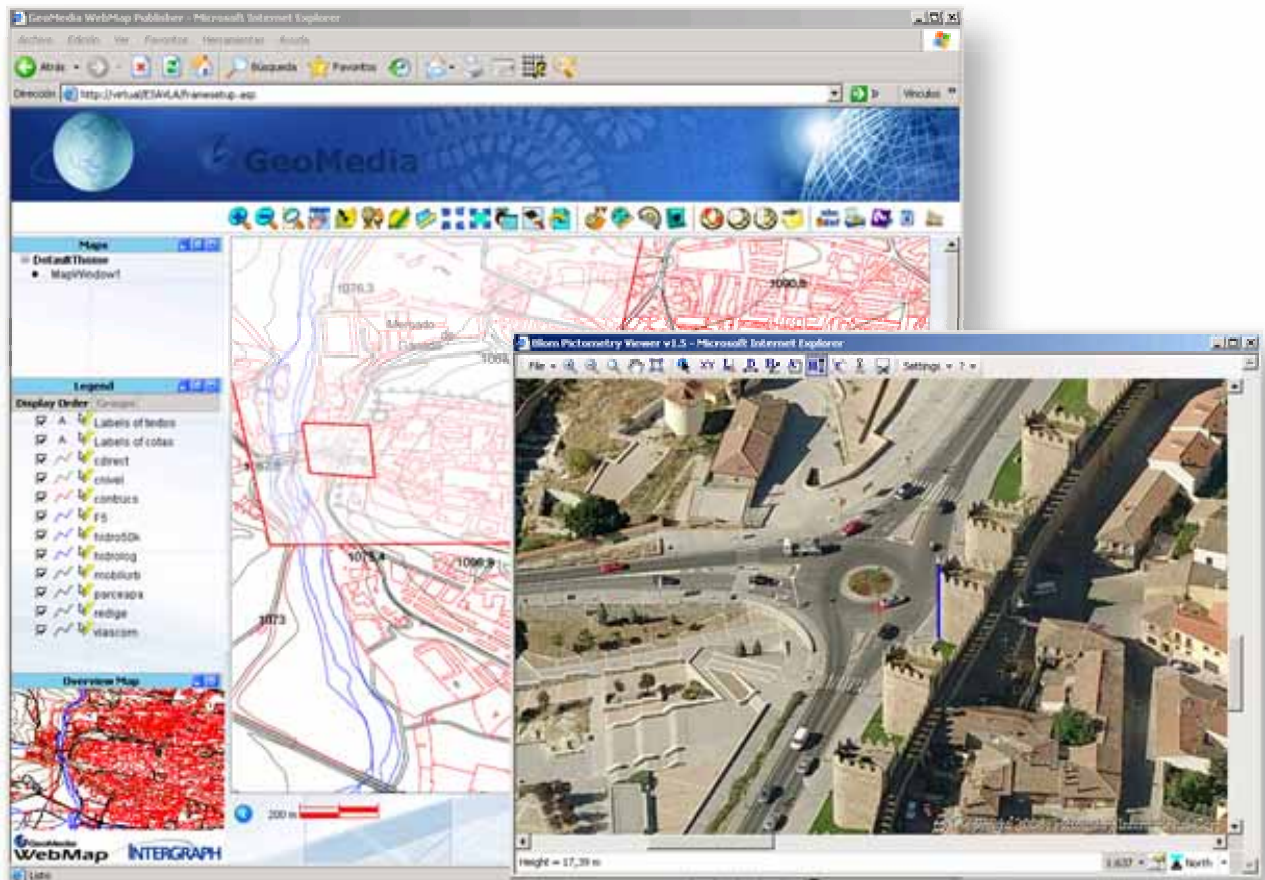




BLOM ASA

Integrating Pictometry in Geomedia WebMap Developer Manual v1.0



Index

1	INTRODUCTION	3
2	SYSTEM REQUIREMENTS	4
2.1	SOFTWARE REQUIREMENTS IN THE SERVER	4
2.2	SOFTWARE REQUIREMENTS IN THE CLIENTS	4
2.3	SAMPLE DATA.....	4
3	INSTALLATION	5
3.1	CREATE A WEB APPLICATION WITH GEOMEDIA WEBMAP PUBLISHER	5
3.2	SET UP A NEW COMMAND IN THE WEB APPLICATION	12
3.3	CHANGE SAMPLE CODE TO SELECT THE PROJECTION AND IMAGE WAREHOUSES	13
3.4	TEST THE WEB APPLICATION	14
4	CODE DESCRIPTION.....	18

1 Introduction

This document explains how to use Blom Pictometry Viewer SDK v1.5 to easily integrate oblique imagery functionality in web applications developed using Intergraph Geomedia™ WebMap.

