MAP-BASED ADVERTISEMENT SYSTEM IN THE WORLD WIDE WEB

Risto Vuorjoki* and Juha Ylä-Jääski*

Keywords: WWW, Multimedia, Publishing, Newspaper, Interface

Abstract

This paper describes the design and implementation of a World Wide Web (WWW) application that allows users to look for classified advertisements based on the locations of the advertised objects. The application lets users limit a search to a certain area by working directly on a map. The locations of the advertised objects are visualized on the map.

New WWW browser programs allow interactive programs written in the Java language to be included in WWW services. More traditional CGI (Common Gateway Interface) scripts can also be used. Database systems can be called from other programs and can provide a basis for the advertisement service. Making use of geographic information systems in a similar way has not been as straightforward until lately.

The advertisement visualiser application includes a client written in the Java language and a CGI program that looks for maps and advertisements stored in a server computer. The application also includes information that is necessary for showing any street addresses on the map. The system is already in routine use through a WWW service provided by Helsingin Sanomat, the largest newspaper in Finland. The database of the service consists of housing advertisements in the larger Helsinki area.

1. Introduction

The WWW is a proper media for services that intend to attract non-expert or occasional users to search for information which they do not need regularly or which they can retrieve from other sources, but not in such a convenient manner. Such users are not likely to pay significant sums of money for sophisticated software to find the information they need.

* VTT Information Technology, P.O. Box 1203, FIN-02044 VTT, Finland

An alternative to the WWW is to distribute a separate program which users can install in their computers. Such a program calls, however, for regular updates, as the information to be searched for is likely to change. A strength of the WWW is that it allows regular updates which only need to be performed once to enable all users to see the changes immediately. It is also much easier for occasional users to connect to a WWW service than to try to install different programs for all the services they wish to use. Naturally it would be possible to use networks other than Internet and the WWW, but no other currently used computer network can be accessed by as many people as Internet.

Typical WWW pages include more or less static information about the service provider and its services such as company and product presentations. Frequently services also contain general information about every-day life or information which is useful for professional tasks. Magazines and films currently shown in cinemas are examples of the former, research projects published by institutes and universities of the latter.

In addition to static information, many WWW pages provide means to look for information on different subjects, either in databases within the server or in other WWW servers. Due to the wide variety of information available in the WWW and the vast number of WWW servers, there are several search engines that find information available in the WWW according to user-specified keywords.

The WWW also provides features to allow the user to give some feedback and have an influence on the service. CGI scripts (Dwight, 1996) provide the means for a service to respond to user actions. A CGI script is a program that runs in the server computer and is called when the user selects a certain link on a WWW page. CGI is essentially an interface that allows a WWW server program and other programs in the server computer to communicate.

CGI scripts have made it possible to allow users to access databases through the WWW. The users type in their search criteria and the script outputs the results in a suitable format. The database itself is stored within the server machine and is not accessible to WWW browsers without calling the scripts.

Java (Gosling, 1996) is a new object oriented programming language very much like C++. It was introduced in 1995 and has since then become increasingly popular. The main reason for Java to have gained so much attention during the past year is that it provides advanced features for Internet programming in general and WWW programming in particular.

The WWW features of Java have been made possible by machine independence. Any Java program written in any computer or operating system can be run in any other Java-supporting system with no need for modifications. Traditionally, WWW pages have contained mostly static data — e.g. text and pictures — that the user can read or view. With Java, it is possible to include runnable programs, applets, in WWW pages. When a user reads a WWW page that includes applets, the browser downloads the applets and runs them in the user's computer. This way WWW pages can contain dynamically changing data as well as traditional static data. The user can also interact with the program e.g. using mouse just as in any other program.

Maps have constituted an integral part of WWW services practically as long as there have been graphical browsers. Ordinary printed maps can only cover one area at a time or one type of a map at a time and any additional information calls for new maps to be looked for. The WWW allows users to select which area or scale they want to view, or possibly the type of information they want to see. While any GIS (Geographical Information System) provides this and much more functionality, such systems are too expensive for an occasional user.

