in a timely manner, even though the agencies might have spent great effort in maintaining and upgrading their distributions. With the popularity of the Internet, more and more government agencies and organizations are working on publishing their geo-spatially referenced statistical data on the Internet, by providing web-based choropleth map services for public access. However, the speciality of Internet technology and variation of Internet connection scenarios, such as network bandwidth and client computing power, prevent the functionalities available in non-web-based system to be put on the web directly or with little modification of the system architecture. Instead, the impact of map-oriented Internet applications on Internet clients, map servers and spatial database systems must be carefully considered. Solutions and techniques are needed to solve the problems suffered by deployment of interactive maps currently available on the Internet. [BT00].

This paper tries to classify the existing web-based choropleth map service solutions according to the architectures, describe the map service features provided by some typical websites, and analyze the relationship between the architectures and possible features enabled. It then summarizes some essential problems in the existing approaches, and describes our own approach to put Dynamaps [DNS01] on the web, based on the geo-spatial-referencing feature of choropleth maps. We believe our approach could be generalized to benefit similar choropleth map service systems.

2 Web-based map service

Research and professional work aiming at putting map services on the web are being carried on in various directions, from proposing standards to developing general tools and integrated environments.

[MHG01] introduces a web-deployable, cross-platform, Java-based software development effort called GeoVISTA studio. GeoVISTA studio is a component-based visual programming environment designed to support the fusing of diverse visual and analytical capabilities (in the form of a cross-platform, JavaBean components, an applet, or an application) into custom analysis tools that enable a multi-perspective approach to knowledge construction and dissemination.

[BMT97] presents an approach to an open infrastructure for geographic information on the Internet. The goal is to enable end-users to access data from several providers simultaneously, and integrate the data locally in a geographic browser, as easily as if he were working with a state-of-the-art GIS package with all required data on his own computer. According to [BMT97], the layer model is the standard model for geographic data. It can be the model of distributing data across different data source vendors, and can be obtained and merged dynamically in a geo-web browser. But standard protocols have to be defined. Though some protocols have been identified to address part of the problem (such as CEN, OGC, and specifications from [OpenGIS]), no comprehensive proposal exists so far that can be used to achieve open access to geographic data publishers on the Internet.

[FD97] introduces an OPEN GIS interface prototype. It uses an object-oriented method to model geographic objects, based on the OPENGIS Geodata Model. The interface includes two main modules: the manipulator and the extractor. The manipulator is responsible for the user interaction, and handles data visualization issues. Considering the Internet is frequently congested, the prototype avoids geographic functions in the interfacing code, and uses client machine caching enabled by Java, to alleviate performance problems.

[GO97] proposes to run GIS service in a service-oriented approach where users make their input data available to some GIS service centers that perform the necessary computations remotely and send the results back to the users. An MMM system is developed as a communication infrastructure to support this business model.

[SM98] presents a methodology to put interactive maps on the web without any special server software. It uses compressed self-defined vector files to represent display primitives derived from geographic database. It then sends the files to the browser, where a Java applet interprets and renders the maps. Java strategies, such as threads are used to increase performance. It tries to accommodate low bandwidth network connection by only transferring display geographic elements in a lossy compressed vector file format.

At the same time, many commercial solutions are already available. Examples are ESRI MapObject IMS and ArcIMS [ESRI], Autodesk MapGuide [Autodesk99], and Intergraph GeoMedia Web Map [Intergraph99].

The research and professional/commercial works described above are just a few examples of recent work. More examples can be found in [Soro98] and in section 3 of this paper. A list of pointers to examples of Java-based cartographic applets can be found at http://www.cgrer.uiowa.edu/servers/servers_references.html

The uniqueness of the Internet environment and the need to consider universal accessibility issues bring forward some problems that need to be addressed when putting a map service on the web. Consider the following scenario: a recently retired person dials up from an old PC at home, trying to figure out the cost of living and retired population distribution pattern of each state from the statistical data published on the Census Bureau web pages, to help him make a decision of where he should move. This scenario represents some of the issues that can be reasonably assumed to remain a problem for a large portion of occasional Internet users or Internet users from lower-developed countries.

- Occasional users usually do not know how to query and analyze the data, and find the answers to their questions. Interactive and dynamically manipulable maps are important to better represent the information delivered by massive statistical data.
- Low network bandwidth is a typical restriction. Occasional users also may not have powerful PC.
- Occasional users usually do not have the required plug-ins or special software (if any) installed, and may not want to invest the time or do not have the knowledge to install them. Due to this, special plug-ins or software should be avoided.

3 Typical architectures of current choropleth map services and map features enabled

Websites that current provide choropleth map services are usually based on one or more of the following three diagrams.

1. **File downloading**: Geographic data (usually organized as map layers) and statistical data can be downloaded from the website (instead of being ordered on CD-ROMs). They are usually in formats used by professional map makers. For example, National Atlas [NA] provides free downloadable map layers that are in Shapefile format. They can be interpreted by most GIS software packages, such as ESRI ArcGIS [ESRI] and Autodesk MapGuide [AutoDesk99]. Prepared map image files are also available for download in some popular image file formats, such as TIFF and GIF. However, the download size is usually very large, which is forbidding for occasional users with low network bandwidth.
2. **Prepared map images or animation movies embedded in web pages**: Prepared map images embedded in web pages as normal pictures are easy, fast to view, but they are usually static and users have no control of the contents and styles. Animation movie is a good way to illustrate the temporal changes of geographic distribution patterns of statistical themes, and also allows some interactions. But they usually require the installation of specific plug-ins, and a long time to download the data before any map can be displayed. For example, [NA] provides multimedia maps that require Macromedia Shockwave Player [Macromedia] to be installed with the browser.
3. **Interactive map service based on client-server architecture**: The previous two diagrams either require massive data to be downloaded, or special plug-ins to be installed, or do not allow much user interaction with the maps. A client-server architecture is needed to achieve the goal of interactive or even real-time dynamic choropleth map service. This architecture is the trend of web-based choropleth map service solutions for most projects and websites, though some websites may be still providing services based on the previous two diagrams at the same time. This paper mainly talks about this third approach in the following section.

