Open Source Software, Web Services, and Internet-based Geographic Information System Development

Brian N. Hilton, Ph.D.
Research Fellow
Claremont Information and Technology Institute
School of Information Science
Claremont Graduate University
130 East Ninth Street, Claremont, CA 91711
brian.hilton@cgu.edu

Abstract

The increasing availability of mature open source software and the continuing evolution of distributed computing now manifest in web services introduce a new dynamic into information system development. This new dynamic is examined in relation to the development of an Internet-based Geographic Information System – a Proximate Commuting system. This system was designed, developed, implemented, and evaluated in cooperation with a private, non-profit corporation dedicated to helping commuters find and use commute alternatives. Specifically, spatial analysis is performed that identifies the commutes of employees of multi-location organizations that could be reduced by working at a closer location. The findings of this study suggest that Internet-based Geographic Information System development, enabled by open source software and web services, is both an effective and efficient method of information system development. Moreover, preliminary findings from a formative evaluation indicate that the Proximate Commuting system addresses both software quality issues as well as meets the needs of its users.

1. Introduction

The purpose of this study was to consider the impact that open source software and web services may have on information system development – specifically in relation to the development of an Internet-based Geographic Information System (GIS). As a result, this study had two objectives: 1) demonstrate the use of open source software and web services as a method to develop an Internet-based GIS, in this case a Proximate Commuting system, and 2) evaluate the degree to which this system addressed software quality issues and hence, the needs of its users.

Information system development will be affected by the increasing use of the Internet. For example, the ability to locate existing software by searching the Internet may help to expedite the early stages of system development. The Internet is already used as a method for acquiring software specifications and software engineering notes (Doernhoefer, 2001). It is also common for application assemblers to use the Internet to find, download, and pay for existing objects they can use (George, 2000). Likewise, the Internet may also facilitate the sharing of parts of internal systems with external systems and partners. These all have the potential to dramatically affect system development. Two recent phenomena that are directly related to these issues are open source software and web services.

1.1. Open Source Software

Free and Open Source Software (FOSS) is software that gives users the right to run, copy, distribute, study, change, and improve it as they see fit, without having to ask permission from or make additional payments to any external group or person (Bollinger, 2003). There are several groups dedicated to managing and promoting FOSS, primary among them being The Open Source Initiative (Open Source Initiative, 2003) and The GNU Project (The GNU Project, 2003). Perhaps two of the most well known examples of open source software are the operating system Linux and the Apache web server. The Linux server is the second most commonly used operating system – 28.5% versus Windows server 49.2% (Netcraft, 2001) while Apache is the principal Web server in use today – 74.67% versus Microsoft 17.92% (Security Space, 2005a).

The President's Information Technology Advisory Committee believes the open source development model represents a viable strategy for producing high quality software through a mixture of public, private, and academic partnerships (President's Information Technology Advisory Committee: Panel on Open Source Software for High End Computing, 2000). For instance, a recent study (Bollinger, 2003) highlighted the fact that FOSS plays a critical role for the U.S. Department of Defense (DoD) in four broad areas: Infrastructure Support, Software Development, Security, and Research.

This study demonstrated the importance of FOSS by proposing, hypothetically, that FOSS be banned from use in the DoD. For example, it was shown that if FOSS were banned from Infrastructure Support, it would negatively impact the ability of the DoD to support web and Internet-based applications. Software Development would also be negatively impacted, as traditional languages such as C and Perl (a direct outgrowth of the development of the Internet) would be banned from use. Security depends on FOSS as well; specific FOSS infrastructure components (e.g., OpenBSD) directly support network security at the DoD. Moreover, the ability of FOSS applications to be updated rapidly in response to cyber attacks greatly enhances the ability of the DoD to defend against these attacks. Lastly, Research would be negatively impacted by the loss of the ability of FOSS to support the sharing of research results in the form of executable software.

In another recent study (Wichmann, 2002), 1,452 European companies and public institutions were queried regarding their motivations for using open source software and the benefits they derive from its use. This purpose of this study, conducted for the European Commission as part of the Free/Libre Open Source Software Survey and Study, was to overcome the lack of information regarding FOSS by: conducting surveys to generate primary data on FOSS usage and development; to identify indicators to measure value creation and dissemination in the FOSS arena; to identify business models based on these indicators; and to identify the impact of, and recommend changes in, government policy and regulatory environments regarding FOSS (International Institute of Infonomics, 2002).

Survey questions were asked along four dimensions relating to area of software use: Server Operating Systems (e.g., Linux, FreeBSD, OpenBSD), Databases (e.g., MySQL, PostgreSQL, Interbase, SAP-DB), Desktop Computers (e.g., Linux, KDE, Gnome, Mozilla, StarOffice, OpenOffice), and Creating and Operating Websites (e.g., Apache, PHP, Perl, Python, Squid, Open Source content management systems). Each organization that participated in the survey indicated how important several selection criteria were for their most recent decision to use open source software rather than commercial software.

The results of the survey indicated that regardless of area software use – Server Operating Systems, Databases, Desktop Computers, and Creating and Operating Websites – the basic picture remained the same: higher stability and better access protection, low or zero license fees, installation and administration cost savings, and open and modifiable source code were the key benefits and motivating factors for acquiring and using open source software.

In contrast to the benefits of open source software enumerated above, a report (Hissam, Weinstock, Plakosh, & Asundi, 2001) by the Software Engineering Institute at Carnegie Mellon

University identified two specific risks that an organization acquiring open source software faces:

1 - The software will not exactly fit the needs of the organization, and
2 - Ultimately there will be no real support for the software

The preceding studies highlight the influence, good and bad, FOSS is having on information system development. As observed at the DoD, FOSS is having a significant impact at all levels of information system development. Furthermore, the Free/Libre Open Source Software Survey and Study found that higher stability and better access protection, low or zero license fees, installation and administration cost savings, and open and modifiable source code were all keys reasons why organizations supported open source software acquisition. On the other hand, as reported by the Software Engineering Institute, there is the possibility that open source software acquisition may not meet the needs of the organization while support for open source software may ultimately lie with the acquiring organization.