The first WWW maps have provided only static images to be viewed, perhaps to show the location of the server in question, but more interactive maps have also been introduced. Typical examples include maps of certain countries that show the physical locations of WWW servers and allow users to select those servers through the maps. There are also map services with more advanced features such as interactive search.

To use a map-based service in the WWW does not set many requirements for the WWW browser. The most important requirement, the capability to view images, is met by all graphical browsers. Difficulties may be introduced by the capacity of the communications line used. If users are satisfied with their communications capacity with other illustrated pages, however, they are presumably satisfied with the rate at which map images are being received as well.

In this paper, an advertisement visualiser application is presented which allows the user to specify a geographic area on which to look for advertisements and to show the results of the search on a map. Instead of selecting from a list of predefined search areas, the user is able to specify the area directly on the maps. In addition to the locations, the application is capable of showing the user additional information on the advertisements by following a link to another page.

The system has been designed to support different types of advertisements: housing, yellow pages, tourist attractions and other information which is tied to a location. In order to let users access the service in a simple manner, the system operates in the WWW. The maps are changeable so that the application is not tied to a fixed set of maps. This enables the software to be used as a basis for further applications or for integrating the system with other pieces of software containing e.g. more powerful search features.

The system has been integrated into the WWW service of Helsingin Sanomat, the largest daily newspaper in Finland. The user interface has originally been implemented using the Java language, but an alternative version not using Java has been developed to allow the service to be accessible to a wide audience.

2. Basic system components

In order to implement a system that shows advertisements on maps, three basic components are needed: maps, advertisements, and information to convert street addresses to map coordinates.

Maps in the advertisement visualiser are raster images in GIF format covering the area of Helsinki in three different scales. An overview of whole Helsinki is seen in a general map. The more detailed versions cover approximately half of the city in a number of rectangular, non-overlapping maps. As it is considered useful for the users to be able to view any location in the centre area of a map, overlapping map images have been created. The overlapping maps shift usually in units of half a map image size. The images have been created as a preprocess before attaching the maps to the application

Creating overlapping images obviously requires more disk space than storing the original images only once, but it enables smoother transfer from one image to another. It also allows the users to view any single advertisement close to the centre of a map instead of close to the border where the immediate surroundings cannot be seen. The number of map image files in the advertisement visualiser - including all overlapping maps in all three scales - amounts to approximately 750 and the combined size of them 15 megabytes. One image file takes typically 10—30 kilobytes of disk space.

An alternative solution to let the users move between overlapping maps is to create overlapping map images dynamically at run time, as opposed to creating static maps in advance. This method allows more flexibility than the one used, as the maps can overlap in different units depending on the situation, instead of always overlapping with half the image size. This way any point in the map can be made the exact centre point of what is shown on the screen. This has not been implemented, however, since the ability to show the location of an advertisement in the exact centre has not been necessary. It has been considered sufficient for the user to be able to view any advertisement in the overall centre area.

The use of raster images has been a choice of convenience during development. Had vector maps been used, scrolling and zooming the maps could have been made as accurate as desired. With raster images, zooming in and out requires different map image files for different zoom levels in order to gain reasonable results. For the task of advertisement visualisation, however, raster images allow a simple implementation that only needs to know the coordinates of the corners of the map images in order to show locations on them. While the maps are detailed enough for showing locations on them, the application does not have to take care of proper visualisation or scaling of vectors

An address-to-coordinate database consists of a table listing all street addresses in the city together with corresponding map coordinates. Using the table it is possible to show the location of an advertisement based on its address, as long as the coordinates and scales of individual map images are known. The accuracy of the database is sufficient for advertisement visualisation. The addresses listed include streets and numbers of all individual houses and buildings in the city. Different doors in blocks of flats are not listed.

The advertisement visualiser relies on having an address-to-coordinate database. If none is available for a certain city or area, new functions have to be added to perform address-to-coordinate conversions. In Finland, address-to-coordinate information is readily available for most towns and cities.

Also a comprehensive database is available for Finland which includes all streets and roads and corresponding address information even for most rural areas of the country. The streets and roads are given in a vector format as separate segments, and for each segment the name of the road is given together with street numbers at each end of the segment and on each side of the road. It is possible to calculate the coordinates of all individual addresses based on that information by using interpolation. As segments can at the longest be from one road crossing to the next, this method is accurate enough for most purposes.