Based on the observation of some recent web-based choropleth map service projects and websites, there are two primary ways of transferring map data to the browser: raster image files and vector files. Accordingly, there are primarily three different client-server architectures: (1) prepared raster maps with lookup, (2) on-the-fly generated raster maps by server-side GIS packages, and (3) vector map files interpreted by browsers. The following sections give general descriptions of the three architectures (in distinct solutions, some specific components in the architectures may miss, data structures and scripting technologies may differ, special strategies may be used to improve the performance). Table 1 at the end of this paper shows a comparison of map features and architectures of some choropleth map service websites.

3.1 Prepared raster maps with lookup

In this architecture, prepared static raster map images are stored on the server, usually as GIF files. Upon client requests, some image files are transferred to and displayed on the browser as normal pictures. These prepared image files are hierarchically organized, according to map ranges, scales, business logic, etc. The data structure is defined to keep the hierarchy navigation information on the server. That is used by the lookup mechanism to decide what raster images should be sent to the browser next upon request, according to the currently displayed map images and user actions. On the browser, buttons, select lists, clickable polygon areas in the raster images, can be defined for users to interactively request what to view next. The static raster map images can be manually generated, but more commonly, are generated from vector map data using some GIS packages due to the extremely large number of raster maps needed.

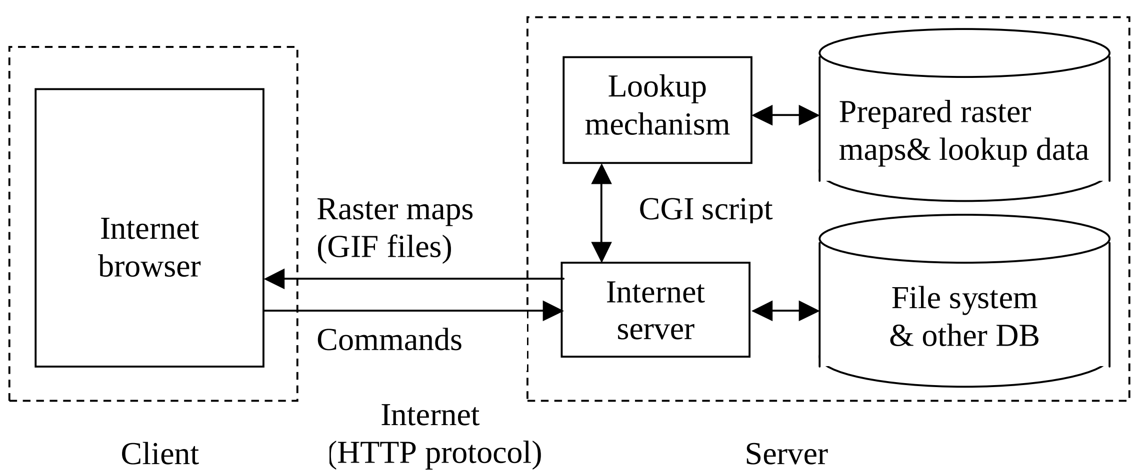


Figure 1 : prepared raster maps with lookup

As an example, [CB95] describes a generic web-based map interface using a collection of simple map images (GIF files) and a standard lookup table (including coordinates of boundary polygons of regions in map images) to provide visual interactive access to geographically-related information, developed by the Environmental Resources

Information Network (ERIN) Unit (http://www.erin.gov.au). The use of MAP and AREA HTML tags to define clickable polygon areas in map images helps overcome one imitation of basic web pages by allowing definition in a single image of multiple regions, disjunct regions, and regions of complex shapes. The coordinates of the polygons clicked by users on the browser, together with other parameters, identify the geographic region. A CGI program on the server consults the lookup table and uses the region identifier to get information about the regions clicked, including more detailed raster map files and statistical data related to the regions, etc. An Arc/info script is used to pre-generate the set of GIF raster map images. The system can also call a graphic package from CGI to generate some new maps on-the-fly, including thumbnail images indicating the current position of sub-map, prepared maps overlaid by existing transparent images (roads, rivers, etc) or objects (text, etc.). But mainly it is based on the pre-generated raster map image set. Interactive map functions are supported such as scaling, zooming (drilling down to more detailed regions or drilling up to overview), panning, identifying, and map overlaying.

As another example, [SWW98] describes a system for access and visualization of hazardous air pollutants. It uses a collection of linked micromap plots (simply called micromaps), which is actually a set of choropleth maps for certain regions and of certain scales. ESRI ArcView is used to generate the micromaps. Users can drill-down (from the US map down to Census tract-level maps) and up by clicking on polygon regions on current map images or selecting items from a list. This invokes the server-side CGI script to look up for the corresponding image files and send them back to the client.

This type of client-server architecture achieves the goal of interactive choropleth map service to an extent, but it suffers from several serious problems. For example, each user interaction results in a complete routine of sending a request, invoking the CGI script (could be other script) on server to find the map image files, and transferring the GIF files to the client for display refreshing. The server can be heavily loaded, and could be a bottleneck if many users were simultaneously accessing the map service and frequently requesting new maps. The frequent transfer of massive raster images is also a burden on network bandwidth, potentially causing substantial delays. This would be even worse for users with a slow network connection. Furthermore, the available maps are restricted to the set of prepared image files stored on the server. The number of maps to be prepared is huge, and increases exponentially if one wants to improve the quality of service (e.g., increase the number of zooming levels). It is also time consuming to add new maps.

3.2 On-the-fly raster maps generated by server-side GIS packages

In this architecture, no map images need to be prepared in advance. Instead, stand-alone GIS software is invoked on the server to generate the requested raster map images (usually GIF files) on-the-fly from vector map data and statistical data. This is typically the solution proposed by commercial GIS vendors. Examples can be found in [FactFinder] and [NA]. This approach does not have the available map restriction. As long as vector map files and statistical data are available in the database, the system can provide any sub-maps via the GIS software. Powerful commercial or free GIS packages for this purpose can be easily found on market.

However, many problems suffered by the previous architecture still remained unsolved. The user interface is quite restricted, interactivity is low, system response to user interaction is slow (sometimes unbearable for users with low network bandwidth). Dynamically manipulating the choropleth map and viewing the resulting changes in real time is not possible if the system relies on the server to generate every map images and send them back to the client as raster image files.

