Development of Web-based Interactive Map Using Object-Oriented Programming Concept

Adeyinka A. **Adewale** and Jeremiah O. **Onaolapo**

***Abstract* — The program incorporates an interactive map which responds to origin and destination selection, by analyzing the relative positions of both locations and creating real-time routes on the road network to display to the user the required path from the origin to the destination and the approximate distance/time required. System design is based on the Model-View-Controller (MVC) design pattern, and the application has been developed using Adobe Flash CS3 (with ActionScript).**

***Keywords* — Adobe Flash CS3, Algorithm, Design Pattern, High Power Object Oriented Programming, Model-View-Controller, real-time mapping.**

I. Introduction

THE problems of navigation have always been a contemplated issue with various institutions and companies deploying solutions to help people get from place to place, reducing the chances of losing their way and ending up at the wrong destination. Various solutions such as high-powered Global Positioning Systems (GPS) and other devices have been developed, which require costly equipment that communicate with satellites which in turn beam back their present geographic location, direction, distance from final destination and the route to traverse to get to this final destination. Such systems are difficult to acquire due to high cost of equipment and deployment.

As a result, there is need for an economic solution that is accessible to all users (the Internet being the most preferred option) without any need to acquire specialized equipment. This project seeks to solve those problems by harnessing the power of a leading graphic application that allows development of PC-specific multimedia and web applications; Adobe Flash CS3 and incorporating its full-fledged object oriented programming language, ActionScript 3.0, to implement all its aspects [1].

Studies have shown that object-oriented approach to software development (compared with traditional approach) results in increased product flexibility and reduced complexity [2]. In addition, object-oriented design patterns help to ensure reusability and consistency in software development projects if handled properly [3].

The Model-View-Controller (MVC) design paradigm was applied in the course of this software development project, since MVC architecture helps to decouple user interface presentation from business logic and data retrieval, hence simplifying interactions between objects [4]. Most interactive applications nowadays are developed based on MVC design pattern [5], for instance, a web application generator [6].

The Adobe Flash CS3 Integrated Development Environment (IDE) is equipped with tools that were used to design all graphical components of the project; it also works in conjunction with other Adobe Creative Suite applications such as Adobe Photoshop [1]. These other applications can speed up workflow and also provide additional content not present in the Flash IDE for graphical component design. The components can then be manipulated using some of the functions and tools to create interactive properties and behaviors that would improve user experience in the application.

ActionScript 3.0 is embedded and associated with Flash projects as backend coding to provide exceptional features and control that cannot be provided by the IDE tools, but similar to those provided by languages like Java and C. These two parts of the Flash package can be merged together to create an application with high user interactivity and also powerful algorithms which could be deployed to multiple users as an online application [1].

II. Web Map

Most maps are two-dimensional; representing position data on a two-axis Cartesian-system, or three-dimensional representation of geometric space. Either could be presented in an interactive and/or dynamic manner. The advent of technology has greatly advanced the functionality of maps via the use of electronic maps that have been implemented to provide extra features such as rainfall levels, population distribution etc. Demographic data integrated into these maps makes for more efficient analysis and better decision-making. One of such map is the Geographic Information System (GIS) map. GIS is an organized collection of computer hardware, software, geographic data, and personnel designed to efficiently capture, store, update, manipulate, analyze, and display many forms of geographically referenced information [7].

While the first web maps were primarily static, due to

Corresponding Adeyinka A. Adewale is a Lecturer in the Department of Electrical and Information Engineering, Covenant University, PMB 1023 Ota, Nigeria (phone: +234.806.885.5772; e-mail: ade.adewale@covenantuniversity.edu.ng).