1.2. Web Services

The use of web services on the World Wide Web is expanding rapidly as the need for application-to-application communication and interoperability grows (World Wide Web Consortium, 2002). Web services consist of a set of messaging protocols, programming standards, and network registration and discovery facilities that expose business functions to authorized parties over the Internet from any Web-connected device (Oracle Corporation, 2001). Basically, web services allow specific business logic to be exposed and used between independent applications with a minimum knowledge of the web service and/or underlying application. Web services range from the simple (e.g., a stock quote service) to the complex (e.g., a vacation planning system). Web services are a progression of the standards and protocols that were used to create today's Internet. The advent of web services promises to let an organization connect its applications to any number of other organizations relatively inexpensively and easily (Hagel, 2002).

There are many definitions of the term "web service", however, the following definition (World Wide Web Consortium, 2002) most clearly defines the term:

> "A web service is a software system identified by a URI, whose public interfaces and bindings are defined and described using XML. Its definition can be discovered by other software systems. These systems may then interact with the Web Service in a manner prescribed by its definition, using XML based messages conveyed by Internet protocols."

In general, the Web Services Architecture illustrated in Figure 1 assumes that advanced web services will be built on the foundation provided by eXtensible Markup Language (XML), Simple Object Access Protocol (SOAP), Web Service Description Language (WSDL), and Universal Description, Discovery, and Integration (UDDI). It should be noted however, that a "web service" as defined above does not require the use of the SOAP as a packaging format or a processing model. Nor does it require the use of the WSDL as a service description language or UDDI to describe business context data structures. At present, there are, and will continue to be, web services that use raw HyperText Transfer Protocol (HTTP) as the data transfer protocol and some form of XML as the message content.

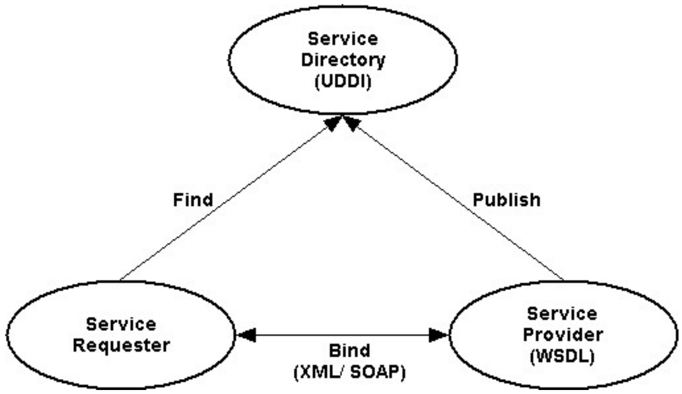

FIGURE 1 – Web Services Architecture

The fundamental Web Services Architecture illustrated above places into relationship various components and technologies that comprise a functional implementation. Figure 1 illustrates the relationships between service requesters, service providers, services, descriptions, and discovery where XML messages compliant with the SOAP specification are exchanged between a service requester and service provider.

Figure 1 further illustrates the basic request / response web service implementation. Here, a service provider publishes a WSDL file that contains a description of the service and endpoint information that allows a service requester to generate a SOAP message and send it to the correct destination. The service requestor and service provider then interact based on the service's description information published by the service provider and discovered by the service requester through some form of agent discovery. The service requester sends a message in the form of a request to the service provider for information or to perform an operation. The service provider receives the request, processes the message and sends a response to the service requester, which contains the result of the request or operation. An example of this basic request / response web service is the stock quote service. A service requester enters a stock symbol, e.g. IBM, into a user interface, submits the information to a service provider, and receives a response in return; an IBM stock price quote.

2. Internet-based Geographic Information System Design / Development

The Internet-based client / server architecture has great potential for improving accessibility to spatial data and to spatial data processing services (Abel, Taylor, Ackland, & Hungerford, 1998). Figure 2 is an illustration of the Internet-based Geographic Information System Architecture utilized in this study.