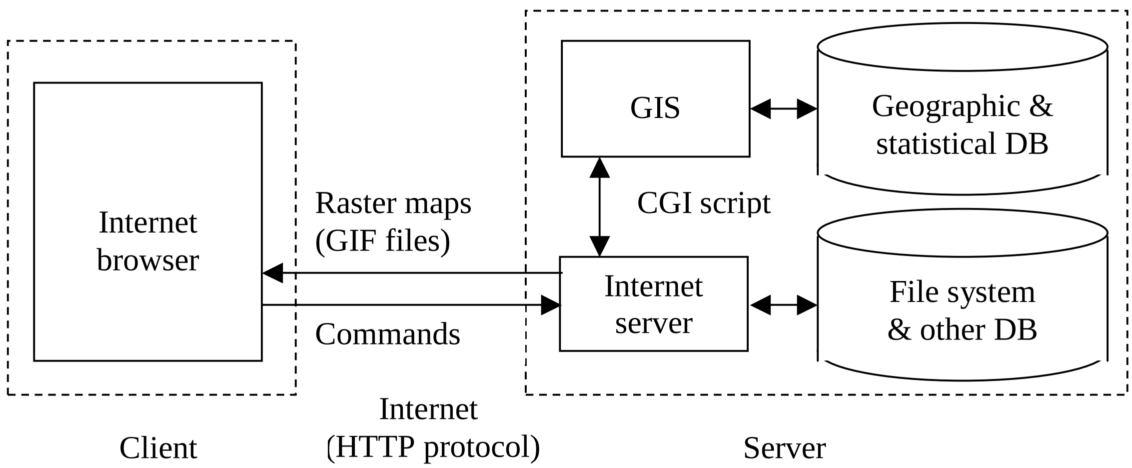


Figure 2: On-the-fly raster maps generated by server-side GIS packages

3.3 Vector map files interpreted by browsers

The previous two approaches cannot support efficient user interaction, or achieve real-time dynamic manipulation of the maps. The main reason is that they use raster image files as the format to transfer maps to the client, so the client has no knowledge of the geographic objects, and are unable to change the appearance of the map without the support from the server where a map display engine of the GIS package is used to interpret vector geographic data and generate maps. One approach to enable the browser to do local dynamic map functionality is to send vector files instead of raster image files to the browser. This way geographic information is not lost. The vector file format can either be self-defined, such as [DJ97] or use available standards. An application running on top of the browser is required to interpret the vector data and display the maps, like what a map display engine does in traditional stand-alone GIS packages. Based on the basic map display functions, other dynamic map functionalities can be further implemented. Such applications are typically plug-ins, Active-X controls, or Java applets [BT00].

The assumption of occasional Internet users requires designers to consider security aspects and to keep the efforts of the user low. The dependence of Active-X controls on the underlying operating system and security aspects restrict Active-X to Intranet applications (within a company or organization). In order to install plug-ins, a setup program must be run. Considering these restrictions, Java applets, which can be executed by almost every web browser, are most suitable [BT00]. The performance concern of Java applets is alleviated with the development of Java technology, and various approaches proposed. For example, the use of just-in-time compilers and the ability of Java to cache in the client machine help to increase the performance of Java. Some other methods that can further improve the performance and decrease system response time are described in [BT00], such as customizing class bundling in JAR files in order to avoid unnecessary Java code transfer (fetch-on-need) to reduce the subjective response time, and splitting the loading and visualization threads so users can change or restrict the map window to the area they are interested in before the data transfer is completed.

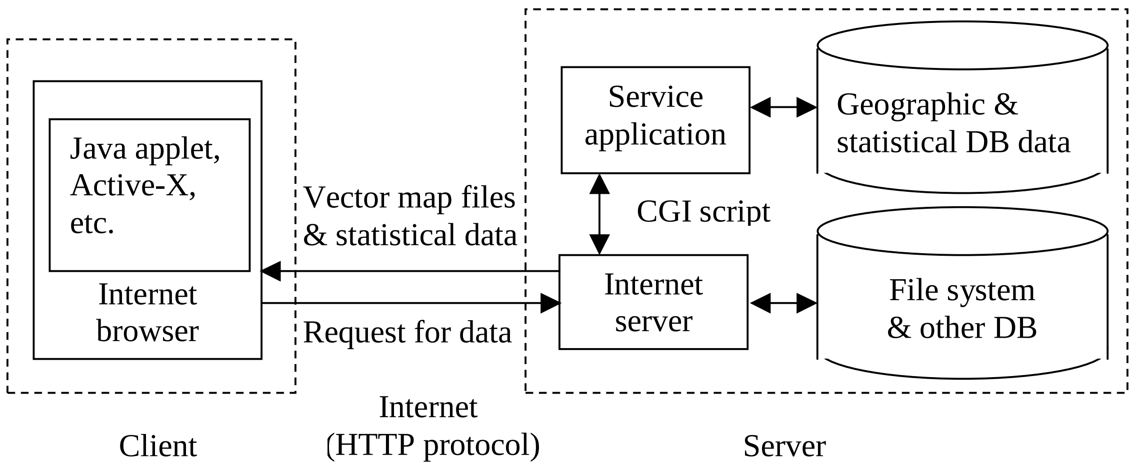


Figure 3 : vector map files interpreted by browsers

An example that does not use a Java applet to implement the browser-side map engine can be found in [DJ97]. It tries to introduce an approach to dynamic cartography based on the Tcl/Tk GUI builder (Tvl/Tk is a non-commercial graphical scripting language. The launch of a Tcl/Tk plug-in for browsers allows Tcl/Tk code, distributed over the Internet, to be interpreted by local browser.). Data structures are defined to model the geographical objects. The data is interpreted and drawn by the Tcl/Tk code in the browser.