Blom Pictometry Viewer v1.5 is a SDK, a reusable component that can be invoked in DDE / ActiveX / .NET frameworks. This manual is provided with code samples showing how to integrate Blom Pictometry Viewer in Geomedia WebMap. The SDK has been built with ease of development in mind and contains only one component, the BlomPictometryViewer control, that conforms a full oblique imagery viewer and analyzer.

2 System requirements

In order to run the samples provided here it is needed first to check if all the requirements indicated below are fulfilled.

2.1 Software requirements in the server

- Geomeia WebMap v6.1 is supported. Older versions may work with code sample provided here, but could require minor changes. It is needed to ensure that a Geomeia WebMap service is up and running, correctly configured. For this read carefully the *Geomeia WebMap Documentation*. To install Geomeia WebMap a requirement is Microsoft Internet Information Sever (IIS) or some other web server software.
- Geomeia WebMap Publisher v6.1. This should be installed and registered. For this read carefully the *Geomeia WebMap Documentation*. To install Geomeia WebMap Publisher is a requirement to have Geomeia Pro installed.
- Pictometry Network Image Warehouse (NIW) must be installed. This software from Pictometry allow publishing Pictometry Image Warehouses through intranet or Internet, although it is recommended intranet-use only or Internet-use but through a high-dedicated bandwidth. It is needed to ensure that a NIW service is up and running, correctly configured. For this read carefully the *NIW Installation and Administration Guide*. To install NIW is needed:
 - Microsoft Internet Information Server (IIS)
 - .NET Framework 1.1
 - NIW license file

2.2 Software requirements in the clients

- Blom Pictometry Viewer SDK v1.5 must be installed in every client machine. For this read carefully the *Blom Pictometry Viewer v1.5 installation guide*. To install Blom Pictometry Viewer v1.5 is needed:
 - .NET Framework 2.0
 - Pictometry ActiveX license file
- Internet Explorer 6 and above. Other web browsers are not supported.

2.3 Sample data

In order to test the web development two different data is needed:

- A Geomeia warehouse in any format supported by Geomeia.
- A Pictometry warehouse with the imagery. Contact with Blom ASA to obtain Pictometry warehouse sample data. Warehouses are folders containing Pictometry imagery in a structured way. Each warehouse has a nodestring code that allows the warehouse to work with certain Pictometry license. It is possible connect to one or several warehouses inside Blom Pictometry Viewer. This is performed through the Settings dialog. Read the *Blom Pictometry Viewer 1.5 User Manual* for more information on this.