Jeremiah O. Onaolapo is a Research Fellow in the Department of Electrical and Information Engineering, Covenant University, PMB 1023 Ota, Nigeria (phone: +234.813.225.2885; e-mail: jeremiah.onaolapo@covenantuniversity.edu.ng).

technical restrictions, today's web maps can be fully interactive and integrate multiple media. This means that both web mapping and web cartography also have to deal with interactivity, usability and multimedia issues. Static web pages are view-only with no animation and interactivity. They are only created once, often manually and infrequently updated. Typical graphics formats for static web maps are Portable Network Graphics (PNG), Joint Photographic Experts Group (JPEG), Graphics Interchange Format (GIF) or Tagged Image File Format (TIFF) for raster files, Scalable Vector Graphics (SVG), Portable Document Format (PDF) or Small Web Format (SWF) for vector files. Web maps are created dynamically on demand each time the user reloads the web page, often from dynamic data sources, such as databases.

There are general-purpose navigation sites developed by Yahoo! Corporation [8] and Google [9], both leading search engine applications. They exist on the internet as platform-independent applications that are easily accessible by users. An interactive map display contains worldwide geographical information about all locations on the planet. Geographical maps make use of static maps, aerial satellite imagery or a hybrid of both to provide geographic information to users, including traffic route display capabilities. Development tools are Adobe Flex Builder and ActionScript 3.0, eXtensible Markup Language (XML) and backend support includes map servers, database and aerial satellite updates. Others examples of navigation sites are Rail Europe [10] and Flash Earth [11]. These are global maps containing worldwide geographical information about all locations on the planet.

Meanwhile, the classification of web map interactivity can be done from three perspectives: navigation techniques, support for collaboration, and data sources available. Navigation techniques refer to the way a user can change the viewpoint of the map, such as through zooming and panning. Support for collaboration refers to an application's ability to allow multiple people to communicate ideas. For example, the map software may allow users to chat in real-time or write comments on the map for later viewers. Data source availability refers to the options available for displaying different data on the map, such as displaying different layers. In this report more emphasis would be laid on map interactivity. The statistical analysis in Fig. 1 explains interactivity of a finite sample of maps.

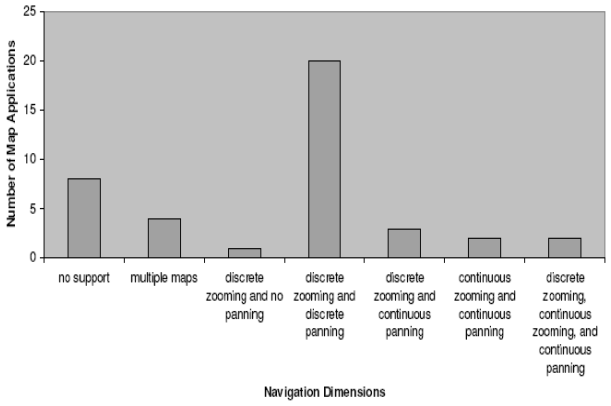

Fig.1. Map survey results from navigational dimension of classification.

From the reviews highlighted above, the provision of navigational facilities over the web has reached a high level of maturity and everyone wants to partake; from institutions that want to increase their international ratings to organizations that offer services from usage to actual development.

III. Application Design and Development

To optimize the look-and-feel of the application, the Graphical User Interface (GUI) was designed using the Adobe Photoshop application as our editing application to design and create all the site graphical components. Adobe Flash and Photoshop work interchangeably as they are part of the same production suite package, with content easily interchangeable between the two.

Preloaders are essential to flash applications as they ensure that all components and assets necessary for it to run effectively are loaded into the user cache memory before the application initializes. This is done to increase both the speed and ensure proper functioning of the application due to its dependencies on these assets. The preloader acts to load all components of the application; the SWF file, XML file and all other assets that are embedded into the application.

The code implementation pattern of choice is the Model-View-Controller framework pattern [3][5]. The model is the content being loaded into Flash, the controller class handles the actual loading of the external SWF and XML files and the view displays the loaded content, as shown in Fig. 2 below.

