Web-based Plankton Data Visualisation

T.Y.Aung and P.S.Dowland

Network Research Group, University of Plymouth, Plymouth, UK
e-mail: info@network-research-group.org

Abstract

Data visualisation over the Internet is a challenging task for web developers. North Sea plankton database is available to the scientists, researchers and students all over the world by using the WinCPR software. However, users need to download the WinCPR software to use the plankton database. The web-based plankton data visualisation research paper provides the various techniques to visualise the data from the North Sea plankton database without downloading WinCPR software. It also suggests using the different database system for online high data traffic. This paper proposes specific drawing methods for each chart type to visualise the data based on the Web-based CPR project which is developed using ASP.NET 2.0 and VB scripts. It also discusses the result of the different drawing methods and using different image formats. At last, it suggests the improvements for the current project and possible features for the future works.

Keywords

Data Visualisation, Image generation dynamically

1. Introduction

The web-based plankton data visualisation is based on the WinCPR software. The WinCPR software is a gridded database browser of North Sea Plankton database which is collected and maintained by the Sir Alister Hardy Foundation for Ocean Science (SAHFOS), Plymouth, UK. This database has kept recording monthly plankton abundance from the Continuous Plankton Recorder (CPR) survey. However, WinCPR software has the limitation of the database and it is the Windows-based software. Therefore, users need to download to use the software. The web-based CPR project is able to provide main functionalities of WinCPR software without downloading any software.

There are a number of processes to accomplish to produce an image dynamically on the Internet. Users have to make a query to get a result for their requirement. So, data collection from users is one part of data visualisation. Once users have filled the query, a connection to database and SQL command has to be done programmatically. After fetching the data from the database, the result data will be used to draw the appropriate graph or chart. The last step is to send the result graph or chart to the client browser. Entire task has to be accomplished dynamically. There are a lot of issues on each step. The foundation of this research paper is the web-based CPR project which is based on the WinCPR software. Three main parts can be classified to visualise the data on the fly in the project. They are Data Collection from the users, Database Connection, Image generation and Visualisation to the users.

2. Data Collection from the users

Each user has different level of understanding and familiarity to the database that they are trying to use. In the most case scenarios, users are assumed to make mistakes while they are completing the query. Thus, the developers have to make sure that the data filled by the users are accurate. In this Plankton Data Visualisation, users have no need to type to fill the query. Users have to choose the required data from the wizard on each step. Appropriate data will be displayed by using server controls on each step of wizard. It makes sure to be accurate the data by using the wizard server control. This Wizard server control <Asp:wizard> is supported by the ASP.NET 2.0. It builds a series of steps to display to the users. It has several properties to customize the appearance and to meet the requirements of developers. In visual studio 2005, the wizard control can be configured visually. Wizard steps can be added or removed visually in the developing time and programmatically at the run time. In the project, the steps of wizard display the data from the database. Database connection has been made depending on the choice of user on each step of wizard. After user has completed the wizard, the SQL command will be created.

3. Database Connection

In this project, although Microsoft SQL Server was intended to use in the first place, Microsoft Access database is used as the backend database. It is because the size of the database used in the project is small and it was not intended to use online as a professional web application. For online used as a professional web application, other relational database system such as Microsoft SQL server, Oracle and MySQL will be appropriate to use as a backend database because Microsoft Access is not designed to use high data traffic. In Microsoft article Q300216, it describes that when a file-sharing database is used in a multi-user environment, multiple client processes are using file read, write and locking operation on the same shared file across a network. If, for any reason, a process cannot be completed, the file can be left in an incomplete or a corrupted state. The other reason to use Microsoft Access in the project is that MS Access is portable and easy to use.

Entire project has used straight forward database connections. All of the SQL commands are created on the fly programmatically by using the user inputs. Moreover all of the command types are “Select” command type. There is no “Update” or “Delete” command to the original Continuous Plankton Recorder (CPR) database through out entire project. Therefore, underlying database has never been modified. If this project goes online and uses Microsoft SQL server or other database system, there is no need to implement to modify the database online. It will reduce several of security threats to the database system.