However, the requirement of special plug-ins will harm the accessibility of the map service for occasional users who are not willing to do or do not know how to do it. Java applets are most suitable as explained before. A typical example that uses a Java applet to implement real-time dynamic choropleth map service is Descartes (formerly IRIS) [Descartes][AA99a][AA99b] [AA99c]. Descartes solves the unique problem of visual exploration of data sets by offering two primary instruments: automated visualization (presentation of data on maps), and dynamical manipulation of the maps and viewing the resulting changes in real time. The server (implemented in C++) provides database access, data manipulation (fulfillment of queries, calculations, transformation of structure of tables, etc.), and map design (selection of presentation methods and settings of their parameters using the internal knowledge base on thematic mapping). The client (Java applet) provides the user interface, draws the vector-based maps, and supports locally interactive manipulation of these maps. Descartes gives extensive support for local dynamic map functionality, such as linked brushing with a choropleth map and a scatterplot, and dynamic geographic regions classification in real time. A potential drawback is that it cannot be run across firewalls since the applet in Descartes communicates with the server through a socket connection. [AA99a]

Another example in [SM98] uses a self-defined vector file format to represent display primitives derived from geography database, and compresses the vector files with loss to reduce the size of data to be transferred on network. It also uses some programming tricks, such as Java threads to increase the system performance.

A huge effort is required to develop such a Java applet. To reduce the effort and time needed to develop and setup a service system, some ready-to-use Java clients and Java applets are available on the market from commercial vendors, such as ESRI ArcIMS, Autodesk MapGuide [Autodesk99], and GeoMedia Web Map [Intergraph99]. Some of the problems of using commercial applets are:

- The user interface of commercial Java clients is usually complex, and not suitable for occasional public users.
- Interactive features are typically supported by commercial clients, but local dynamic features, such as dynamic queries [DNS01], are still absent to the author's best knowledge when writing this paper. Customization from commercial clients is possible but it is quite difficult.
- Huge amounts of data, including applet code and data, need to be transferred before it can be used (at least hundreds of kilobytes or even several megabytes). As a result, the waiting time is very long. This is the case with [MetroGIS], which uses ESRI ArcIMS.

The approach of sending vector data to the client has the following main disadvantages:

- The Java applets are usually large, because they have to support complex map functionality, especially because they implement the basic functions of map display engine.
- It requires much longer development time, and more effort to develop such an applet from scratch. Commercial applets are available but the functionality and performance are not satisfactory in some cases
- A powerful client machine is required to execute the complex applet.

Our approach (described in the next section) can alleviate these problems for thematic choropleth map service, by redistributing the work between the client and the server.

4 Dynamaps on Web – Our Approach

The approach is originally proposed to put Dynamaps [DNS01] on the web for US Bureau of Census. Dynamaps is a generalized map-based information visualization tool designed by the Human Computer Interaction Laboratory at University of Maryland College Park [HCIL], for dynamic queries and brushing on choropleth maps. Users can use color-coding to show a variable on each geographic region, and then filter out areas that do not meet the desired criteria. In addition, a scatterplot view and a details-on-demand window support overviews and specific fact finding. The stand-alone tool has been implemented using Visual Basic, using ESRI MapObject as the map display engine. Figure 4 shows a snapshot of Dynamaps. More information about the Dynamaps Visual Basic version is available at http://www.cs.umd.edu/projects/hcil/census/.

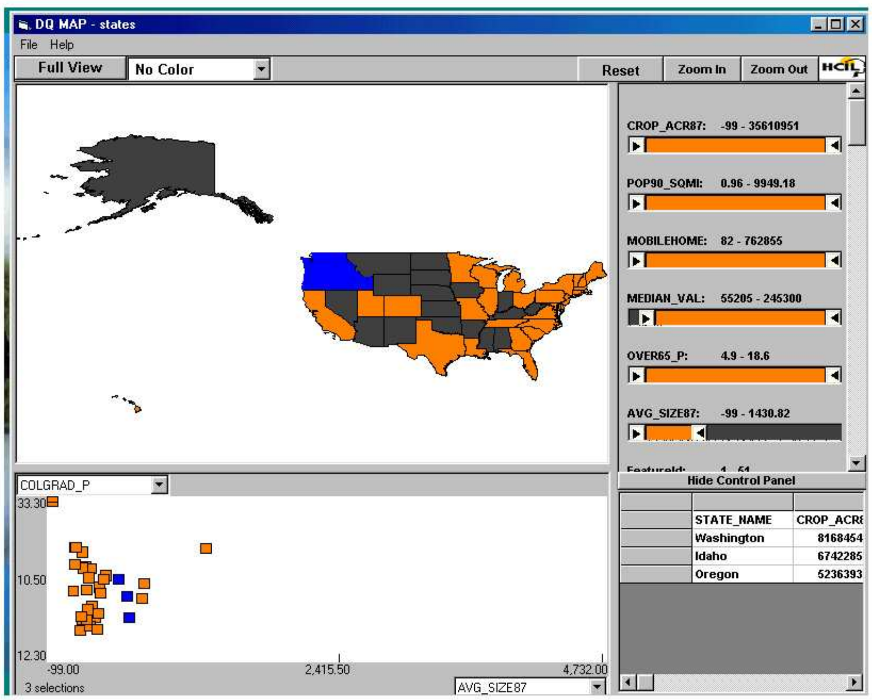


Figure 4: A snapshot of Dynamaps Visual Basic version

4.1 Goals of our approach

Web-based choropleth map services are better for public access as analyzed in previous sections. Technology exploration is being carried on to find a way of putting Dynamaps on the web, while fulfilling the following goals:

- **Functionality**: The web-based Dynamaps version should have equal functionality or even more compared to the Visual Basic version.
- **Performance**: Smoothly interactive filtering with the sliders should be achieved. This was also the main difficulty faced in the Visual Basic version [DNS01], where several algorithms are proposed to improve the refresh speed of the displayed map while sliders are scrolled. In the web-based version, this problem may be even more serious, because of the network bandwidth. To reduce the system response time, minimum network transfer and delay need to be achieved.

- **Accessibility**: The system should accommodate both occasional users and professional users, with high network connection or low network connection. It should be accessible to the public who use Java-enabled web browsers. No special plug-ins or software components are needed.
- **Scalability**:
    - *Data scalability*: The system should be easy to be maintained while data is updated or new statistical data themes are added to the system.
    - *Performance scalability*: The performance should be relatively stable even when number of geographic regions in a map and census attributes increase, or the number of users simultaneously accessing the system increases.

4.2 Main idea

The essential problem to be solved is how to enable the browser to display the map, recognize and manipulate statistical geographic regions on map without interacting with the server, implementing a map engine, or using a commercial map engine component.