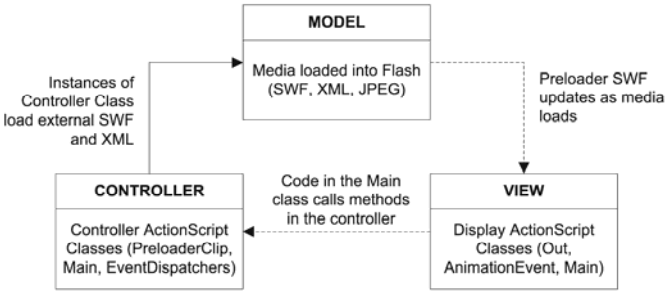

Fig. 2. Model-View-Controller interaction for the Preloader.

The pages are interrelated by a collection of buttons, which can be likened to hyperlinks on traditional HyperText Markup Language (HTML) websites, and how these hyperlinks relate together to form a web of movement. Every page is equipped with a set of icons that act as hyperlinks to enable the user move within pages and get to the feature required. This is illustrated with the chart below in Fig. 3 below.

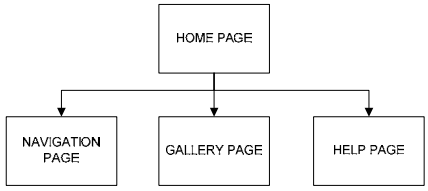

Fig. 3. Block diagram of system modules, which also define navigation flow within site pages.

The InfoBoxx class makes use of the XML class to display brief information to the user about the specific location clicked. Fig. 4 illustrates MVC interaction for Navigation Page.

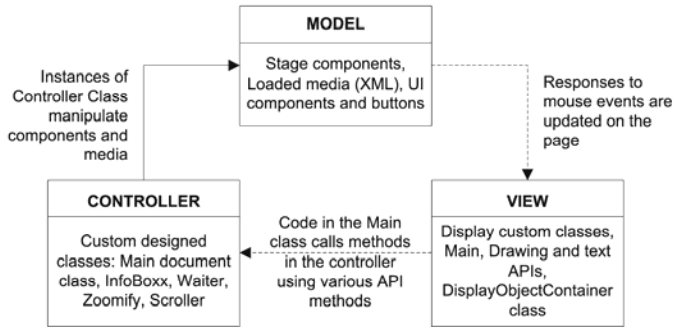

Fig. 4. Model-View-Controller Interaction for Navigation Page.

To make the map usable and optimized for both Flash and our application, the map was converted into vector-based graphic components which exist on different layers; allowing each element to be unique. Using Adobe Photoshop CS3 (a high-end graphic program for photo editing, vector drawings and other media), which supports compositing and layering the map layers, each element was designed and organized according to similarities and functions. Below (Fig. 5) is a cross-section of the map, displaying a snapshot of map conversion.

Fig. 5. Converted Covenant University Master Plan.

Besides the look-and-feel that can be added to the objects to improve user perception, the buttons also possess instance names which can be referenced through ActionScript [1] to allow them perform their functions. The following code snippet is the main function that allows these buttons to work as they should by employing timeline labels (frame identifiers which can be referenced using ActionScript).

```
navigationButtonInstance.addEventListener(
MouseEvent.CLICK, function )
function (){
    MainTimeLine.goAndStop( "LocationLabel" );
}
```

IV. Implementing the Routing Feature

The routing feature is the core function of the web app and this is implemented using User Interface (UI) components; combo boxes for user to select locations from a list (and for application to collect variables) and a 'navigate' button to start the routing process. The combo boxes fetch their data directly from an external XML file that contains location names and details. The XML file allows easy updating of information with as regards locations, as it is external to the application. The code snippet below shows a sketch form of the code listing to implement this feature.

```
var xmlObject:XML = new XML();
var xmlLoaderObject:URLLoader =
      new URLLoader();
xmlLoaderObject.load(new URLRequest("file.xml"));
xmlLoaderObject.addEventListener(
Event.COMPLETE, function);
function ( ){
      xmlObject = XML( );
      originComboBox.dataProvider = new
   DataProvider(xmlObject);
      destinationComboBox.dataProvider = new
   DataProvider(xmlObject);
}
```

