Implementing a User-Oriented Web-based Traffic Data Management and Archive System

Timothy J. Welch  
Department of Computer Science  
Portland State University  
P.O. Box 751, Portland, OR, 97207  
E-mail: twelch@cs.pdx.edu  
Phone: 503-252-9954  
Fax: 503-243-1934

Kristin Tufte  
Department of Civil and Environmental Engineering  
Portland State University  
P.O. Box 751, Portland, OR, 97207  
E-mail: tufte@cecs.pdx.edu  
Phone: 503-725-3312  
Fax: 503-243-1934

Ransford S. McCourt  
DKS Associates  
1400 SW Fifth Avenue Suite 500  
Portland, OR 97201  
E-mail: rsm@dkspdx.com  
Phone: 503-243-3500  
Fax: 503-243-1934

Robert L. Bertini  
Department of Civil and Environmental Engineering  
Portland State University  
P.O. Box 751, Portland, OR, 97207  
E-mail: bertini@pdx.edu  
Phone: 503-725-4249  
Fax: 503-725-5950

Alan Snook  
DKS Associates  
1400 SW Fifth Avenue Suite 500  
Portland, OR 97201  
E-mail: aws@dkspdx.com  
Phone: 503-243-3500  
Fax: 503-243-1934

Submitted for presentation and publication to the  
86th Annual Meeting of the Transportation Research Board  
January 21–25, 2007

*Word Count: 4,355+ 7 Figures = 6105*

November 15, 2006
ABSTRACT

Transportation professionals are constantly collecting, processing, and managing more and more data on planning, operations, and design projects. Rapid, easy, and ubiquitous access to fundamental traffic count data is critical for ever more detailed transportation analysis as well as convenient retrieval. In Portland, Oregon, a “Google Local” web-based map interface and associated geo-database structure to access data has been developed whereby a user can access via the web historical traffic count data for intersections and links. This paper highlights the process used to develop the transportation data system, how Google Local was used to create an easy to use graphical interface and the framework to expand the transportation data information system beyond traffic counts to other frequently available data sets (such as collisions, traffic signal timing cards, freeway management system data, transit stop data, speed limit conditions, traffic control conditions, probe vehicle data, automatic traffic recording stations). The paper outlines the prototype development and the user features. Details regarding traffic counts (turn movement and machine counts) and how they are stored, retrieved, and placed on the Google map in an open architecture are explained. The combination of a map interface and geo-database structure allows users to query the system and access the needed data. Users are able to effectively use the interface immediately. There are still issues that need to be resolved such as how to best manage and distribute the potentially large amount of transportation data and records such that administrative overhead is minimized.
INTRODUCTION

With the advent of intelligent transportation systems (ITS), the transportation community has transitioned from a data poor environment, to a data rich environment where data management is an increasing challenge. Transportation professionals are constantly collecting, processing, and managing more and more data on planning, operations, and design projects. Rapid, easy, and ubiquitous access to fundamental traffic count data is critical for ever more detailed transportation analysis as well as convenient retrieval. Currently, most transportation organizations and firms have some amount of paper-based transportation data, or their transportation data is digitally stored, but it is restricted to a limited number of people. Either way it is difficult for an individual to know exactly what data is available without a system for organizing it. Some systems do exist for accessing that data, but often times the interface is difficult to use.

We present a system that provides rapid, simple, and ubiquitous access to location-based traffic and transportation data. The system provides a web-based interface allowing the viewing, insertion, and management of transportation data using only a web browser. This allows the data to be made available potentially to anyone with access to a web browser and an internet connection. The idea of using the Internet to provide access to transportation data and tools is not a new one (1, 2). Our purpose was to make transportation data available in a novel way: using web-based mapping. This allows the user to interact with a map and retrieve data spatially by simply zooming to the location they are interested in and querying the types of traffic data that are available. Constraints can be placed on the queries allowing the user to quickly access only the data they want. The system is designed such that the data itself can be stored anywhere as long as it is accessible by the user’s web browser.

A similar program, the Regional Screenline Traffic Count Program (3) has been developed for use within the Southern California region. The paper describes the design and setup of the program and outlines the future requirements of the system. The Regional Screenline Traffic Count Program has objectives similar to the system introduced in this paper including: development of a traffic count database system and collection of existing available traffic counts from cities. The system is different however in its use of use of proprietary desktop software including ESRI ArcView and Microsoft Access for storage, viewing and analysis of the data. The system outlined in this paper is a web application which makes use of open-source and/or free third-party software and provides only storage and limited viewing capabilities. The user is expected to retrieve the data and use it for analysis in whichever way they choose.

We envision this system being used by all types of transportation professionals as a central place to store and access fundamental traffic data. For example, analysts often need traffic data for generating reports and forecasts, planners need traffic count data for justifying design decisions. This type of data is fundamental to the modeling and decision-making process and would benefit from being available through a single system.