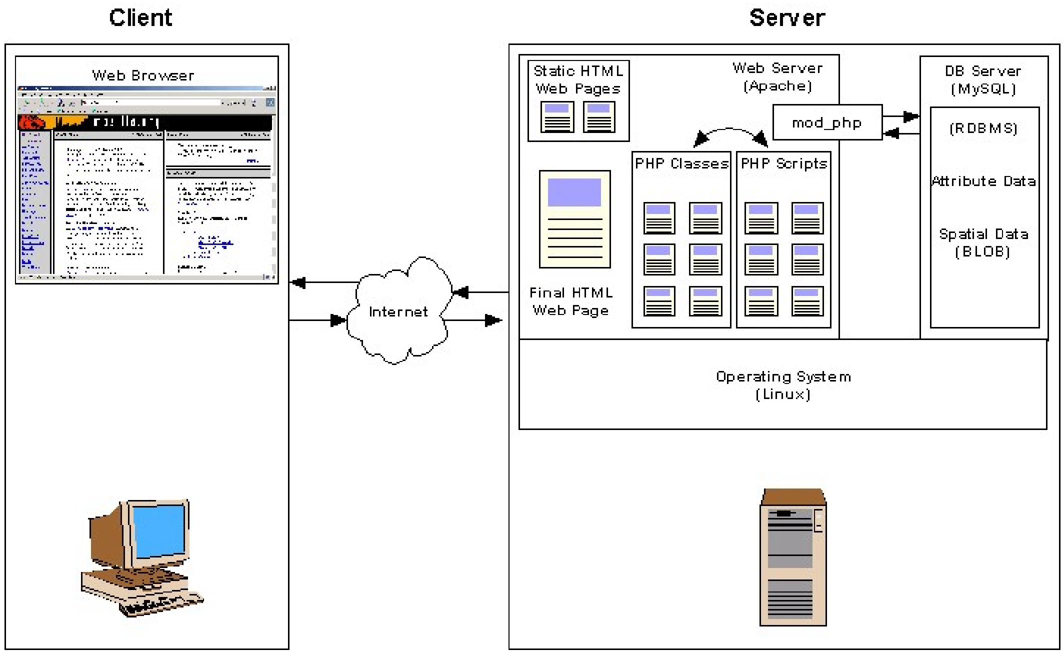

FIGURE 2 – Internet-based Geographic Information System Architecture

2.1. Thin-Client versus Fat-Client

The open standards of this architecture permit numerous approaches to the interaction between client and server. A way to categorize these approaches is in terms of functionality, i.e., are processing functions assigned to the client, the server, or a combination of both. The terminology used to describe these approaches are "thin-client / fat-server" and "fat-client / thin-server" where "thin" indicates that an insignificant amount of processing is required while "fat" indicates that a significant amount of processing is required. Put another way, "thin" implies that a small amount of software is required to perform the specific task while "fat" implies that a large amount of software is required to perform the specific task. These approaches are illustrated (Abel et al., 1998) in Figure 3.

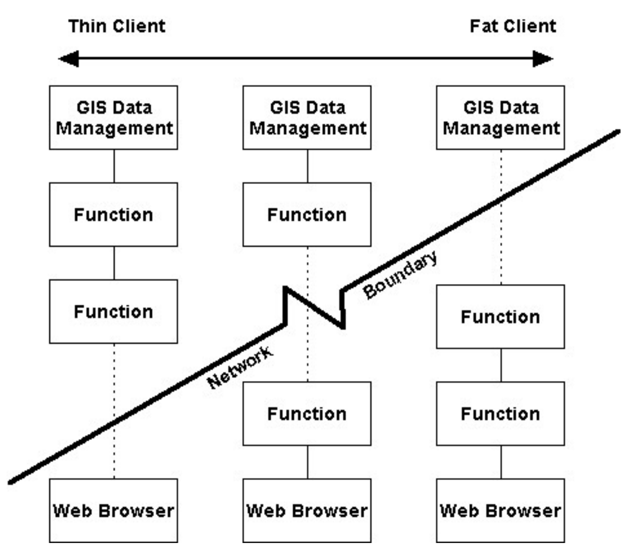

FIGURE 3 – Thin Client versus Fat Client Function Assignment

As seen in Figure 3, the uncomplicated approach to Internet-based client / server interaction is the "thin-client". This approach is not uncommon for the typical user where the client system consists of a basic Web-browser. The benefit of this approach for the GIS user is that spatial information is delivered quickly to the Web-browser in the form of images (e.g. PNG, GIF, JPEG). The shortcomings of this approach however, are the minimal functionality provided by the Web-browser and the requisite request(s) sent to the server for any additional information.

A more complex approach to Internet-based client / server interaction involves a movement towards a "fat-client" or thicker client, the two most common methods being "plug-ins" and "applets". A "plug-in" is a software component that extends and/or enhances the functionality of the basic client application – when data of the type associated with the plug-in is encountered, the plug-in launches, providing the required functionality. An "applet" is a Java application that is delivered to the client at the time of the request for information – the applet, along with the requested data, is sent to the browser and executed. One benefit of these approaches for the GIS user is the improved interaction with the spatial data, i.e. the ability to manipulate the data locally without the need to repeatedly interact with the server. Another benefit for the GIS user is the ability of these thicker clients to display vector data, i.e. data of a type not traditionally supported by Web-browsers. The shortcomings of these approaches are that plug-ins must be installed before they can be used while applets are only used once, i.e. any future requests for the same or similar data necessitate that the applet be re-sent.

2.2. Scalable Vector Graphics

The approach utilized in this study – Scalable Vector Graphics (SVG) – places it in the middle of the "thin-client" / "fat-client" approaches.[1] The World Wide Web Consortium has developed a specification that defines the features and syntax for SVG (World Wide Web Consortium, 2003). The following specific elements from this specification are the key reasons SVG was selected for this study:

- SVG is a language for describing two-dimensional graphics in XML,
- SVG allows for three types of graphic objects: vector graphic shapes, images, and text,
- SVG drawings can be interactive and dynamic,
- "onmouseover" and "onclick" can be assigned to any SVG graphical object, and
- Scripting can be done on XHTML and SVG elements simultaneously within the same Web page.

As related specifically to GIS, SVG also offers the following (Anderson, 2003):

- Geographic coordinates stored in a spatial database can easily be converted to SVG,
- Features may be grouped together and processed as distinct layers. For example, street segments with the same ID can be dynamically symbolized together to visually simulate dynamic segmentation, and

[1] At present, the major drawback to SVG is that Web-browsers do not currently provide native support for SVG, and hence, there is the need for a plug-in. However, future releases of all the major Web-browsers should have this functionality "built-in".

- SVG can be spatially referenced, allowing mouse events to "grab" real-world coordinates from the map and upload to a remote server for useful geoprocessing such as spatially querying features in a spatial database.