For choropleth maps, based on its features (regions associated with statistical data), we can reconcile the Java applet complexity and functionality. Instead of sending vector map data to the client and implementing a map display engine on the client to interpret and display the vector data, we use a simple raster image to keep the knowledge of geographic objects. We call it the *base map*. The base map is sent to the browser, so the browser can change the map display based on the knowledge contained in the base map. Local dynamic manipulation is enabled without server support (so low network bandwidth is taken care of).

- Use a simple color-geographic-region mapping rule to keep the knowledge of geographic objects in the base map. The Java Applet uses this rule to recognize geographic regions in the base map, and to change the color attribute of regions. The applet does not implement a traditional map display engine, so it is much simpler and thus small in size.
- Use a commercial map display engine on the server side to generate the base map from vector map data that is in some industrial standard formats. The use of a commercial map display engine can shorten the development period because there is no need to implement a map display engine. Because commercial map display engines have been tested intensively, they are relatively stable, and usually compatible with variant geographic data formats.
- It can not only deal with regional geographic objects, but can also change the display of other geographic objects by overlapping the raster regional map with other raster geographic layers such as rivers and roads, and switching those layers on or off.

4.3 Architecture

The architecture of our approach is shown in Figure 5. In our exploration, we used ESRI MapObjectIMS as the map display engine to generate the base maps, but it can be replaced by other map display engine products.

The main procedures and technologies used when a user accesses the map service through a web browser are given as following:

1. After the user inputs the URL of the map service website, A JSP (Java Server Page) page is generated on the server and sent to the browser for the user to view the available statistical data sets. The JSP page is dynamically generated from the metadata (the data that describes what statistical data sets are in the system and how they are organized). The use of metadata ensures that the user can always view the up-to-date information about what data sets are available. Also, it achieves good data scalability of the system. When new data sets are added to the system, or the organization of data sets need to be updated, the system administrator only needs to update the content in the metadata structure, and the change will automatically take effect when the user reloads the JSP pages. No new code needs to be written.

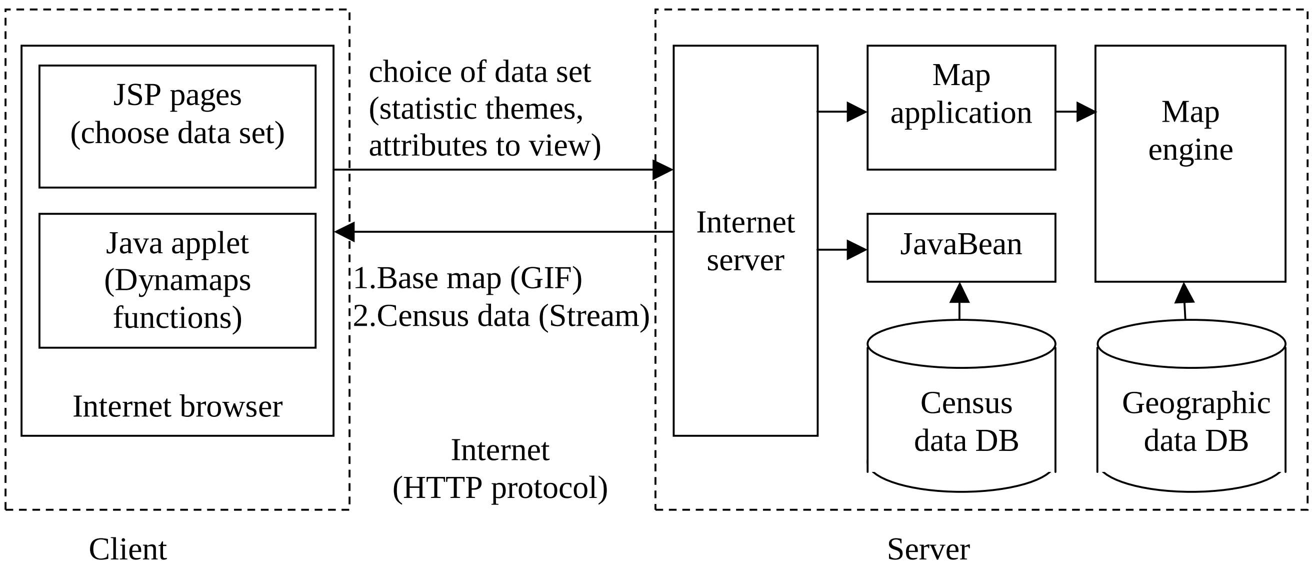


Figure 5: The architecture of our approach

2. The user can view the description of available data sets on JSP pages, and decide what data set and what level of map he is interested in, and what attributes in the data set he would like to query on and view the distribution pattern on choropleth maps. The user simply clicks and selects the data sets and attributes. The choices are sent back to the server.

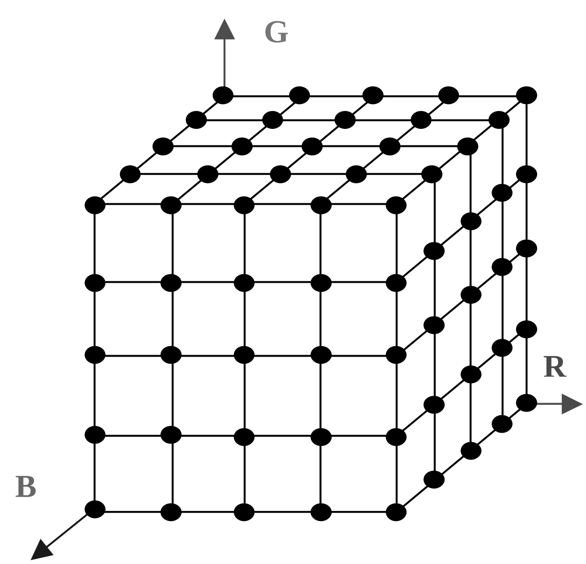
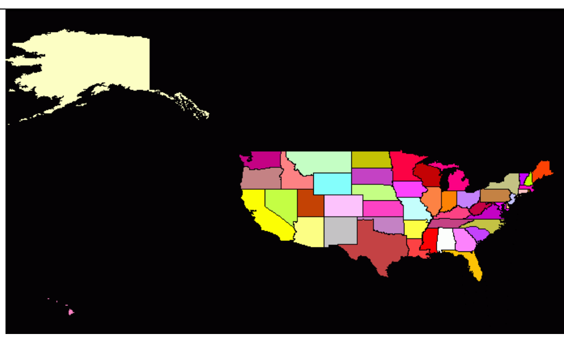