The advertisements in the advertisement visualiser are taken directly from an SQL-database of Helsingin Sanomat. Each advertisement includes fields for address, price, part of the town the house lies in, type of the kitchen and type of the building (e.g. house, block of flats), although not all the advertisements contain values for all those fields.

3. Implementation

To implement the advertisement visualiser, two separate areas have to be dealt with. First, there must be a user interface that allows the users to specify areas on maps and search for advertisements. The user interface must also be capable of showing map images and locations on them as well as further information on advertisements that have been found. Second, there must be a module in the application that performs the actual search operations and finds the appropriate map images to be shown. Since the system is to work in the WWW, a means of communication over the network must also exist.

The general approach has been to divide the application in two parts: a client and a server which communicate over the Internet network using common WWW methods. The client is a Java applet and the server a CGI script written in the C programming language. The advertisement information and the map images are stored in the same computer as the server program and are transmitted over the network when requested by the client.

The application handles advertisements and maps as tables of appropriate objects. A map consists of fields similar to those in the map index files. An advertisement contains fields for the address, the map coordinates, the number of rooms etc. Maps and advertisements are defined in the server as C language structures. In the client applet the structures are represented as Java classes, consisting of a subset of the fields.

3.1 Client

The client has been written in the Java language. Its main function is to provide a user interface for the application and show the maps and the advertisement information on the user's screen. The client interprets the user's actions and contacts the server for a response. It doesn't perform actual search operations but reads in advertisement information found by the server and shows the locations of the advertisements on the map. It is provided with sufficient information on the maps so that it can convert map coordinates to screen pixels and vice versa, draw markers for the advertisements and decide whether or not a mouse click has occurred over one of the markers.

The client is designed to include as little information on the advertisements and maps as possible. This enables it to be adapted to other types of applications easily. In fact, in the event of customising the system for another field of use, the Java client would require the least modifications. In particular the client does not know of the types or contents of the search criteria fields. All the criteria the user specifies are sent to the server as plain text. Similarly, the client reads all the information on any advertisements as text only and shows them to the user with no modifications. The field names shown beside the search criteria fields are read from a file on the server computer. This information is read only once, during the initialisation of the applet. Should any of the meanings or types of any advertisement fields change, the client needs not be modified.

When showing maps the client needs more information than when transmitting search criteria and receiving advertisement information. As with the criteria field names, information on different zoom levels is read once during initialisation. This information includes the scales and sizes of the maps on each level. Information on individual maps is not read until a particular map is needed.

The advantage of reading zoom level information during initialisation is that no read operations are needed later. As the number of levels is likely to be relatively small even in large services, this does not cause a significant delay. The number of individual maps, on the other hand, is larger and loading information on all of them caused, when tested, a noticeable delay in the initialisation even over a fast network connection.

Communication between the client and the server is performed with CGI-scripts. The client makes use of a feature of all Java applets that enable them to contact the host computer. The server runs as a CGI script and the client calls it with necessary arguments such as search criteria or a request for a new map image. The server responds with either a list of advertisements found or a new map, depending on the type of the call.

When the client requests for new map images, it receives a pointer to the new map image together with information on the map. The information includes the coordinates of the map so that the client can place markers for advertisements on the correct positions. The client discards old images when new ones are loaded. If users move back to maps that they have visited previously, the system has to load the map images again.

In addition to enabling easy customisation, the design of the client allows both the maps and structure of the advertisements to be updated simply by modifying the initialisation files. The service administrator can thus update the service without the need to change the client program.

3.2 Server

The server program takes care of searching for advertisements and map images. The client does not have direct access to the advertisements or the map images. When the user wants to perform a search or view a new map, the client sends a request to the server. The server then fetches a new map image or performs a search, according to the type of the request, and transmits the results to the client.

The server has been implemented using the C language and communication between the client and the server is performed as CGI calls. The client calls the server using necessary arguments which indicate the desired action of the user. The server obtains the arguments in a format common to CGI scripts and can transmit information to the client by outputting it using standard C functions.