This paper presents the initial implementation of this system which supports the storage and retrieval of traffic count data for intersection and roadways. We describe the main features of the interface and how to use them including querying, viewing and insertion of traffic data. Next, the design of the system is outlined including a description of the main components. The database design is also presented and the flexibility in that design for adding additional traffic data types. Finally, issues that we encountered are discussed and conclusions and next steps are presented. This system was developed within the Portland State University ITS portal which provides the foundation for the system including user management, database access, storage and network resources. These components are not discussed within this paper.

USING THE SYSTEM

Querying Traffic Counts

Querying for traffic counts consists of specifying where to look and what to look for. The user specifies where to look using the Google map, which is the central part of the user interface (See Figure 1). The map controls in the top left corner of the map allow the user to pan and zoom to the location and level of detail that they want. In addition, a Yahoo! geocoder interface, located in the top of the browser window (See Figure 2), allows a user to type in a location and jump directly to it on the Google map. The user can give a specific address/intersection or simply a city or state. Once the desired area has been brought up on the map, the user can then specify the type of data that they are interested in. They can specify the time of day (AM, PM), range of time (last 2 years, last 6 months, etc.), data type (intersection or roadway) and finally a specific roadway type if the roadway data type was chosen (classification count, tube count, etc). Selecting “view markers” then queries the system’s traffic count records and refreshes the map with markers. Each marker represents a location that contains traffic counts that meets the user’s criteria.
Viewing Traffic Counts

Markers can be selected with the mouse to bring up additional information about the location including a descriptive name and the types of actions that can be taken (See the marker info “bubble” in Figure 2). Selecting “View Counts” loads the bottom frame of the application with a table where each row provides metadata and options for each count available at the location selected (See bottom of Figure 2). The metadata, or data about the data, consists of essential information about a given count including the count type, day of the week it was collected, year, start time and end time. This allows the user to glance through a potentially long list of counts and quickly see which data entries they are interested in. In addition, a “File” link is provided in each row allowing the user to download the count and a “More Info” link is provided in each row allowing the user to view additional metadata and options available for an individual count.

Clicking the “More Info” link in any row of the table shown in Figure 2 reloads the bottom frame of the application with a new table containing all of the metadata and options available for that count (See Figure 3). The additional metadata includes the “essential” information listed previous and also includes the file format that the count is in, the username of the person that inserted the count, a timestamp of when the count was received, the city, state, and country that the count is located within and finally any additional comments provided by the person who inserted the count. All of this information is collected at the time that the count is inserted into the system, some of it in an automated way. In addition this table provides links for downloading the count and, if the user is a moderator, options for editing and deleting the count from the system.

The benefits of the layered access approach described above, whereby the user accesses progressively more detailed and specific information, is that they can quickly move through a potentially large dataset and only mine the data as deep as they need to. Many times the user may only need to find out if a count is available, or who inserted it and not access the actual count itself.

FIGURE 1 Google Traffic user interface.
FIGURE 2 Viewing all traffic counts available at a single location.

FIGURE 3 Viewing detailed information of a single traffic count.
FIGURE 4 Adding a location to the system by clicking on the map

Insertion of new locations into the system is done by selecting the “Add” button in the top menu and clicking a spot on the map to place the marker, as shown in Figure 4. This brings up a form in the bottom window frame allowing the user to provide information about the location and type of data then submit it. The latitude and longitude of the spot clicked is stored with the location record in the database. Once submitted the new location marker can be seen by clicking “view markers” to refresh the map.

Inserting Traffic Counts

If the user is a moderator or administrator, then every marker info window will give them the option to add a count to that location (See Figure 5). Selecting “Add Count” loads a form in the bottom window frame and allows the user to enter all of the required parameters for a count record. First, if the count file is located on the user’s local machine, and they want to upload and store it on the central file server, then they browse and select the appropriate file. If the file is stored on a remote web server then the web address of the count is provided. Finally, the user enters the start time, end time and data type of the count and submits the data.
FIGURE 5  Adding a traffic count to the system

When a user inserts a traffic count, a check is first done for duplicate counts of the same type at the same location. If any duplicates exist then the user is queried for whether they want to continue in inserting the count. Automatic checks such as this are essential to maintaining the integrity of the data.

Administration and Management

System users are separated into three basic groups: user, moderator and administrator. The “user” type allows searching and viewing of the data through the web interface as described above. Moderators can also edit or remove locations and counts, and perform basic administrative tasks such as adding/removing count datatypes and adding/removing file format descriptions directly through the web interface. Moderators are very important to maintaining the integrity and consistency of the records and data. Without them, it would be difficult to maintain a system used by multiple groups. A fully-implemented system would allow moderators to perform all of their necessary tasks using only the web interface. Finally, administrators can access everything available through the web interface and also the back-end system including the database management system (DBMS) for more complex changes. The DBMS is the software which manages the database tables.