2.3. Open Source Software Internet Platform

LAMP is an acronym for the technologies that represent the open source Internet platform: LAMP = Linux + Apache + MySQL + (PHP | Perl | Python). Moreover, it represents the development environment of the Internet-based GIS in this study (see Figure 2). As a multi-purpose operating system, Linux is used for a wide variety of purposes including networking, software development, as well as an end-user platform. Apache is an implementation of an HTTP server and is the principal Web server in use today. MySQL is an implementation of a database server that is known for its speed and reliability. PHP is general-purpose scripting language that is particularly suited to Internet-based system development and is the most widely used Apache module (Security Space, 2005b; The PHP Group, 2005).

2.4. PHP Web Services

Of the many ways to implement web services, PHP Hypertext Processor (PHP), as a widely used general-purpose scripting language, is especially well suited for Internet-based system development as it was designed from the beginning to work on the Web and can be embedded into HTML (Ayala et al., 2002). In addition, PHP has several features that make it useful for web service development, including:

- Object-oriented programming capabilities,
- SOAP and XML-RPC classes, each able to support web service transactions,
- XML support, as well as several PHP extensions to expand XML functionality, and
- CURL extension, allowing communication via various protocols such as HTTP, HTTPS, FTP, Telnet, and LDAP

NuSOAP (Ayala, 2004; Ayala et al., 2002), a collection of PHP classes, allows users to send and receive SOAP messages over HTTP. It is an open source software application that forms the core of several web services toolkits for PHP including PEAR-SOAP. One of the major benefits of NuSOAP is that it is not a PHP extension; it is a PHP class. As a consequence, Web developers, regardless of the Web server platform available and/or permission restrictions, are able to use NuSOAP to create web services by including this class in the directory structure of their PHP-enabled Web server. In addition, the LAMP environment illustrated in Figure 2, while not the only Web server platform available (e.g., Windows), may be the only platform available to Web developers as it is the platform of choice for many web-hosting companies. As a result, PHP web services may be the only option for these developers. These were the key motivations for using PHP web services, and hence NuSOAP, in this study.

3. Internet-based Geographic Information System Implementation

In the previous section, the motivation for each design element utilized in the development of this study's Internet-based GIS was given. These design elements consist of an Internet-based client / server architecture, a SVG-based client, the LAMP-based server platform, and PHP-based web services. In this section, details of the Proximate Commuting system are given as well as the processes used to develop and implement this system.

3.1. Proximate Commuting

Commute Trip Reduction (CTR) programs give commuters resources and incentives to reduce their automobile trips (Victoria Transport Policy Institute, 2002). These programs, such as Rideshare Matching, Alternative Scheduling, and Tele-work, may be encouraged or required by local, regional, state, or federal policies.

One CTR program, Proximate Commuting, allows employees to shift to work sites that are closest to their homes. Here, spatial analysis is performed that identifies the commutes of employees of multi-location organizations that could be reduced by working at a closer location. Consequently, Proximate Commuting is a potential employer-based travel-demand management program under which large, decentralized employers reassign each voluntary participant to a job location (with the same employer) closer to the participant's residence to reduce commuting distances (Rodriguez, 2001). One type of information system that is particularly suited to perform the spatial analysis required of Proximate Commuting is the Geographic Information System – a group of procedures that provide data input, storage and retrieval, mapping and spatial analysis for both spatial and attribute data to support the decision-making activities of the organization (Grimshaw, 2000).

3.2. Open Source Software and Web Services Prototyping Methodology

There are a number of methodologies used to develop and support information systems two of the most well known being the Waterfall Model or Systems Development Life Cycle and the Spiral Model. The cyclic, or iterative nature of these approaches is especially characteristic of Rapid Application Development methods such as Prototyping (Hoffer, George, & Valacich, 1998). Prototyping is as a four-step interactive process between a User and Builder where an initial version is defined, constructed, and used quickly; as problems are discovered, revisions and enhancements are made to the working system in its user's environment (Naumann & Jenkins, 1982). In a similar manner, implementation of this system was achieved through the use of an Open Source Software and Web Services Prototyping Methodology illustrated in Figure 4.

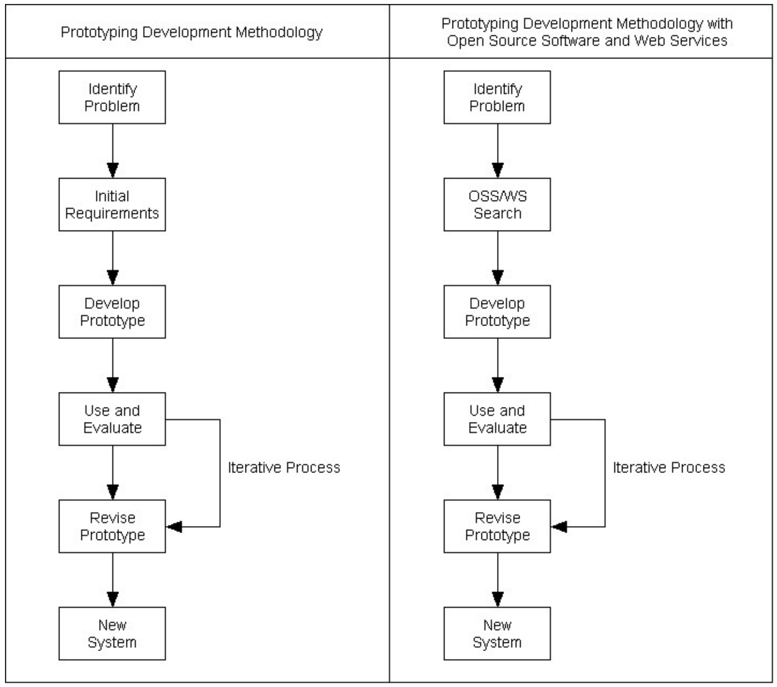

FIGURE 4 – Open Source Software and Web Services Prototyping Methodology

In both approaches, a problem is first identified. The difference between these two methodologies, however, lies in the next step, Initial Requirements versus Open Source Software and Web Services Search. Here, rather than engage in the development of a series of initial user requirements, a search is performed to determine whether or not an Open Source Software / Web Services solution(s) exists. If so, a prototype is quickly built, used, and evaluated. Iterative revisions are made as necessary to the prototype. Finally, a new system emerges.