Figure 6: Color values for color-coding

3. A base map is generated by the MOIMS application according to the user's choice of map level (e.g., US map by states, Maryland map by counties, etc.), range (e.g., east region of US) and scale (e.g., 1:20000). A data file is generated by the JavaBean, containing the statistical data related to the regions for the selected attributes, ordered by the region identifiers. The base map is a choropleth map image color-coded to our predefined color-region mapping rule. The GIF format is used because GIF is very compact in encoding images with large single color regions, while the image quality is high enough to keep region boundary clear. The color-region mapping rule is illustrated in Figure 6. Each of the RGB components is divided into several sections (5 sections in our prototype, can be more). The 5 * 5 * 5 = 125 grid points comprises an array of color values that can be assigned to the regions in the base map. (The reason why not all 256 * 256 * 256 color values are used is explained in section 4.6) An algorithm is written to compute the unique color value from an integer (beginning from 0), and restore the integer from the color value. The regions in the base map are colored in the way that the integer corresponding to its color is the record index of this region in the data file. A specific color (black in our prototype) is preserved for the background of base map. Figure 7 shows a base map for US map by states, and a corresponding data file.



Attribute_1	Attribute_2	Attribute_3	State_ID	……
4319154	4040389	3894025	01	……
609311	550043	401851	02	……
4554966	3665339	2716546	03	……
2522819	2350624	2286357	04	……
32268301	29785857	23667765	05	……
3892644	3294473	2889735	06	……
3269858	3287116	3107564	07	……
……	……	……	……	……

Figure 7: Sample base map and data file

4. The base map is sent back to the client as a GIF image and the data file is transferred as a data stream.

5. Most of Dynamaps' functionality is implemented on the client by the Java applet without interacting with the server. The Java applet scans the pixels in the base map, uses the color-integer mapping algorithm to get the region index from the color, uses it as index to the attribute records, and changes the display if necessary, e.g., filter out some regions while the user scrolls the sliders.

4.4 Map features enabled

The idea of using color-region mapping to keep geographic knowledge in a raster image file enables all the Dynamaps features that are available in the Visual Basic version. The way in which some of the features can be implemented is described below.

- **Dynamic query by scrolling sliders**. Java applet scans the pixels in the base map, uses the color-integer mapping algorithm to get the region index from the color, uses it as index to the attribute values, decides if the region should be filtered out, and changes the display accordingly.
- **Scatterplot and linked brushing with the choropleth map**. With the data file on the client, scatterplot is easily implemented using a Java graphic package. The mapping between map color and record index in data file makes the linked brushing possible.
- **Overlap with other map layers**. Sometimes map layers other than the regions, such as rivers, may need to be displayed with the region map. Those map layers can be generated on the server and sent back as GIF images with transparent background, and overlapped with the resulting choropleth map image of the dynamic query.
- **Zoom to change the map scale**. This can be done by either interpolating the current map or having the server generate a new base map of the required scale. The first choice may cause an alias problem by interpolation, and does not generate as good quality map image as the second choice, and should only be used when the scale does not change much. If the user continuously zooms, new base maps should be obtained from the server to ensure the image quality.
- **Zoom to more detailed level maps**. The user clicks on a region (e.g., Maryland) to view the more detailed level map of that region (e.g., counties in Maryland). Region identifier can be obtained by the color-region mapping rule, so a base map for that region can be generated on the server and sent back, used as the basis of other operations such as dynamic query and linked brushing..
- **Identify a region**. The data can be retrieved from the data file and displayed for that region when the user clicks a region. Color-region mapping is used.

The functionality enabled is not limited to those in the Visual Basic version or described above. With Java graphics package, other functionality such as barcharts, graphs can also be implemented.

4.5 Characteristics

The original goals of our exploration are achieved. The system has the following main characteristics.

- High performance: using color-region mapping rule to keep the geographic knowledge, dynamic map features can be implemented locally on the client without interacting with the server. This ensures the real-time effect of dynamic map features. The load on network bandwidth is low, because the Java applet is small, GIF base map is small, and only necessary attribute values are transferred as stream. No network transfer is needed unless new base map or data set is requested.
- Good accessibility: it is accessible to users using Java-enabled web browser to access Internet. High network bandwidth and low network bandwidth are both taken care of.

No extra plug-ins or software component is needed on the browser. Only one copy of commercial map engine (e.g., MapObjectIMS) is needed on the server, which is good to the license issue.

- Good performance scalability, architecture scalability and data scalability: The performance is relatively stable even when number of geographic regions in a map and attributes increase, or number of users simultaneously accessing the system increases. With the use of metadata, the system is easy to be maintained while the data sources are reorganized or new data sources are added.

4.6 Prototype Implementation

In the Java applet prototype, we implemented choropleth map with dynamic query by sliders, zooming to enlarge, zooming to more detailed level maps, identifying, and changing the attribute to color-code the choropleth map. The applet classes are compiled with JDK 1.2.2. No Java swing package is used. A demo of the choropleth map functionality is available on www.cs.umd.edu/~haixia/map/map.html.

4.7 Potential problems and solutions

The main potential limitation of our approach is that the number of different geographic regions supported in one base map is limited. The limitation comes from the image format chosen to represent the base map. In our prototype, we choose GIF as the image format because GIF is very compact in compressing images with large single color regions. GIF can typically support 256 different colors. However, the GIF format uses a lossy encoding, so we only use part of the colors in the base map, considering the encoding error. We tried 5 * 5 * 5 = 125 colors in our prototype, which works very well (124 geographic regions are supported since one color is reserved for background). In the case where there are more regions, there are two solutions that could overcome the limitation.

- Use other image formats: there are a lot of other image formats that could be chosen, such as JPEG, BMP. JPEG supports 256*256*256 colors, but the image quality is not as good as GIF for maps with large single color regions (the boundary is blurred). 24-bit BMP can support 256*256*256 colors, and the image quality is good. The drawback is that BMP files are larger. However, it is possible to find an image format with balanced color numbers, quality, and size.
- Another solution is to use hierarchical color-coding of the base map. Figure 8 shows an example where the image format can only support 2 colors, but there are 4 regions. California (CA) consists of pixels with black color in base map 1 and black color in base map 2. Texas (TX) consists of pixels with black color in base map 1 but white color in base map 2.