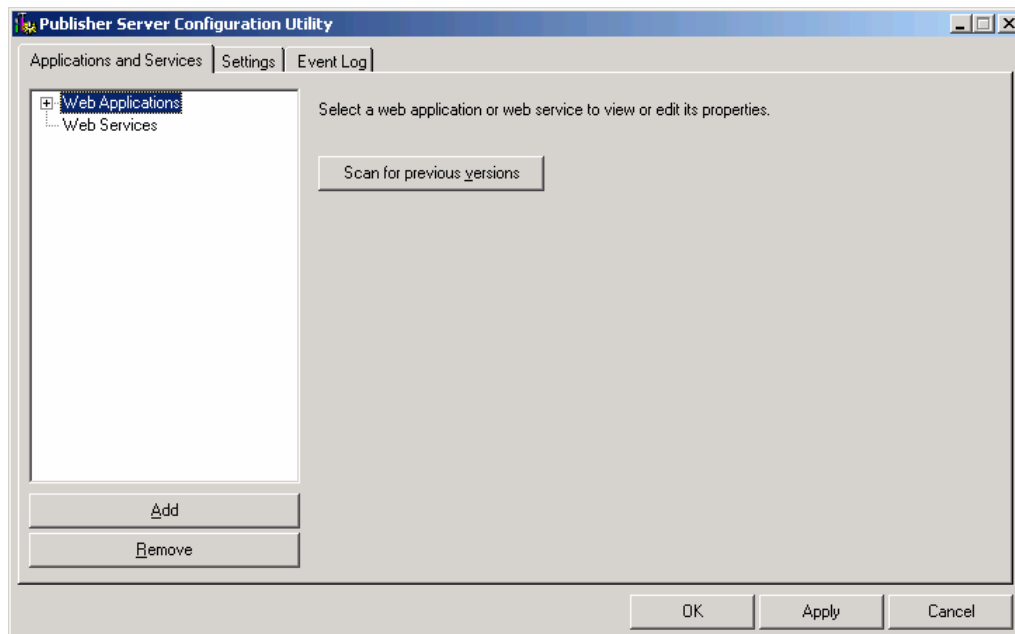
3 Installation

Sample code provided is an extension to the web application created using the Geomeia WebMap Publisher product. This utility is a Geomeia Pro plug-in that eases the creation of a full mapping webpage from a Geomeia Pro workspace. To know more about this read the *Geomeia WebMap Publisher Help*.

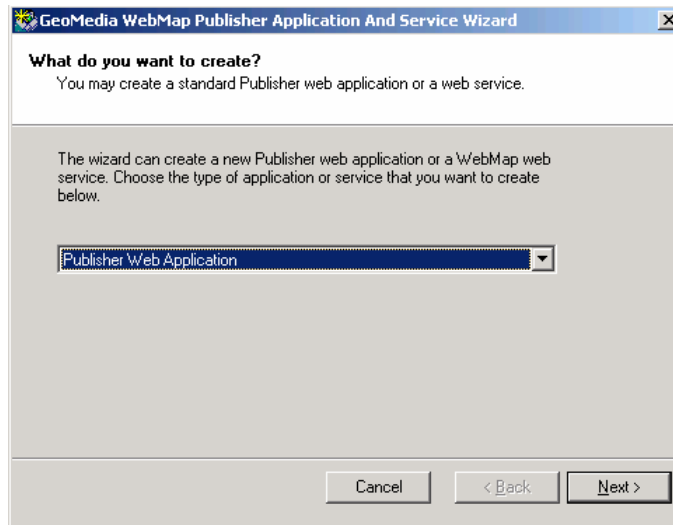
To install the sample code follow this steps:

3.1 Create a web application with Geomeia WebMap Publisher

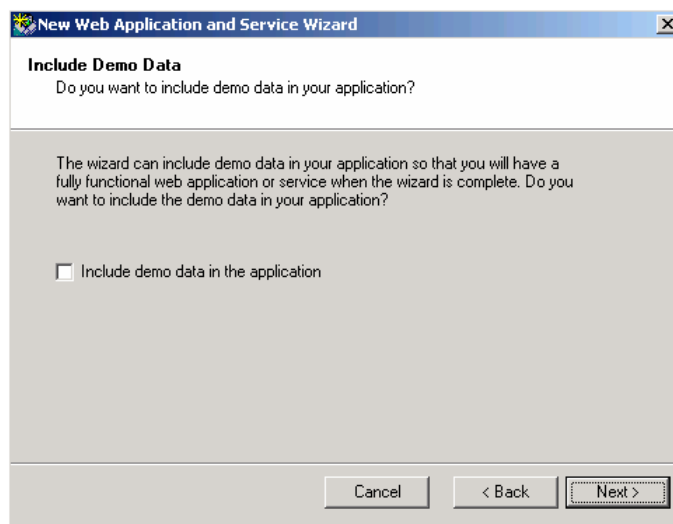
1. Ensure that all the requirements are installed in the server machine and in the clients.
2. Open the Geomeia Server Configuration Utility selecting Start > All programs > Geomeia WebMap Professional > Publisher > Server Configuration Utility.



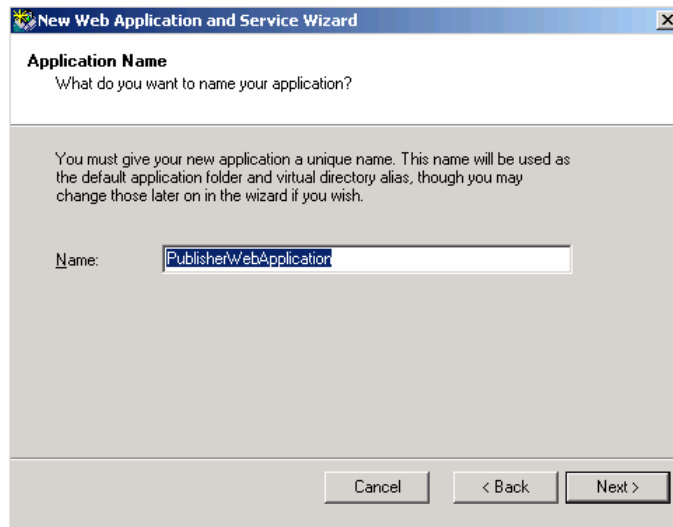
3. In the “Applications and Services” tab click in Add. A wizard is opened. In the first step of the wizard select Publisher Web Application in the dropdown box.