For this study, the Open Source Software and Web Services Search consisted of a search of the Open Source Software websites www.freegis.org, www.opensourcegis.org, www.sourceforge.net, and www.freshmeat.com as well as the Web Services website of www.xmethods.com. This search yielded a number of potential Open Source Software applications such as Deegree, GeoServer, GeoTools2, and MapServer. However, given the design element outlined in the previous section, OpenSVGMapServer was selected.

In a similar manner, and based on the Proximate Commuting requirements outlined in this section, three web services – CalTrans (provided by Xmethods), AirportWeather (provided by Cape Science), and GetDirections (provided by MapQuest) – were selected. In addition, a new web service GeoLocator (provided on the localhost) was developed to provide the necessary spatial search features required of the Proximate Commuting system. NuSOAP was used to consume and create these various web services.

3.3. Proximate Commuting System

Illustrated below are three examples of this Internet-based GIS. In Figure 5, the user has clicked on a City / Zip Code feature (e.g. Berkeley, CA), thus sending a request to the GeoLocator web service where the user is presented with a new window displaying a listing of all "Commute-Friendly Jobs" within a specified radius of the city.

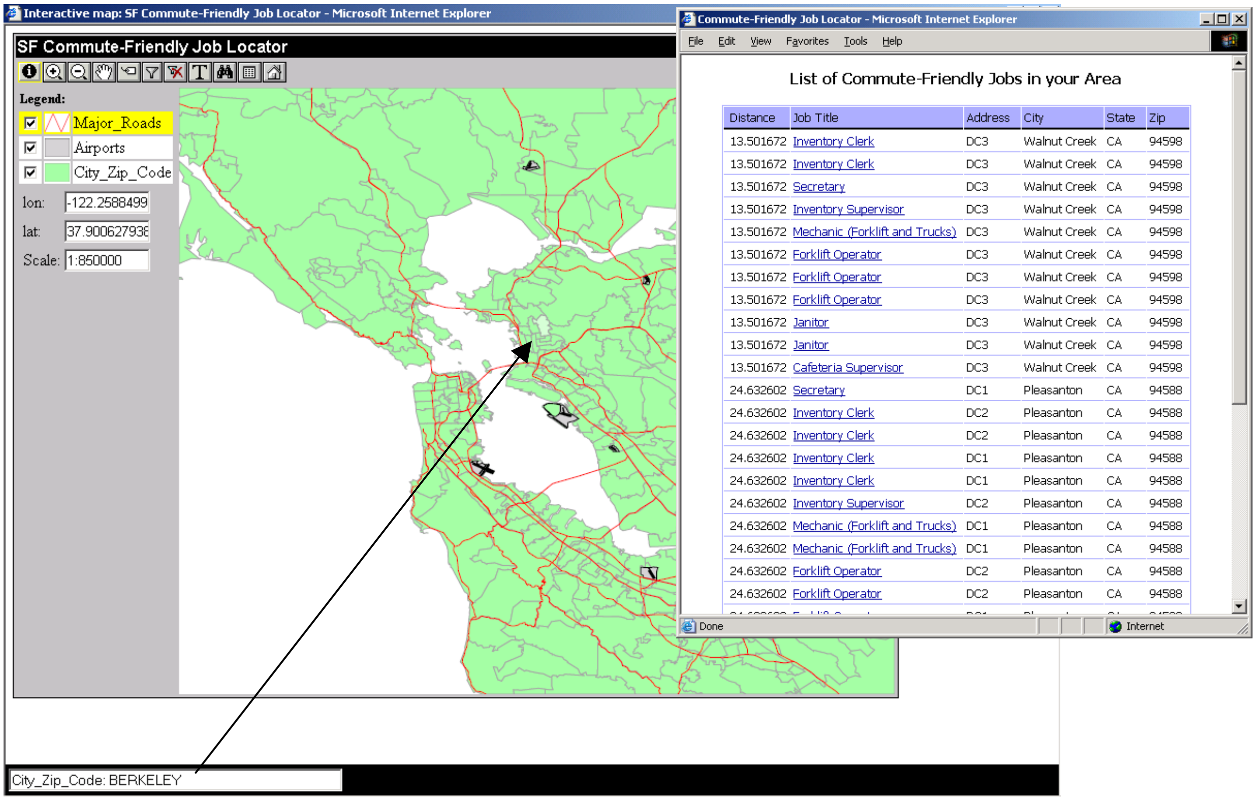

FIGURE 5 – GeoLocator Web Service

At this time, the user can either browse through this list for a new job location, perform a new spatial search by entering in a new search radius, or perform a new spatial search with additional attribute parameters entered, e.g. Job Type or City. In addition, the user can also retrieve driving directions by filling-in a form that sends a request to the GetDirections Web Service. Figure 6 displays this form while Figure 7 displays the response received.