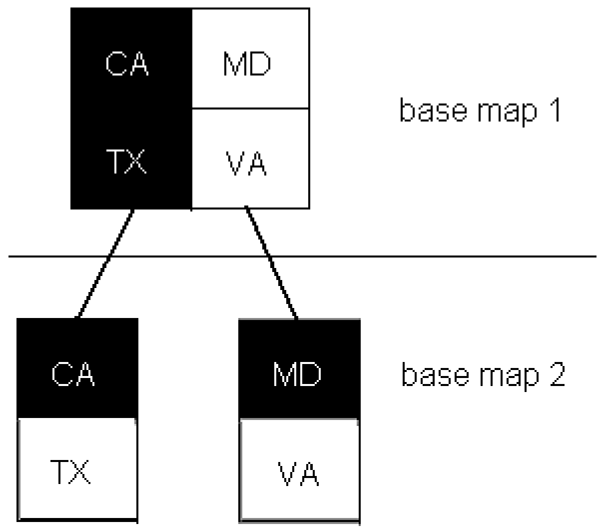


Figure 8: Hierarchical base map color-coding

5 Conclusion

Web-based choropleth map services have become the trend of publishing geo-spatial referencing statistical data for public access. Researchers and developers have done a lot of work to maximize its functionality and improve the performance. A client-server architecture is considered to be a necessity for web-based choropleth map services, but the three current client-server architectures suffer from some problems, and cannot satisfy every user case. Our exploration of using color-region mapping to keep geographic knowledge does not aim at competing with any of the existing work, but rather is an approach that enables putting Dynamaps on the web. We do believe that these techniques could be of benefit to similar choropleth map service systems.

Websites	[FactFinder]	[TransStat]	[Virginia Atlas]	[NA]	[MapStats]
How statistic data themes are presented by colors?	- Different statistic data themes are shown separately by static colorpleth map layers. - No overlapping of may layers.	- Different statistic data themes are shown separately by static colorpleth map layers. - No overlapping of may layers.	- Different statistic data themes are shown separately by static colorpleth map layers.	- Different statistic data themes are shown separately by static colorpleth map layers. - Some map layers can be overlapped by checking them on.	No thematic choropleth maps, Only one clickable 'US by states' map is given as geographic region index to the statistic tables
Map Levels	- country by states - one state by counties subdivisions - one state by census tracts - one state by congressional districts	- country by states, - one state by counties.	- state (virginia) by counties	- country by states, - one state by counties.	- country by states (as geographic index for query)
Relation Between Map Levels	Click a state on the 'country by states' map will NOT bring up lower level maps. To change map levels, choose from Drop Down Dialog Box.	click on a state brings up 'state by counties' map for that state.	No	Zooming into a state brings up the ' one state by counties ' level map of that state.	No
Architecture	Servlet calls ESRI ArcIMS Application Server to generate maps in-fly on brower's request and send maps back as GIF images. Server keeps track of geographic range and scale of GIF image so it knows	Use html AREA tag to define polygon clickable areas in a map image. CGI script sends back the requested GIF map	Pre-prepared static GIF map images are requested by choosing themes from drop down list boxes. No	ASP pages invoke ESRI middleware esrimap.dll to generate maps in-fly. Maps are sent back as GIF images. Server keeps track of geographic range and scale	Use html AREA tag to define polygon clickable areas in the 'US by states' map image. HTML pages containing

	what to generate next according to the mouse click position.	images according to the area clicked.	clickable area in maps.	of GIF image so it knows what to generate next according to the mouse click position	statistical data tables are fetched according to the area clicked.
Zoom	- Click on the zooming scale to zooming-in-out, or click on the map to recenter and zooming-in-out. - Zooming-in and zooming-out only change the scale of the currently shown map layer.	No	No	- Specify the zooming scale and click on the map to recenter and zooming-in-out. - Zooming in and zooming out only changes the map scale.	No
Identify	Click on the 'one state by counties' map to view the statistical data of the current shown map layers related to the clicked region.	Click on the 'country by states' map to view the statistical data related to the clicked state.	Statistic data of all counties listed in a table, no click-to-identify	Click on the map to view the statistical data of the currently selected map layers related to the clicked state and county	No
Dynamic features (such as dynamic query by sliders, or similar functions)	No	No	No	No	No

Table 1: Colorpleth Map Features of Some Statistic Data Publishing Web Sites

6 Acknowledgement

I would like to thank Dr. Ben Shneiderman, Professor, Founding Director, Human Computer Interaction Lab., Computer Science Department, University of Maryland, College Park. This paper is finished under his direction. I would like to thank Gunjan Dang, Graduate Student, Department of Computer Engineering, University of Maryland, College Park, for introducing me the Dynamaps Visual Basic version. I would also like to thank Dr. Evan Golub, Computer Science Department, University of Maryland, College Park, for commenting this paper. Finally, I should thank Yu Deng, Graduate Student, Department of Computer Science, University of Maryland, College Park, for the happy cooperation in implementing web version Dynamaps prototype. She is also the person who implements the metadata part.

7 References

[AA99a] G.L.Andrienko, N.V.Andrienko, etc, Thematic Internet Maps for Cities: Data Analysis and Publishing with Descartes. *Proceedings of 21st UDMS'99 conference* (Venice, April 21-23, 1999), Urban Data Management Society, Delft, pp.XII.1.1-XII.1.11

[AA99b] G.L.Andrienko, N.V.Andrienko, Interactive Maps for Visual Data Exploration, *International Journal of Geographical Information Science*, vol. 13 (4), June 1999, pp.355-374.

[AA99c] G. L. Andrienko, N.V. Andrienko, Data Mining with C4.5 and Interactive Cartographic Visualization. *User Interfaces to Data Intensive Systems*, 1999, IEEE Computer Society Los Alamitos, CA, pp. 162-165. ISBN 0-7695-0262-8.

[About] Summary of GIS software packages currently on market, either commercial or freeware, http://gis.about.com/library/blsoft.htm.

[Autodesk99] Autodesk MapGuide, http://www3.autodesk.com/.