3.1 Microsoft Access Database Connection in .NET framework

The .NET framework provides the “System.Data.Oledb” name space to access an OLE DB data source. This name space has a number of classes to use to connect to the MS Access database. In the project, only three classes are used to connect to the database. They are “OledbConnection” class, “OledbCommand” class and

"OledbReader" class. When user has finished the wizard and click to view the graph or map, all the data user has chosen will be sent to the image generate page via the URL. All the data will be retrieved from the URL on the image generate page by using "Request.Params ()" method and put them into local variables. Then a database connection object will be created by using "OledbConnection" constructor which takes a connection string as a parameter. The connection to the database will be opened by using "Open ()" method of "OledbConnection" object. The variables which are generated from the URL will be used to create a SQL command. Then, an "OledbCommand" object will be created using the SQL command and the "OledbConnection" object. After that, "OledbReader" object will be created by using the "ExecuteReader ()" method of the "OledbCommand" object. That method will return the rows of the result of the SQL command. The values of rows will be put into the appropriate variables and it will be used to draw the image.

3.2 SQL Server Database Connection in .NET framework

To use SQL server in the future works, "System.Data.SqlClient" and "System.Data" name space are required to import. The "System.Data.SqlClient" name space provides access to SQL server database and the "System.Data" name space consists of most of the classes that constitute ADO.NET architecture (source: MSDN). "SqlDataAdapter" class will connect to the database and fill the dataset with data from the specific SQL command. "SelectCommand ()" method of "SqlDataAdapter" object will send a query to the database along with a connection object. In this case, the connection object will be created using "SqlConnection" object. "Dataset" object will then be filled by using "Fill ()" method of "SqlDataAdapter" object. "DataSet" object will contain "DataTable" which consists of "DataColumnCollection". The required data will be cached into the dataset and the necessary processes to draw images can be accomplished by using it.

4. Image Generation and Visualisation to users

The main subject of this research paper is generating the image on the fly and sending it to the users. It can be divided into two parts. The first one is drawing the image and second one is sending the image to the users. Both parts have different techniques and approaches.

The first part is to generate the image from the result of the query that user made. In ASP.NET 2.0, Graphics Device Interface (GDI+) supports the powerful graphics interfaces which allow developers to generate graphics and image handling code. The GDI+ interacts with graphical display devices such as a screen or printer instead of developers. Therefore, the developers have no concerned to the particular devices. "System.Drawing" namespace provides basic drawing functionality and "System.Drawing.Imaging" namespace and "System.Drawing.Drawing2D" namespace support advanced drawing and imaging functionality. The "System.Drawing" namespace is used in this project to draw the image using its classes such as Bitmap, Color, Pen, Font, Graphics and Image classes and "System.Drawing.Imaging" namespace is used to set the image file types.

A Bitmap object is used as a canvas to draw the images and graphs. It can be created with specific width and height and also from a bmp image. When the Bitmap object has been created, a Graphics object is created by using the Bitmap object. After the Graphics object has been created, all the drawing can be started on the Bitmap by using Graphics object. The Graphics object supports all the functions to draw rectangles, circles, lines, strings and images. After drawing required elements on the Bitmap object, it has the function to save all the drawings. The save () method of Bitmap object takes 2 parameters. The first one is the location to save and the second one is the image type. The location can be the file system as well as memory stream buffer. The image can be saved in the file system by providing the specific file path and name and the image can be saved in output stream buffer to send it to the client browser. The image format can be set using ImageFormat class. This class provides several image formats including GIF, JPEG, BMP and PNG format.

To send the result image to the client browser, the content of the page must be set to image. The “Response.Content” has to be set to “image/gif” or “image/Jpeg” to let the client browser know that the sending page is an image file. “Response.BufferOutput” is set to “True” to process all the functions in the page before the image is sent to the client browser. Then, “Response.Flush” method will send the image stored in output stream buffer. The general idea and codes of creating drawing place and sending it to the client browser are described in listing (1).

```
Dim objBitmap As New Bitmap(800,1000)
Dim objGraphic As Graphics = Graphics.FromImage(objBitmap)

“Drawing Process Here”

Response.Clear()
Response.BufferOutput = True
Response.ContentType = "image/jpeg"
objBitmap.Save(Response.OutputStream, ImageFormat.Jpeg)

objGraphic.Dispose()
objBitmap.Dispose()
Response.Flush()
```

Listing (1)