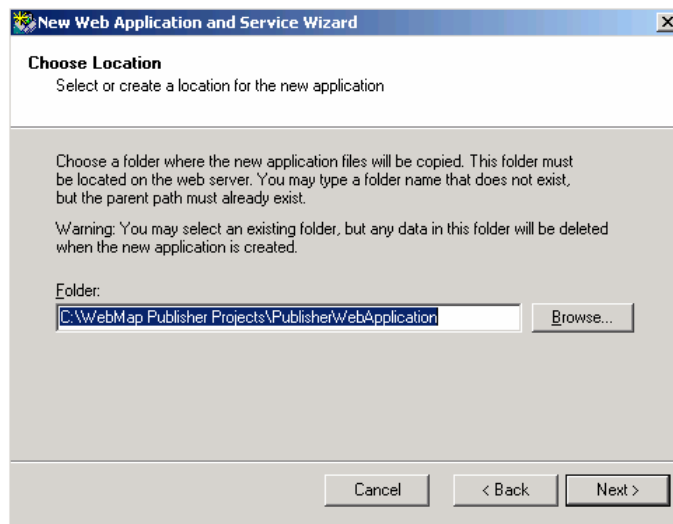
4. In the next step of the wizard leave the checkbox unchecked.



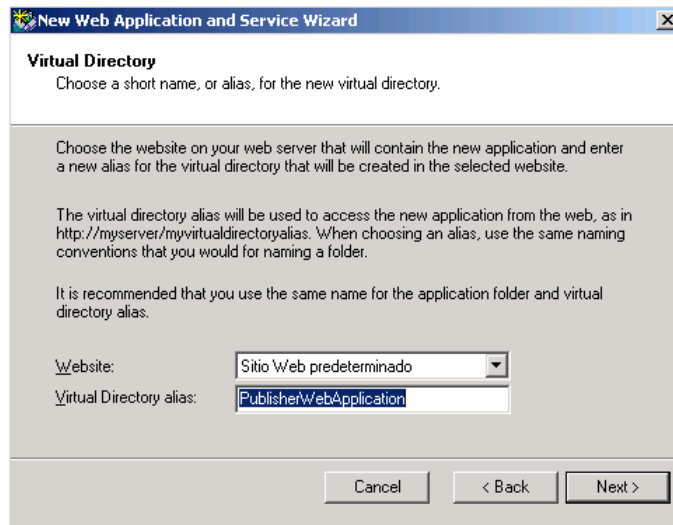
5. In the next step of the wizard type a short name for the web application.



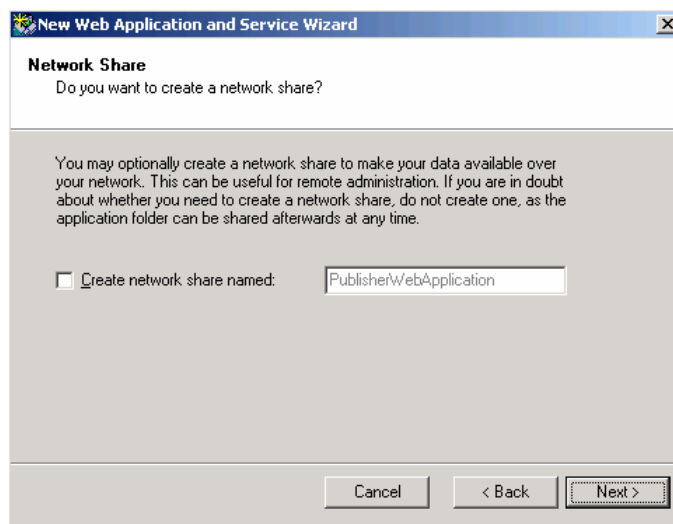
6. In the next step of the wizard type the correct path for the folder where to store the web application. It is recommended to leave the default.