After receiving a request from the client, the server first divides the input string into several substrings, each representing one argument. The values of individual arguments are thus available if they are needed later. If the arguments indicate that a search is to be performed, the server starts by further parsing the search criteria. Embedded SQL is used to call the database management system which finally performs all search operations.

Embedded SQL is an extension to a programming language and it allows the program developer to include SQL statements in a program. Thus it is possible for the program to search for data in a database without the need to implement complex search algorithms. Establishing and maintaining the database can be left to the Database management system in question. Formally, a program using embedded SQL is precompiled by an SQL preprocessor to normal program code that is then compiled by an ordinary compiler. Embedded SQL can be used in many common programming languages, including the C language.

When the client requests for new maps, the procedure is somewhat similar to advertisement search. Initially the server transmits the client the first map to be shown together with the index of the map and the coordinates of its south-west corner. When the user wants to move to a different location, the client sends a request that includes the index of the current map and the direction (zoom in, zoom out, north, south, etc.) the user wishes to move to.

After receiving a request for a new map image, the server first looks for the information on the current map in the index files. This search is performed as a binary search, based on the index of the map. The information for the current map contains indices of the maps that are next to it in each direction. Thus the server can look for the new map based on its index, according to the direction the user wishes to move. Finally, the information on that map is sent to the client.

When a map image including a certain location needs to be found, as when the user wishes to zoom into a location, the server scans through the whole index file of the appropriate zoom level. It compares the coordinates of the specified location to the coordinates of each map on the zoom level and returns the map that includes the location closest to the centre of the map image.

A notable feature with the server is that as a CGI script it is started separately each time a user sends a request. Consequently, the server reads the index files for the maps and zoom levels from the disk each time it receives a request to get a new map. The index files are not large, however, and reading them repeatedly from a C program does not cause a noticeable delay in the application. The time of loading the files from disk is insignificant compared to the time to establish network connections and to transmit the map images.

3.3 CGI version

The CGI version which does not use the Java language provides an example of an alternative way to implement the service. It does not have a client process, since Java is basically the only way to include machine-independent runnable code in WWW pages. Instead, all necessary input is typed or selected by the user in form fields which are a part of the WWW page. The contents of these fields is transmitted to the server by the browser program by standard CGI conventions. The browser and the server perform this automatically with no intervention from the CGI program or the program developer. When a user performs a mouse click on a map image the position of the mouse is transmitted. The server responds with a new HTML page with the map image included as an ordinary HTML image tag.

The server for the CGI application is implemented as an extension to the server. In practice all the functionality which is part of the Java client has to be added, since there is no client to perform those tasks. This involves controlling which advertisements to show on the maps and drawing markers for those particular advertisements. The server must also maintain the persistent state of the actions of the user i.e. keep track of which advertisements have previously been found and which map the user is currently viewing. This allows the user to navigate properly and without a need to perform a new search every time a new map is shown.

In the actual implementation the two servers, one for the Java client and the other for the CGI user interface, are combined into a single process. This way the two servers can easily share common parts of the program code. After parsing the arguments the process jumps to different functions depending on which client has sent the request.

The map files are stored in a GIF format. In order to draw advertisement markers on top of the images, the GIF files have to be loaded and their contents have to be interpreted. This is performed using a publicly available C library handling GIF images and providing functions to manipulate the images in a simple and effective manner. The marker for an advertisement is a small arrow that is stored in another GIF file. The arrow is copied to appropriate locations on top of the map images using standard library functions.

When advertisements are to be visualised, the server program loads a GIF image from the disk, draws the markers and outputs the results to a new, temporary file. The name of this file is included in the server's output so that the user's WWW browser can display the correct image.

The persistent state of the user's actions is controlled by including the necessary information in the HTML page that is transmitted as output. HTML forms can include hidden fields that the user cannot see but that are transmitted back to the server as a part of a form.

4. Security issues

When working with a service which is available in a public computer network security is an important factor. It has to be ensured that no unauthorised user can break in to the server and perform any damage in it. It may also be desirable that access to the information provided by a service is limited, and only predefined users are allowed to access the service. Security issues are especially important in the Internet and the WWW since the underlying protocols are known to be insecure.