SYSTEM DESIGN

The system is what’s referred to as a mashup (4), a hybrid web application which combines services and data from multiple sources. Traffic data records are stored in a central database server, a mapping interface is provided by Google for displaying information, and the data itself can be stored anywhere that is accessible by the users’ web browser. Each component is independent of the others. This means that each component can be replaced with a similar solution with minimal effort.

As Figure 6 shows, the system centers around the user’s web browser. The browser makes file requests to the central web server. If the file is a PHP script then it is passed to the server’s PHP interpreter where location and count queries are made if necessary to the database server and HTML content is generated. An HTML file is then returned and displayed in the user’s web browser. The Google map is generated by the Javascript interpreter on the local machine and returned to the browser for display. Map tiles are requested as necessary from various Google mapservers and returned to the user’s web browser for display within the map. As the user pans around on the map the asynchronous loading of new map tiles can be seen. When a user clicks a link to download a count, a file request is made to the server that stores the count, whether it’s the central web server or a remote server. The traffic count file is then returned to the requesting web browser for viewing by the user.

The decisions on how to design the system were determined almost entirely by the goals of the project and the third-party services that were to be made available within the application. For example, one requirement of the project was that the application be accessible using a web browser, thus providing ubiquitous access, and that the
application work within the two major web browsers, Microsoft Internet Explorer and Mozilla Firefox. This limited our options for creating an interactive, dynamic application to using JavaScript, which is built into both browsers, or using a client-side browser plug-in such as Adobe Flash or Java. The third-party web-mapping interface chosen, Google Maps, is only available through a JavaScript library, limiting our choice of client-side languages to JavaScrip. PHP was chosen as the server-side programming language because the language interpreter is free and numerous features are built-into the language that allow web applications to be quickly developed. Similar languages such as Python and Perl could have been used instead. Overall, the system components and languages work well together. Passing data between the components is the single biggest issue, and is a challenge when working with a web application. We proceed to describe the individual components of the system and their functionality.

![Architectural design diagram](image)

**FIGURE 6** Architectural design diagram.

**Google Local Map Service**

Google Local was chosen for the mapping component because it is easy to use, provides the necessary features for the initial system implementation, is widely used by the public, and provides a public API, or application programming interface, allowing the service to be used by any web application as long as Google's API Terms of Use are met. The Google Maps API is well designed and documented making the creation of maps a relatively simple task. The map itself simply is a part of the interface to the system, it gets the user to the data and allows data to be inserted. At some point the user always moves “off the map” into another frame within the application to do further work.

**Yahoo! Geocoding Service**

The Yahoo! Geocoding web service is used to translate location information into latitude and longitude coordinates. This service is also provided through a public API which is easy to use and extensively documented. The user simply provides a street address, intersection, city or state and a request is sent to Yahoo!. If the location given is valid then its latitude and longitude is returned. The Google Local map is then refreshed with that location. This allows the user to quickly jump from one location to the next. The beauty of the geocoding web service is that the user can provide a specific address or be more general and only provide the name of a city or state and the map will center on the location with an appropriate zoom level based on how specific the user’s request was.
PostgreSQL DBMS

The open source relational DBMS (Database Management System) PostgreSQL is used to store location and traffic count records. These records are queried numerous times while using the application whether it's to generate markers and data tables or to insert and edit the records themselves.

PHP / Javascript

The PHP and Javascript scripting languages were used to “glue” all of the components together. PHP was used to generate database search queries, process query results, generate HTML, process submission forms, and handle the flow of data between all of the components of the application. PHP also provides a PostgreSQL API making the execution of database queries and retrieval of results extremely simple. Javascript was used to generate maps with the Google Maps API. A Google map is represented with a Javascript object. Each time the map is refreshed a new map object is created. Additional objects such as markers are generated dynamically using PHP and loaded onto the map.

DATABASE DESIGN

Locations

Basic geographic information is stored for each location including a latitude and longitude as provided by the Google map object when a location is inserted. The coordinate is used to place the location on the map.

Counts

Counts are tied to locations via the LocationID attribute. Any traffic data types that might be added in the future, like traffic signal timing cards, would be linked to locations in the same way. The rest of the attributes describe the count including what it is, when it was collected, where it's located, and who provided it.

Users

A user account and login system provides a simple way to tie data insertions to a user. The username of a person who inserts a count is made available to all users. The idea is that this ties data quality to a persons credibility.