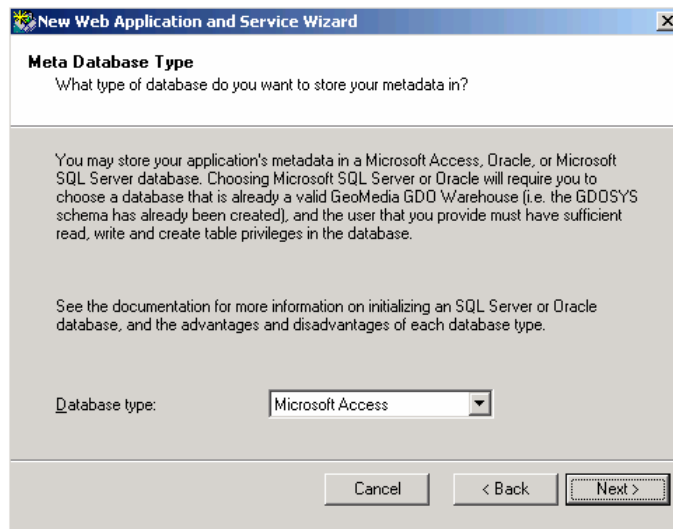
7. In the next step of the wizard leave the default values.



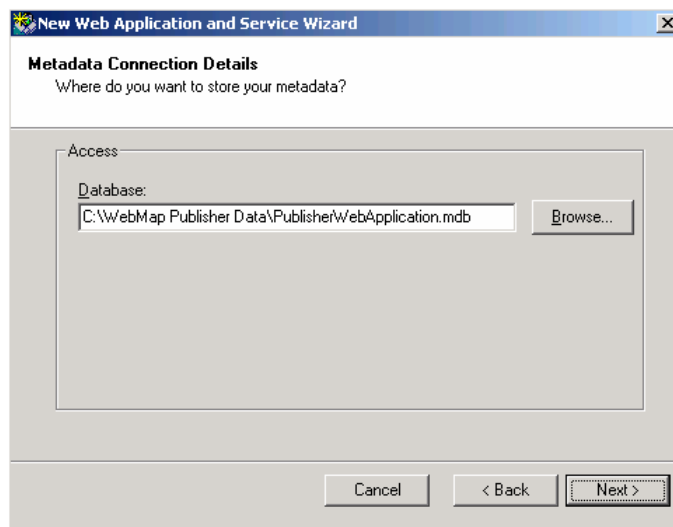
8. In the next step of the wizard leave the checkbox unchecked.



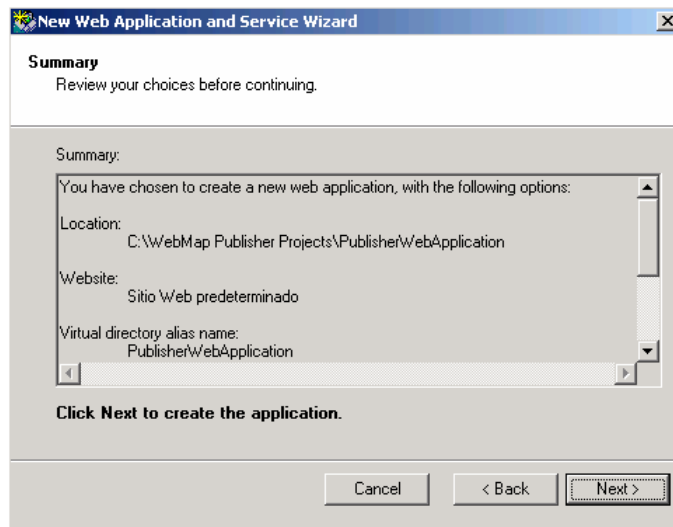
9. In the next step of the wizard select your preferred metadata database engine. It is recommended, for simplicity, to choose Microsoft Access.