Location inputs are collected from the combo boxes when user selects a specific option from the list, as shown in the listing below, while Fig. 6 shows the location combo boxes.

```
originComboBox.addEventListener(Event.CHANGE,
functionCollectOrigin);
destinationCB.addEventListener(Eve
```

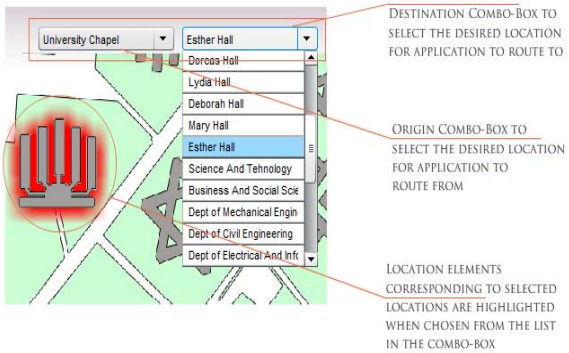

Fig. 6. The combo boxes used to collect location inputs for origin and destination.

After the user has selected the location pair of interest, to view the evaluated route it is necessary to click the navigate button. It is the means by which the algorithm begins the iterative process of analyzing the road point structure. The

code snippet below shows a sketch form of the code needed to implement this function, while Fig. 7 shows a computed route.

```
Navigate_btn.addEventListener(Event.CHANGE,
function);
function ( ){
    collectInputsfunction ();
    conditionalIterationFunction ();
}
```

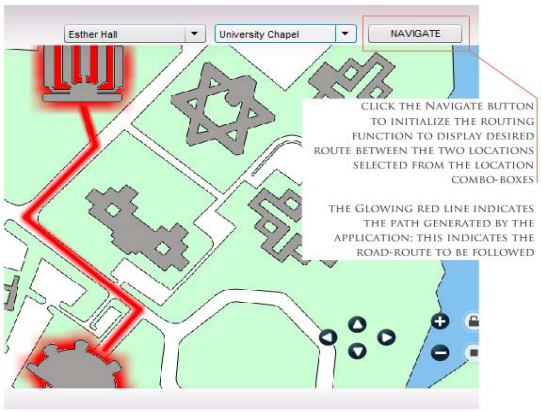

Fig. 7. Cross section of the application showing UI components and a sample path routed based on locations indicated in the combo boxes.

During the routing process, various elements are added and removed from stage; the Caution box which pops up when an invalid pair of locations are chosen, the processing box with a processing meter that plays while the routing algorithm is executed and the direction box which displays the textual directions, distance and time required to cover the computed distance. The processing box is invoked by calling the waiter class, so that the processing message box, which exists as a component in the library, is added to stage. It is invoked once the routing algorithm is executed and it serves to give the user visual feedback of the execution process. The code snippet below illustrates the process described above.

```
proccessingObject:Waiter;
if ( isRoutingBegin ) {
   processingObject = new Waiter ( objectParameter );
}
```

The caution message box also exists in the library and an instance of it is added to the stage in the event that a user selects an invalid pair of locations and tries to initialize the routing function. The asset is also invoked by calling the waiter class.

```
cautionObject:Waiter;
if ( invalidLocationInput ) {
   cautionObject = new Waiter ( objectParameter );
}
```

The Direction Box is invoked once the iteration process is complete, by adding an instance of the Direction object on stage (see Fig. 7).

```
directionObject:Direction;
if ( iterationEnded ) {
     directionObject = new Direction( textDirection,
     distance, requiredTime );
}
```

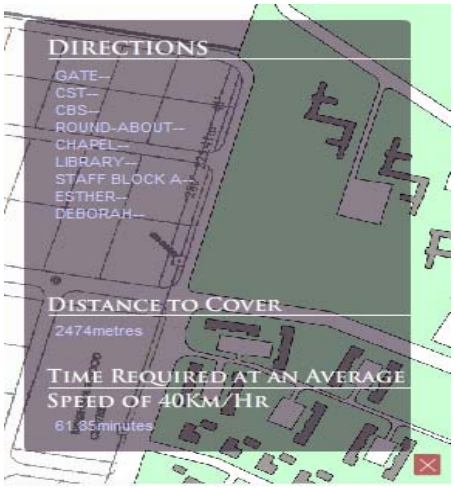

Fig. 8. Direction box showing information about the routed path.