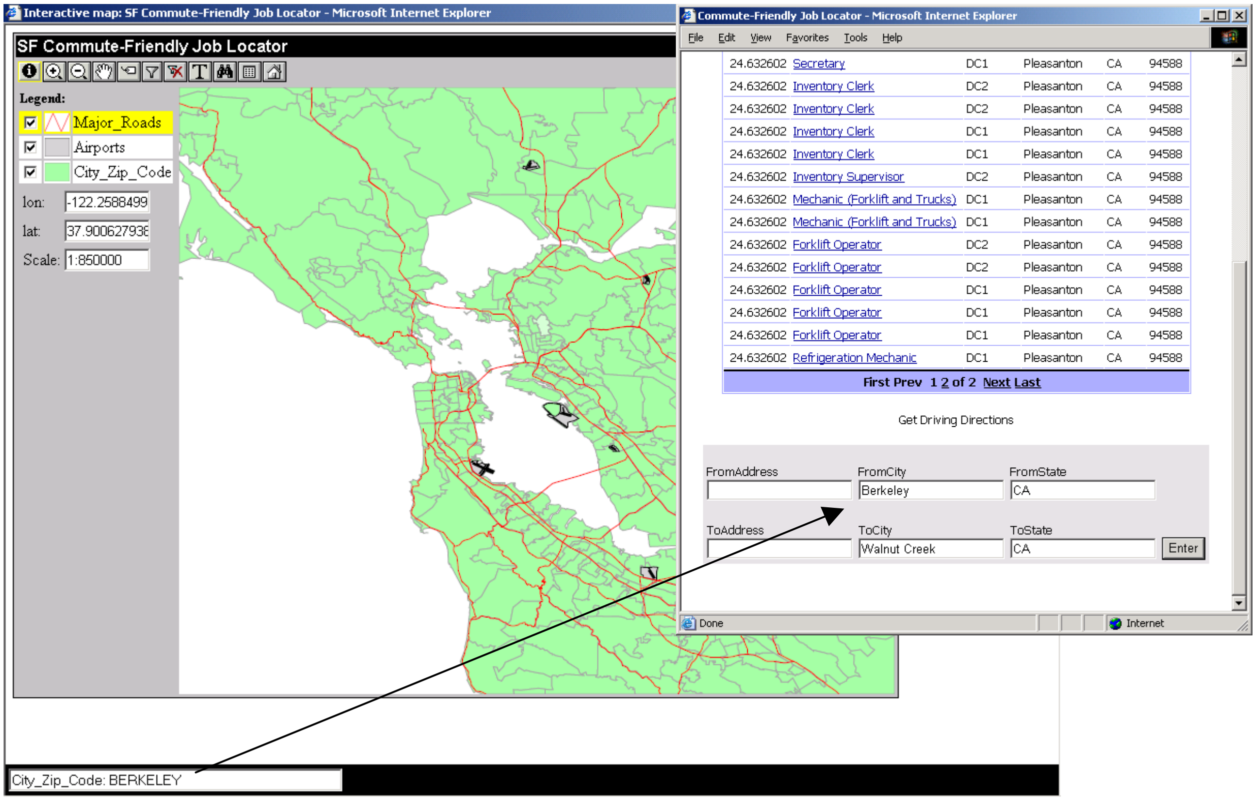

FIGURE 6 – GetDirections Web Service

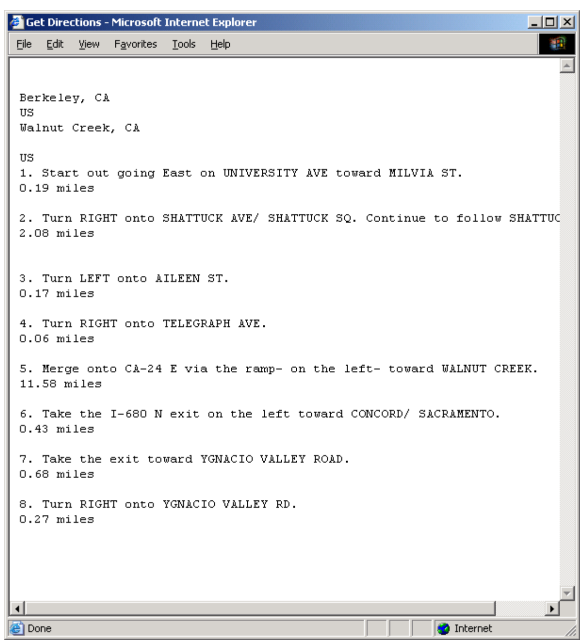

FIGURE 7 – GetDirections Web Service Response

In the Figure 8, the user has clicked on a Major Road feature (e.g. Interstate 80), thus sending a request to the CalTrans Web Service where CalTrans Highway Information is returned.