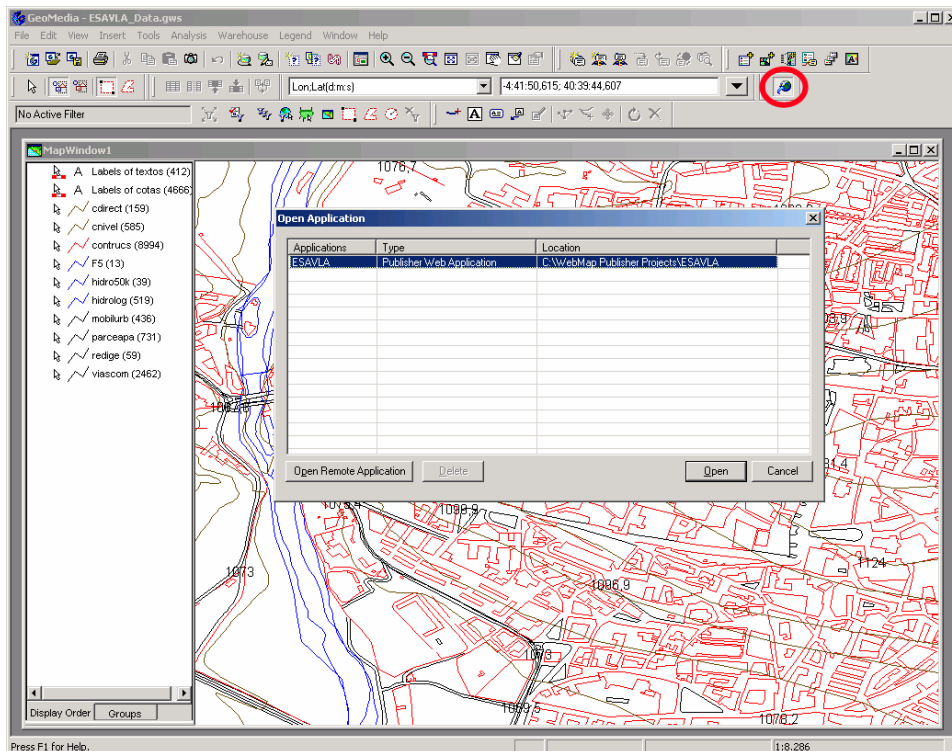
10. In the next step of the wizard type the correct path for the Microsoft Access database (if this was the database type). It is recommended to leave the default path.



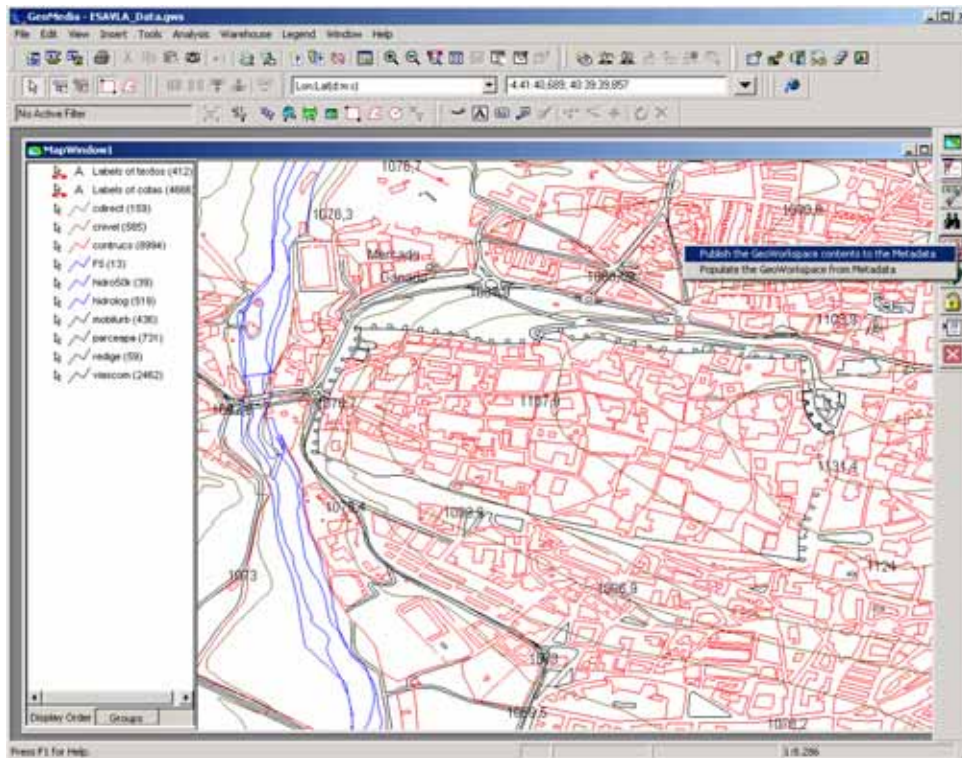
11. The last step is a summary. Click Next to execute the wizard with the options selected.