Even though original WinCPR software provides 8 different chart types, there are three different chart types to visualise the data in the project. According to the survey which is carried out for this project, the selected three chart types are the most useful chart types for the users. Those chart types are Seasonality Graph, Annual Plankton Abundance and Monthly Plankton Abundance. The survey was carried out in summer 2006. 12 Marine Biologists, 4 postgraduate students, 2 undergraduate students, 1 teacher and 5 people from other area are participated in the survey. The survey was organised with 15 questions. There is a question in the survey to rate the usage of the chart types in WinCPR. Half of the participants rates very useful for Monthly Plankton Abundance chart type and Annual Plankton Abundance. Seasonality Graph has been voted as very useful by (37.5%) of participants. According to those results, those 3 chart types are included in the project. Moreover, those three chart types have different drawing approaches.

Seasonality Graph is a graph presenting the selected species data of specific location in the North Sea of a selected Year. The user has to select the species name, a year range between 1948 and 1997 and a pixel location. The pixel locations on the North Sea are shown in figure 1. Based on the user's input, the required information is retrieved from the database to draw the graph. 12 decimal values are retrieved from the database to draw the seasonality graph. Figure 2 displays a sample of seasonality graph produced by web-based CPR. First, create a Bitmap object with specific width and height and create a Graphics object using that Bitmap object. Then the bars will be drawn with same width and different height according to the result values. And the legend will be drawn and save the image to the Output stream buffer. That buffer then will be sent to the client browser.

Monthly Plankton Abundance is a chart presenting the density of specific species on the North Sea of a selected year. The user has to choose species name and a year. 2 approaches are provided to draw this chart type. First approach is to draw the image of each month separately and put it together in a table and show it to the client browser. The Bitmap object will be created with North Sea map bitmap file. Then the pixel location boxes on the map will be filled with specific colours according to the value of that location. The image will be saved into the memory output stream and send it to the browser. This process will repeat for each image for 12 months.

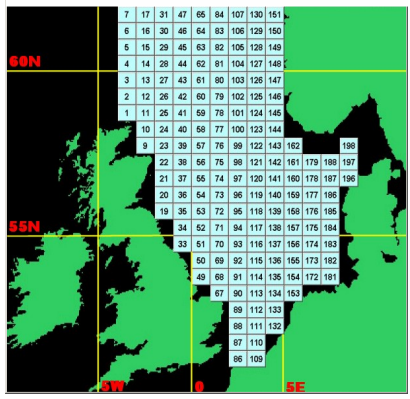

Figure 12: Pixel Locations on the North Sea Map

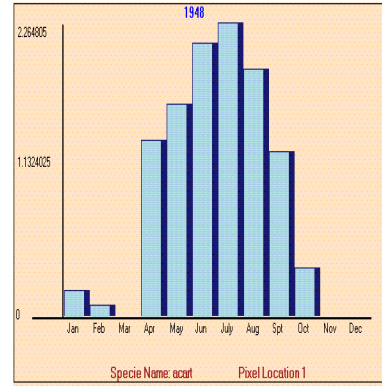

Figure 13: A sample Seasonality Graph

The second approach is to draw each month on a single image and send it to the client browser. This chart type uses the North Sea map as the background. Figures 3 and 4 show the sample of Monthly Plankton Abundance chart type produced by both methods. In this approach, a Bitmap object array which holds 12 elements is created. The image for each month of the year is drawn and put into each elements of the array. Then, a big Bitmap object and a Graphics object are created. The DrawImage () method of the Graphics object draws all the images in the array into the big Bitmap object. The big Bitmap object is saved in the output stream buffer as an image and sends it to the client browser. The listing (2) shows the brief demonstration of all the process.

```
Dim objBitmap(12) As Bitmap
Put all the images for each month in the array
Dim bigBitmap As New Bitmap(1200, 1600)
Dim bigGraphic As Graphics = Graphics.FromImage(bigBitmap)

        For a = 0 To 11
         bigGraphic.DrawImage(objBitmap(a), xpos1, ypos1, 350, 350)
            xpos1 += 400
            If a = 2 Or a = 5 Or a = 8 Then
                ypos1 += 400
                xpos1 = 0
            End If
        Next

        Response.Clear()
        Response.BufferOutput = True
        Response.ContentType = "image/gif"
        bigBitmap.Save(Response.OutputStream, ImageFormat.Gif)
        bigGraphic.Dispose()
        bigBitmap.Dispose()
        Response.Flush()
```