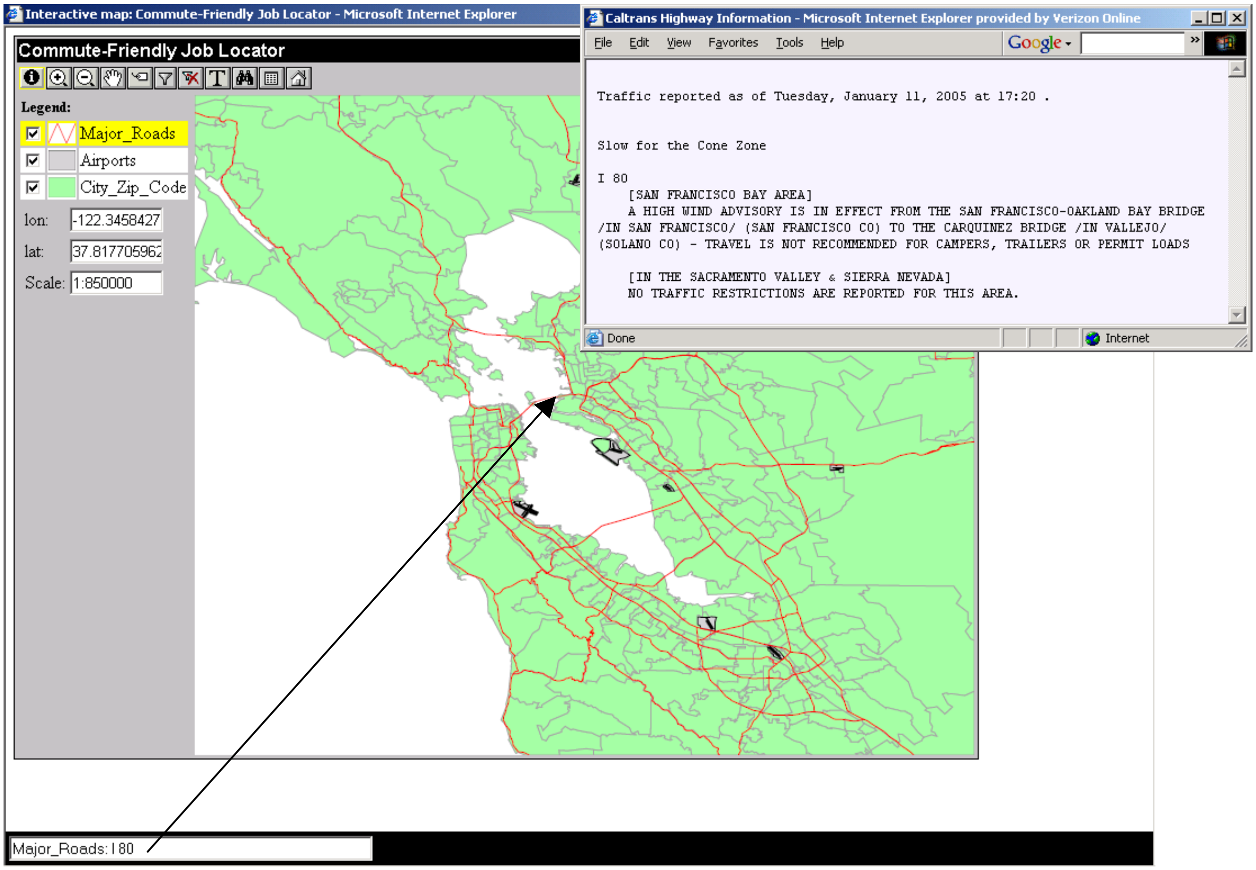

FIGURE 8 – CalTrans Web Service

4. Internet-based Geographic Information System Evaluation

A formative evaluation was conducted of this Internet-based GIS using an online survey to collect data regarding System Usability and Quality in Use. Volunteer subjects were drawn from two groups: graduate students drawn from four graduate level courses and employees of a private, non-profit organization comprising account managers and administrative personnel. This non-profit organization is dedicated to helping commuters find and use commute alternatives. These two groups represent novice and expert users respectively. Each subject was given two role-playing scenarios to complete. Each scenario is a simulation of a situation where the Proximate Commuting system would be used to locate a commuting alternative for a commuter.

4.1. System Usability and Quality in Use

The online survey consisted of the System Usability Scale (SUS), a simple, ten-item scale giving a global view of subjective assessments of usability (Brooke, 1996). With the SUS, a statement is made and the respondent then indicates the degree of agreement or disagreement with the statement on a 5-point scale. The selected statements cover a variety of aspects of system usability, such as the need for support, training, and complexity, and thus have a high level of validity for measuring the usability of a system. The SUS provides a single number, which is a composite measure of the overall usability of the system being studied. SUS scores range from 0 to 100. In addition to the SUS, five open-ended questions focusing on Quality in Use benefits were included. Respondents were asked to comment on their experience with the Proximate Commuting system as it related to the following benefits (Bevan, 1999): increased efficiency, improved productivity, reduced errors, reduced training, and improved acceptance.

4.2. Evaluation Results

As stated above, these two groups represented novice (N = 17) and expert users (N = 6). To verify that these two groups truly represented novice and expert users, a *t* test for independent samples was performed to determine if there was a significant difference between the means of the two groups. Results of this *t* test indicated that no significant difference between the means of the two groups existed. The significance value for the Levene test was greater than 0.05 (0.687) indicating that equal variance for both groups could be assumed. The significance value (two-tailed) for the *t* test was also greater than 0.05 (0.106) and the 95% confidence interval for the mean difference contained zero (lower = -27.54, upper = 2.84) both indicating that there was no significant difference between the two group means. Consequently, since no significant difference between the two groups existed, the SUS scores from both groups could be combined. Descriptive statistics from this combined group are presented in Table 1.

TABLE 1 – Descriptive Statistics

	N	Range	Minimum	Maximum	Mean	Std. Deviation
SUS Score	23	57.50	35.00	92.50	69.13	16.02

The descriptive statistics of this combined group N = 23 and Mean = 69.13 compares favorably with a similar study conducted for the European Union regarding the Networked Social Science Tools and Resources (NESSTAR) system – an Internet-based system for publishing statistical data. It was reported (Musgrave & Ryssevik, 2000) that the mean SUS score for the alpha version was 69 (N = 72) and 69.5 (N = 53) in the beta version. In addition, study participants comprised different user types (e.g. statisticians, students, researchers, and European Union representatives) and were drawn from different organizations (e.g. German Data Archive, Copenhagen University, and NESSTAR).

Subjective responses to the five open-ended questions focusing on Quality in Use benefits indicated that overall, users were satisfied with the system. In general, users liked the system and believed that it would be helpful in completing the task at hand. Suggestions for improvement included adding online help / training as well as making minor improvements to the user interface.

5. Discussion

For many organizations, the increasing availability of mature open source software and the Service-Oriented Architecture of web services offer new alternatives to traditional methods of information system development. The availability of thousands of open source software applications should help these organizations to more fully meet the needs of the end-user. Web services offer an innovative means for organizations and their partners to share information and applications in ways not possible or as easily until recently.