FIGURE 7 Database design diagram.
ISSUES ENCOUNTERED

Data Collection

Many local organizations have traffic data and are willing to make it available to everyone within a system such as this, but the biggest roadblock is getting the data in the system. Many traffic counts are stored in “dumb” (or static) formats, such as a PDF, where the necessary count attributes required by the system cannot be retrieved except by a person. The person-hours necessary to prepare the data for insertion is enough that some organizations are apprehensive about doing it. Another issue is the current method for inserting data where it is done manually by a person through a web form interface. An improvement would be to provide an automated way to insert counts.

Data Quality

Different organizations use different data formats. In addition, the completeness and quality of that data varies. During this project we quickly realized that if the system were to be used, certain data structure requirements would need to be put forth and met by anyone providing information. This would ensure some level of quality and completeness in the information provided.

Manageability

Currently, traffic count records are stored on a central server while the actual traffic count data can be stored on any outside file server accessible to the Internet. This means that each entity providing traffic count data must ensure that the record in the central database is correct at all times, especially the location of the external data file. This creates an additional workload on the data provider who may be maintaining their own internal records on their traffic data. One possible solution is to have outside data providers maintain their own traffic count records and provide them to the central server.

Efficiency

One of the greatest issues is the volume of records that could potentially be stored. As the number of records in the database increase, bottlenecks in the application will occur resulting in performance loss. For example, the number of results for the average traffic count request will increase with the number of records and thus so will the number of markers placed on the map. Generation and placement of even 1,000 map markers can take minutes on slower client systems. One method used to decrease the number of markers is to only display traffic count locations which are in the viewable area of the current map. If the user pans the map to a different location they must issue a new traffic count query. Another method of decreasing the number of map markers is to cluster areas that have a high density of markers into a group and represent it with a distinct marker. Zooming in on these locations will effectively decrease the density of the markers on the map and locations will be given their own individual markers.

Service Limitations

Free services such as Google Maps and the Yahoo! geo-coding web service have limitations on how much their services can be used. For example, Yahoo! allows no more than 5,000 geo-coding requests to be made from a single system in a 24 hour period. If this application were to be used by a large number of people then the limitations of these free services would be surpassed and a switch would need to be made from free to pay services. Recognizing this fact made keeping these components distinctly separate from the core application a priority so that switching would be relatively easy.

CONCLUSION AND FUTURE WORK

Overall, the system is very effective for the storage and retrieval of traffic data. It is a step towards an ambitious goal of providing open ubiquitous access to a broad amount of traffic data in the easiest way possible. Throughout the design and development of the current system the main focus was on the user interface. The combination of a mapping component and the ability to view the data in progressively greater detail allows the user to quickly query the system and access the data they need in an efficient manner. Most users with a minimal amount of knowledge are able to use the web interface right away. This is mainly attributed to people's familiarity with navigating websites and basic interactive mapping functions such as panning and zooming.

The system in its current state is not a complete solution. The main goal was to take advantage of the fact that traffic data has a spatial component and use it in the insertion and retrieval process. This goal was accomplished with the clickable map interface. This solution works well for the average user who is mainly querying data, but not for the user that is inserting a large number of records. More work needs to be done to
streamline the insertion process by automating as many steps as possible. One solution is to provide an interface that accepts traffic data records in a flexible format such as XML. XML, which stands for Extensible Markup Language is simply a way to encode structured information (7). A "language" can be created for defining traffic data records and the user only needs to provide the records in that "language". This process of encoding a record, transmitting it, and decoding it is already used in the current system for transmitting map marker locations/text and making geocoding requests to Yahoo!. It is a widely used and extremely useful method for moving data between systems.

In addition research can be done on methods for sharing between servers hosting traffic data records. It's not feasible to store all records on one server, but if a server could query other known traffic data servers then the data would be truly ubiquitous. This would also allow the participating entities to contribute their resources while still maintaining control over their own data. These record queries made from one server to another could be accomplished in the same way as record insertion, by encoding requests and results using a markup language such as XML.

Future work should also include adding support for additional types of traffic data to the system. This could include collisions, traffic signal timing cards, freeway management system data, transit stop data, speed limit conditions, traffic control conditions, probe vehicle data and automatic traffic recording stations. Steps should be taken to require the data be provided in formats that maintain the underlying structure of the data. This would make the data much more valuable because it can then be analyzed further using additional software by the user. For example a user could download a spreadsheet, database, CAD drawing, or GIS layer.

ACKNOWLEDGEMENTS

The authors gratefully acknowledge DKS Associates for providing the initial traffic count data, the resources to insert it into the system and valuable feedback from users. The National Science Foundation has funded this research. The contents of this report reflect the views of the authors, who are responsible for the facts and the accuracy of the data presented herein. The National Science Foundation assumes no liability for the contents or use thereof. The contents do not necessarily reflect the official views or policies of the National Science Foundation. This report does not constitute a standard, specification, or regulation.

REFERENCES