Listing (2)

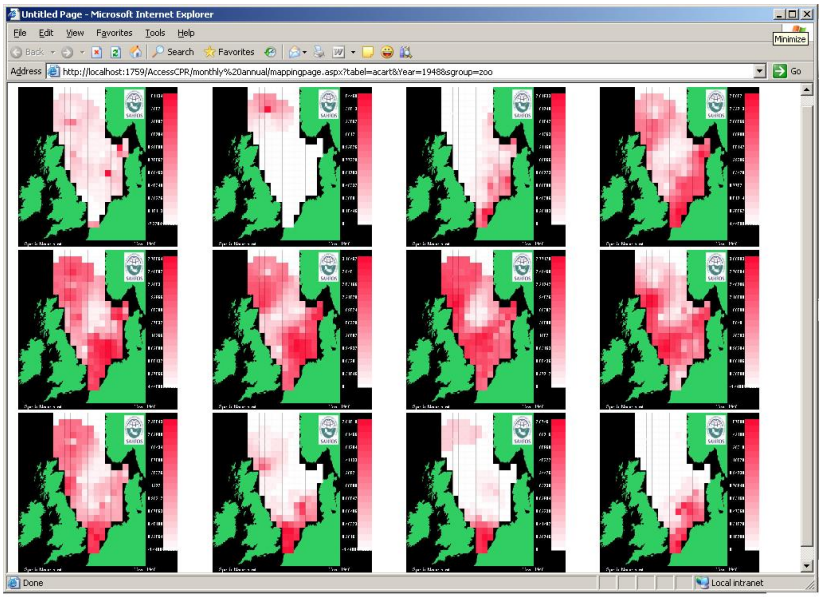

Figure 14: Monthly Plankton Abundance produced by first method

Annual Plankton Abundance is a chart presenting the density of selected species on the North Sea of a selected year. The user has to choose species name and a year to visualise the data. This chart type also uses the North Sea map as the background and fills the colour boxes of each location on the map. Figure 5 shows a sample of Annual Plankton Abundance chart type. In this chart type, a Bitmap object is created with North Sea map as the background and a Graphics object is created by using it. The pixel boxes are filled with colours according to the density of species that user selected. Then the image will be saved into the output buffer and send it to the client.

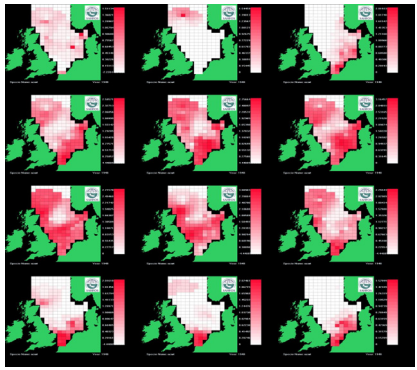

Figure 15: Monthly Plankton Abundance produced by second method

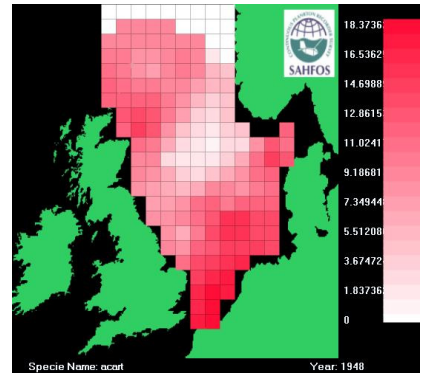

Figure 16: A sample of Annual Plankton Abundance chart type

5. Discussion

5.1 Image Quality and Image Size

The size of the image is critical in the data visualisation techniques. The image transfer time over the Internet depends on the size of the image. The smaller image size increases the speed of the transfer time. The image size produced by the project is between 10 kb and 300kb. The Seasonality graph has the size between 10kb and 30kb, Annual Plankton Abundance chart type has the size between 35kb and 40kb. Monthly Plankton Abundance chart type has the image size between 250kb and 300kb with second method. First method will send 12 images to the browser at the same time and each image has the size between 35kb and 40kb. Therefore, the first method increases the amount the data sending to the client browser. Two image formats are available to produce in the project. JPEG and GIF image formats are widely used over the Internet. JPEG format has larger size than GIF format but it has better quality.