Protection against breaking into a computer in a network is generally taken care of by the operating system and administrative personnel of the server computer. The WWW server program must also be secure enough so that nobody can use it as a passage into the system. To have a WWW service provide information to certain users only is the responsibility of the WWW server program and its administrators. Most WWW servers provide means to restrict access to a service.

In spite of the security measures built in WWW servers, CGI scripts can be a source of security leaks. A CGI script is referenced by a URL and thus it can be referred to from any WWW page, not only from the page that is intended as its host page. Any user or any program acting as a WWW browser can call the CGI script using any arguments the script may or may not expect. The script therefore cannot trust that it is called from a certain page using certain known arguments.

Especially, the script should not work unexpectedly or cause the underlying system to work unexpectedly if it receives incorrect arguments. Nor must it use such default values for arguments that may cause the script to return information that is not intended to be accessible by all users.

Security is also an issue with Java applets in the WWW. The problems involve, however, mostly applets that try to attack the computer they are being run in. This has to do with programmers developing hostile applets and distributing them in their own WWW pages. Such applets can try to find leaks in the security features in Java or in WWW browsers and cause damage in computers that load those WWW pages. As for the moment, there have not been reports on security problems that involve applets being a threat to the service they belong to.

The advertisement visualiser is integrated into a public WWW service in such a manner that the WWW server and the underlying operating system take care of access restrictions that are necessary. The security measures in the advertisement visualiser involve handling of incorrect input to the CGI script. In the case of unrecognised arguments, the server produces an error message. Furthermore, the server never writes to any files and does not access any information that is not publicly available. Therefore it is not expected to cause security leaks even if it were made to act incorrectly.

5. Discussion

The Java language provides elegant features, but as an interpretable language it is far too slow for the server of the application. Reading from and writing to a large file in a disk in Java are an order of magnitude slower than in C or C++. This can be understood considering that a large file involves a long loop in the program and the Java virtual machine has to interpret the commands every time the program goes through the loop. This is a typical example of a situation where so called just-in-time compilers are likely to improve performance significantly. A just-in-time compiler is essentially a virtual machine that compiles each method the first time it is called. Thus in a long loop compiling is performed only once.

In the user interface the speed is not as important, and Java has proved to be a proper tool. Java also provides the only currently available way to implement a WWW service with the same level of user interaction as in our application. Java has also other benefits. Many operations are more straightforward to implement using Java than with CGI scripts, since Java applets are constantly running but CGI scripts have to be started again every time when a call is made. To provide the CGI script with information on advertisements which have have been found requires information to be transmitted as part of the HTML code to the browser and then back to the new CGI call. An alternative is to perform the same search again. The Java applet runs constantly and does not have to discard information unless it explicitly wants to. Network connections in the Java client can thus be used to transmit essential information only.

The Abstract Window Toolkit that provides Java classes for a graphical user interface seems to suffer from minor defiences. Most importantly, machine independence has not worked as well as expected. The Java client does work on different platforms, but the early implementations on Java virtual machines on different platforms and in different browsers have caused problems. Some mouse events seem to differ slightly between different platforms. Also, the map images have not always shown up properly in certain environments. These problems have diminished with new Java versions, but some users may still experience problems when using the service.

We are currently investigating possibilities to extend the utilisation of the system to cover a wider range of advertisements in electronic versions of newspapers as well as yellow pages telephone directories. The system can also directly be used to visualise other information involving geographic information such as tourist attractions. Many potential applications involve, besides a mere visualisation of a location on a map, also the task of finding the best route between two locations. We are also investigating possibilities to utilise our experience in routing methodology (Linnainmaa, 1994) to develop more advanced WWW services. Further, new technologies such as intelligent agents represent a potentially interesting field to provide more advanced WWW services.

6. References

Dwight J., and Erwin M.
1996. "Using CGI, special edition" Que Corporation, 828 pp.

Gosling J.,and McGilton H.
1996. "The Java language specification" Addison – Wesley, 864 pp.

Linnainmaa S., Savola J., and Jokinen O.
1994. "EPO: A knowledge based system for wood procurement management" Proc. of the 7th IAAI Conference, AAAI Press (Ca) pp. 107-113.