12. A full web application is created. The web application uses a database to store the initial map settings. To populate this metadata database close the Geomedia Server Configuration and open an instance of Geomedia Professional.
13. Create a map and add some layers to it. Set up colors and scale ranges in the map. Set up the projection of the map using the Geomedia tools. The projection selected for the map must match one projection in the existing list in the “epsg” file. This file is stored in the installation folder of Blom Pictometry Viewer v1.5. The file is a text file and can be opened in any text editor. Below is an explanation of this file and how to use it to select the correct projection for the web application.
14. Once the map is set up, click the “Geomedia WebMap Publisher Administrator” button in the Geomedia toolbar to open the application selector. Select the previously created web application in the list and click Open. A new toolbar will open.

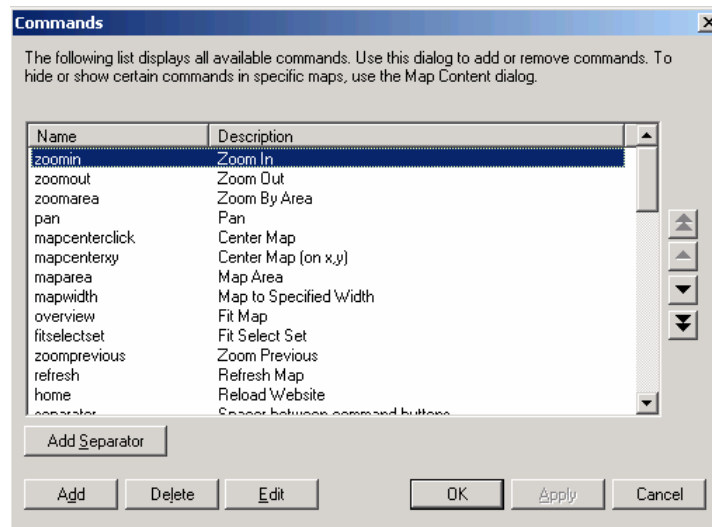


15. Click in the “Publish and Populate GeoWorkspace” button, and then in the “Publish the GeoWorkspace contents to the Metadata” button. This populates the metadata database with the map settings. Do not close Geomedia and proceed to next steps.

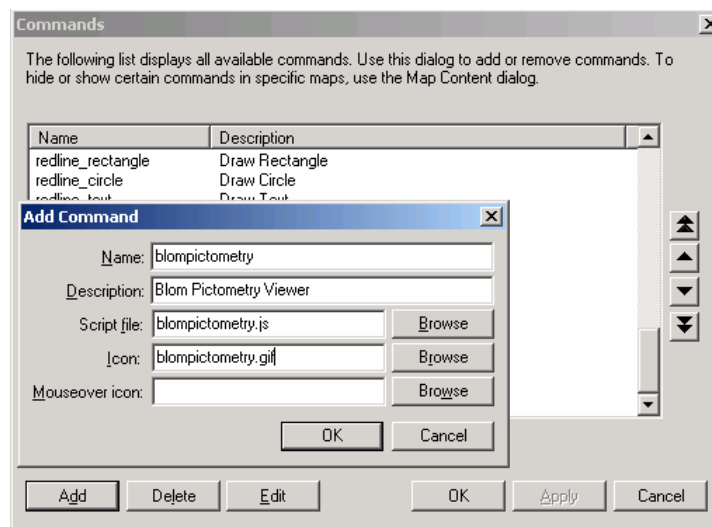


3.2 Set up a new command in the web application

16. Unzip the BlomPictometryGWM.zip to any folder location in the server machine.
17. Copy the “blompictometry” folder unzipped to the folder that contains commands in the web application. If used the default values in previous wizard, the folder path is “C:\WebMap Publisher Projects\\commands”. Now a new “C:\WebMap Publisher Projects\\commands\blompictometry” folder must exist.
18. In Geomedia Professional click in the “Commands” button. The Commands dialog opens.



19. Select the last command available. Click in Add. The “Add command” dialog opens. Type blompictometry as name, “Blom Pictometry Viewer” as description, blompictometry.js as script file and finally blompictometry.gif as icon. Do not change these values. Then click OK in the “Add command” dialog and click OK in Commands dialog to create the new command and link it with the files previously copied. Finally exit Geomedia.