V. Conclusion

This interactive map application satisfactorily provides preplanned journey arrangement for the user, allowing him or her to know the geographic structure of the case study location before setting out on the journey. Users can use the application from any computer with internet connection, and usually regardless of what operating system that computer is running. Installation is not required - updates and distribution of the application are handled instantly and automatically. In addition, being a web-based application, it is less prone to viral infection than running an actual executable.

Also, we have been able to implement user-interface behaviors not obtainable using only the HTML widgets available to standard browser-based web applications. Network traffic will be significantly reduced because an application-specific client engine can be more intelligent than a standard web browser when deciding what data needs to be exchanged with servers especially when based on OOP concepts. This can speed up individual requests or responses because less data is being transferred for each interaction, and overall network overhead load is reduced.

However, a major improvement on this application would be to develop a mobile version so as to ensure mobile access to the real-time navigation feature, that is, an effective hardware-independent GPS application that users can access on the move.

REFERENCES

[1] Adobe Systems Inc. (2007). *Adobe Flash CS3 Professional User Guide*. [Online]. Available: http://www.adobe.com/support/documentation/archived_content/en/flash/cs3/flash_cs3_help.pdf [Oct. 25, 2012].

[2] N. Munassar and A. Govardhan, "Comparison between Traditional Approach and Object-Oriented Approach in Software Engineering Development," *International Journal of Advanced Computer Science and Applications*, Vol. 2, No. 6, pp. 70-76, 2011.

[3] R. Sridaran, G. Padmavathi and K. Iyakutti (Mar.-Apr. 2009). "A Survey of Design Pattern Based Web Applications." *Journal of Object Technology* [Online]. Vol. 8, no. 2, pp. 61-70. Available: http://www.jot.fm/issues/issue_2009_03/column5

[4] T. Dey, "A Comparative Analysis on Modeling and Implementing with MVC Architecture," *IJCA Proceedings on International Conference on Web Services Computing (ICWSC)*, vol.1, pp. 44-49, Nov. 2011.

[5] A. Leff and J.T. Rayfield, "Web-application development using the Model/View/Controller design pattern," in *Enterprise Distributed Object Computing Conference (EDOC '01) Proceedings*. Fifth IEEE International, 2001, pp.118-127.

[6] S. Lazetic, D. Savic, S. Vlajic and S. Lazarevic, "A Generator of MVC-based Web Applications," *World of Computer Science and Information Technology Journal (WCSIT)*, vol. 2, no. 4, pp. 147-156, 2012.

[7] T.L. Nyerges, "Understanding the scope of GIS: Its relationship to environmental modelling," in *Environmental Modeling with GIS*, Ed. M.F. Goodchild, B.O. Parks and L.T. Steyaert, NY: Oxford University Press, 1993, pp. 75-93.

[8] Yahoo! Inc. *Yahoo! Maps*. [Online]. Available: http://maps.yahoo.com/ [Oct. 25, 2012].

[9] Google Inc. *Google Maps*. [Online]. Available: http://maps.google.com/ [Oct. 25, 2012].

[10] Rail Europe Inc. *Rail Europe*. [Online]. Available: http://www.raileurope.com/europe-travel-guide/ [Oct. 25, 2012].

[11] Microsoft Corporation. *Flash Earth*. [Online]. Available: http://www.flashearth.com/ [Oct. 25, 2012].