[BMT97] Carel van den Berg, Co Meijer, Frank Tuijnman, etc., Multi-server Internet GIS: Standardization and Practical Experiences. *International Conference and Workshop on Interoperating Geographic Information Systems*, December 3-4, 1997. Santa Barbara, California, USA

[BT00] Thomas Brinkhoff. The Impacts of Map-Oriented Internet Applications on Internet Clients, Map Servers and Spatial Database Systems. *Proceedings 9th International Symposium on Spatial Data Handling* 10-12 August 2000, Beijing, China.

[CB95] D. Crossley, T. Boston, A generic map interface to query geographic information using the world wide web. *The Fourth International World Wide Web Conference* (Boston, Massachusetts, USA, 1995). Also available at http://www.w3.org/Conferences/WWW4/Papers/australia/.

[CMS96] D. Cook, J.J. Majure, J. Symanzik, etc., Dynamic Graphics in a GIS: Exploring and Analyzing Multivariate Spatial Data Using Linked Software. *Computational Statistics: Special Issue on Computeraided Analysis of Spatial Data*, 1996, Vol. 11, No. 4, 467-480.

[Descartes] Demo of interactive maps proposed as an addition to the paper [AA99b], http://borneo.gmd.de/descartes/IcaVisApplet/.

[DNS01]G. Dang, C. North, B. Shneiderman, Dynamic Queries and Brushing on Choropleth Maps. http://www.cs.umd.edu/projects/hcil/census/, To appear *Proc. International Conference on Information Visualization 2001*. Available from IEEE Press (July 2001). HCIL-2001-08 , CS-TR-4254 , UMIACS-TR-2001-37.

[DJ97] J.A. Dykes, Exploring Spatial Data Representation with Dynamic Graphics. *Computers & Geosciences*, 23 (4), 345-370.

[ESRI] Environmental Systems Research Institute, http://www.esri.com.

[FactFinder] American Fact Finder, Bureau of Census, http://factfinder.census.gov

[GeoVISTA] GeoVISTA Studio in [MHG01], http://www.geovista.psu.edu/

[FD97] F.T. Fonseca, C.A. Davis Jr., Using the Internet to Access Geographic Information: An OPEN GIS interface prototype. *Interopí97 First International Conference on Interoperating Geographic InformationSystems* , Santa Barbara, CA. Also available from http://www.spatial.maine.edu/~fred.

[GisVis] Geographical Information System Visualization project, US Environmental Projection Agency, http://www.epa.gov/gisvis/

[GO97] Oliver Gunther, From GISystems to GIServices: Spatial Computing on the Internet Marketplace. *Proceedings of the Int. Conference on Interopating Geographic Information Systems*, Santa Barbara, CA, December 3-4. 1997.

[HCIL] Human Computer Interaction Laboratory, University of Maryland College Park. http://www.cs.umd.edu/projects/hcil/.

[Intergraph99] Intergraph Corporation, GeoMedia Web Map, http://www.intergraph.com/gis/support/.

[Macromedia] Macromedia, http://www.macromedia.com.

[MapStats] MapStats – United States, FedStats official statistical, http://www.mapstats.gov/.

[Soro98] Alexandre Sorokine, Introduction to MapTerminal, 1998, http://members.tripod.com/~asorokin/intro.html.

[MBH98] A.M. MacEachren, F.P. Boscoe, D. Haug, and L. W. Pickle. Geographic visualization: Designing Manipulable Maps for Exploring Temporally Varying Georeferenced Statistics, *Proceedings, Information Visualization '98*, IEEE Computer Society, Reliegh-Durham, NC, Oct. 19-20, 1998, pp.87-94.

[MBH99] A.M.MacEachren, M. Wachowicz, D. Haug, R. Edsall, and R. Masters. Constructing Knowledge from Multivariate Spatiotemporal Data: Integrating Geographic Visualization with Knowledge Discovery in Database Methods. *International Journal of Geographic Information Science*, 13(4): 311-334.

[MetroGIS] MetroGIS DataFinder, http://www.datafinder.org/.

[MHG01]A. M. MacEachren, F. Hardisty, M. Gahegan, etc., Supporting visual integration and analysis of geospatially-referenced data through web-deployable, cross-platform tools. *National Conference for Digital Government Research*, May 21st - May 23rd, 2001, Los Angeles, California, http://www.dgrc.org/dg.o2001/papers/session-1/maceachren.pdf.

[Mon89] M. Monmonier, Geographic brushing: Enhancing exploratory analysis of the scatterplot matrix. *Geographical Analysis*, 21(1), pp.81-84, 1989.

[NA] National Atlas, U.S. Geological Survey, http://www.nationalatlas.gov/.

[PopViz] Cartographic Visualization of Population Data Tutorial, Manchester Information & Associated Services, http://www.mimas.ac.uk/argus/Tutorials/CartoViz/PopViz/.

[OpenGIS] Open GIS Consortium, The OpenGIS Abstract Specification, http://www.opengis.org/techno/specs.htm#abstract.

[SM98] A. Sorokine, I. Merzliakova, Interactive Map Applet for Illustrative Purpose. *Proceedings of 6th Internatinal Symposium on Advances in Geographic Information Systems*, ACM-GIS'98, Nov. 6-7, 1998, Washington DC, ACM Press, 1998, pp. 46-51.

[SWW98] J. Symanzik, D. Wong, J. Wang, etc., Web-Based Access and Visualization of Hazardous Air Pollutants. *Geographic Information Systems in Public Health: Proceedings of the Third National Conference* August 18-20, 1998, San Diego, California, Agency for Toxic Substances and Disease Registry. Also available at http://www.atsdr.cdc.gov/GIS/conference98/.

[TransStat] Geographic Information Services – Map Gallery, Bureau of Transportation Statistics, http://www.bts.gov/gis/maps/

[VirginiaAtlas] Statistical Atlas of Virginia, Geospatial & statistical data center, Univ. of Virginia Library, http://fisher.lib.virginia.edu/va_statlas/

[VisEarth] Visualizing Earth project, Penn State Geographic Visualization Science, Technology, and Applications Center, http://www.geovista.psu.edu/grants/VisEarth/

All websites appearing in this paper were last accessed by the author on August 1st, 2001.