5.2 Image Producing Techniques

The image can be produced in two ways. The first one is to produce the image on the server side and the second one is to produce on the client side. The server-side scripts are used to produce the image on the server and this technique is used in the project. Producing the image on the server increases the speed of image generation process rather than producing the image on the client side. The result image only is sent to the client browser. In this technique, all the imaging and drawing functions are done on the server. Therefore, all the requirements to draw the image are supported by the server. However, if the multiple users are generating the image on the server at the same, the performance of server will be decreased.

The second approach is to produce the image on the client side using client-side scripts. In this approach, server will send the client-side scripts to the client browser along with necessary data to draw the image. This approach will reduce the numbers of processes to generate the image on the server. Therefore, it will increase the

performance of server. However, the client machine has to permit to run the scripts on it. Most of the client-side scripts such as Java Applet need permission from the browser to run. The developers need to be aware of the target audience environment to use the specific client-script. Office and educational environment are not permitted to run the scripts on their system for security reasons. One of the security threats for client-side scripts running on the machine is that the client-side scripts can access to the resources of the machine. Moreover, the target audience for Web-based CPR is office environment and educational environment. Under those circumstances, the server-side scripts are more appropriate to use in the project than the client-side scripts.

5.3 Image Saving Methods

According to the survey result, (54.1 %) of the participants wanted to save the images on the server. However, there is no save function in the project. There are three possible approaches have been suggested to save the images on the server. Each approach has advantages and disadvantages.

First approach is to save the image as an image file on the server. If user chooses to save the image, the image name and location will be specified programmatically. When user retrieves the image next time, the image will be fetched from the server's file system and display it to the user. In this approach, server will need to provide the significant amount of space to store the images. However, when user retrieves the image, there will be no process to produce the image again. The server has to send the image directly to the client browser.

The second approach is to save the Bitmap object into a file. The Bitmap object will have all the drawings in it before it is sent to the file. When the user retrieves the image next time, user will have the chance to choose the image file type. This approach will reduce the size of file to save on the server. Nevertheless, it will have a small process to change it to the image file and send it to the browser.

The third approach is to save the result of user's query into the file along with a flag which can describe the chart type of the image. When the user retrieves the image, the flag will be checked to choose the chart type and the other data will be used to generate the image. This approach will have the smallest amount of data to save in the file system but it will also have the whole process to draw the image.

6. Conclusion

This study has accomplished to visualise the plankton data over the Internet without downloading any software. Moreover, this study has used only server-side scripts to produce the images on the fly. The advantages and disadvantages of using server-side and client-side scripts are highlighted. However, client-side scripts can achieve some of the features which server-side scripts are not able to achieve. For instance, the client-side scripts can save the images on the client machine by the web application itself. The density of plankton data on North Sea is shown by the colours in the project. However, producing the images with the client-side scripts can also let

user to change the colours and image file types on real time. This research paper also advices three approaches to save the images on the server. Nevertheless, this study is the basic step to visualise the plankton data over the Internet and it gives a range of ideas to produce the images dynamically.

7. Reference:

Cogan, A., (2004), "Moving Your Access 2002 Database to SQL Server" Microsoft Regional Director, Australia, December, white paper

Esposito. D., (2004a), "Cutting Edge Image generation Service for asp.net 1.1" MSDN magazine, April.

Esposito, D., (2004b), "Image generation service for ASP.NET 1.1" http://msdn.microsoft.com/msdnmag/issues/04/04/CuttingEdge/ (accessed 15 August 2006)

Latarre, U., (1998), "Graphic file formats", http://www.why-not.com/articles/formats.htm#INDEX (accessed 20 August 2006)

Microsoft Developer Network, "System.Data name space", http://msdn.microsoft.com/library/default.asp?url=/library/en-us/cpref/html/frlrfsystemdataoledboledbconnectionclasstopic.asp (accessed 30 august 2006)

Microsoft Developer Network, "System.Data.SqlClient name space", http://msdn.microsoft.com/library/ default.asp?url=/library/en-us/cpref/html/frlrfSystemDataOleDb.asp (accessed 30 august 2006)

Microsoft Developer Network ,"System.Data.Oledb name space", http://msdn.microsoft.com /library/default.asp? url=/library/en-us/cpref/html/frlrfsystemdataoledboledbcommandclass topic.asp (accessed 30 august 2006)