The results of this study indicate that this is no less so than with Internet-based GIS development. Specifically, open source software and web services were successfully utilized in the development of an Internet-based GIS, a Proximate Commuting system (Objective 1). In addition, the preliminary findings from the formative evaluation indicate that this system addresses both software quality issues as well as meets the needs of its users (Objective 2). These findings suggest that open source software and web services are both effective and efficient methods of information system development.

It has been observed (Green, Kennedy, & McGown, 2002) that increasing international competitiveness and technological advances have prompted the need for organizations to:

- Exploit emerging technologies more rapidly,
- Reduce design time-scales,
- Provide "right first time" design,
- Innovate more frequently and produce more innovative products, and
- Improve the reliability of products and systems.

It is clear that Internet-based GIS development, enabled by open source software and web services, meets these needs.

6. References

Abel, D. J., Taylor, K., Ackland, R., & Hungerford, S. (1998). An Exploration of GIS Architectures for Internet Environments. *Computers, Environment and Urban Systems, 22*(1), 7-23.

Anderson, G. (2003, June 2003). The Door Opens for Open-Source GIS. *GeoWorld, 16,* 42-45.

Ayala, D. (2004). *NuSOAP Home*, 2005, from http://dietrich.ganx4.com/nusoap/

Ayala, D., Browne, C., Chopra, V., Sarang, P., Apshankar, K., & McAllister, T. (2002). In *Professional Open Source Web Services*: Wrox Press, Inc.

Bevan, N. (1999). Quality in use: Meeting user needs for quality. *The Journal of Systems and Software, 49*, 89-96.

Bollinger, T. (2003). *Use of Free and Open Source Software (FOSS) in the U.S. Department of Defense* (No. MITRE Report Number MP 02 W0000101 v1.2.04): The MITRE Corporation.

Brooke, J. (1996). SUS: A 'Quick and Dirty' Usability Scale. In P. Jordan, B. Thomas, B. Weerdmeester & I. McClelland (Eds.), *Usability Evaluation in Industry* (pp. 189-194). London: Taylor & Francis Ltd.

Doernhoefer, M. (2001). Surfing the Net for Software Engineering Notes. *ACM Software Engineering Notes, 26*(1), 11-13.

George, J. F. (2000). The Origins of Software: Acquiring Systems at the End of the Century. In R. W. Zmud (Ed.), *Framing the Domains of IT Management: Projecting the Future Through the Past*. Cincinnati, OH: Pinnaflex Educational Resources, Inc.

Green, G., Kennedy, P., & McGown, A. (2002). Management of multi-method engineering design research: a case study. *Journal of Engineering Technology Management, 19*, 131-140.

Grimshaw, D. (2000). *Bringing Geographical Information Systems Into Business*. New York, NY: John Wiley and Sons, Inc.

Hagel, J. (2002). *The Strategic Value of Web Services* (No. November 2002). New York, NY: McKinsey and Company.
Hissam, S., Weinstock, C., Plakosh, D., & Asundi, J. (2001). *Perspectives on Open Source Software* (No. CMU/SEI-2001-TR-019). Pittsburgh, PA: The Software Engineering Institute, Carnegie Mellon University.
Hoffer, J. A., George, J. F., & Valacich, J. S. (1998). *Modern Systems Analysis and Design*. Reading, MA: Addison-Wesley.
International Institute of Infonomics. (2002). *Free/Libre and Open Source Software: Survey and Study*, 2003, from http://www.infonomics.nl/FLOSS/
Musgrave, S., & Ryssevik, J. (2000). *NESSTAR Final Report*. Brussels: Networked Social Science Tools and Resources.
Naumann, J. D., & Jenkins, A. M. (1982). Prototyping: The New Paradigm for Systems Development. *MIS Quarterly, 6*(3), 29-44.
Netcraft. (2001). *Netcraft Web Server Survey - June 2001*, 2005, from http://www.netcraft.com/Survey/index-200106.html
Open Source Initiative. (2003). 2003, from http://www.opensource.org/index.php
Oracle Corporation. (2001). *Oracle 9i Application Server: Web Services Technical White Paper*. Redwood Shores, CA: Oracle Corporation.
President's Information Technology Advisory Committee: Panel on Open Source Software for High End Computing. (2000). *Developing Open Source Software To Advance High End Computing*. Washington, DC: National Coordination Office for Computing, Information, and Communications.
Rodriguez, D. A. (2001). *Proximate Commuting: Hype or Potential? An Evaluation* (No. Transportation Research Record 1675). Washington, DC: Transportation Research Board, National Research Council.
Security Space. (2005a). *Web Server Survey January 1st, 2005*, 2005, from http://www.securityspace.com/s_survey/data/200412/index.html
Security Space. (2005b). *Apache Module Report January 1st, 2005*, 2005, from http://www.securityspace.com/s_survey/data/man.200412/apachemods.html
The GNU Project. (2003). 2003, from http://www.gnu.org/
The PHP Group. (2005). *Usage Stats for December 2004*, 2005, from http://www.php.net/usage.php
Victoria Transport Policy Institute. (2002). *Online Transportation Demand Management Encyclopedia: Commute Trip Reduction (CTR)*, 2003, from www.vtpi.org
Wichmann, T. (2002). *Use of Open Source Software in Firms and Public Institutions: Evidence from Germany, Sweden and UK*. Berlin: Berlecon Research.
World Wide Web Consortium. (2002). *Web Services Architecture: W3C Working Draft 14 November 2002*, 2004, from http://www.w3.org/TR/ws-arch/
World Wide Web Consortium. (2003). *Scalable Vector Graphics (SVG) 1.1 Specification W3C Recommendation 14 January 2003*, 2004, from http://www.w3.org/TR/SVG11/