3.3 Change sample code to select the projection and image warehouses

20. Open the “blompictometry.js” file that is inside the “C:\WebMap Publisher Projects\\commands\blompictometry” folder with any text editor. Read the first commented lines to know how to change projection settings. For example, to change projection to British OSGB change this code:

```
var __blompictometry_settings = "LinkEPSG=4326\r\n" +  
  "LinkEPSGName=WGS 84\r\n" +  
  "LinkEPSGParams = +proj=longlat +ellps=WGS84 +datum=WGS84 +no_defs\r\n" +  
  "Warehouse=http://virtual/NIW/Avila\r\n" +  
  "SelectedWarehouses= http://virtual/NIW/Avila " ;
```

21. to this code:

```
var __blompictometry_settings = "LinkEPSG=27700\r\n" +  
  "LinkEPSGName=OSGB 1936 / British National Grid\r\n" +  
  "LinkEPSGParams = +proj=tmmerc +lat_0=49 +lon_0=-2 +k=0.999601 +x_0=400000 +y_0=-  
100000 +a=6377563,39600000 +b=6356256,90923729 +towgs84=446.448,-  
125.157,542.060,0.1502,0.2470,0.8421,-20.4894 +units=m +no_defs no_defs\r\n" +  
  "Warehouse=http://virtual/NIW/Avila\r\n" +  
  "SelectedWarehouses= http://virtual/NIW/Avila " ;
```

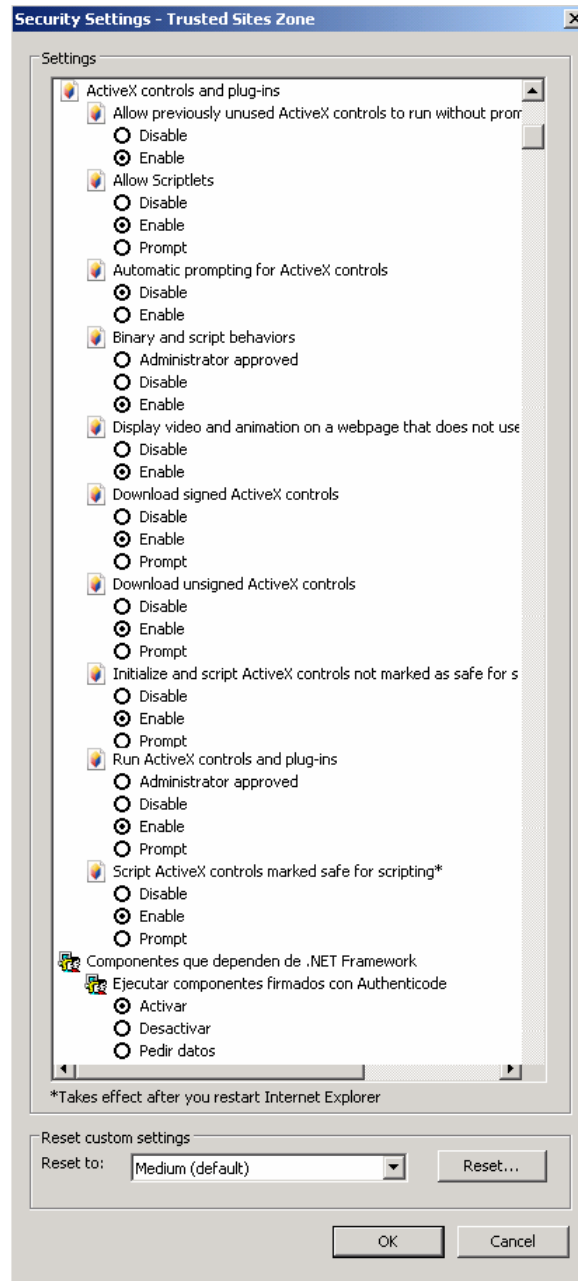
22. In the lines of the code indicated above change the lines with the parameters Warehouse and SelectedWarehouses to point to the correct warehouses in the server or in the local network. It is possible to point to several image warehouses separating them with semi-colon. To know more about changing Blom Pictometry Viewer settings programmatically refer to the *Blom Pictometry Viewer 1.5 Developer Manual*.

3.4 Test the web application


23. Ensure the client computer has the requirements specified above.
24. Follow the installation steps specified above.
25. Open the Internet Explorer options dialog. Select the Security tab. Select the Trusted sites (if testing a remote site) or the Local intranet (if testing a local page) zone category from the list and then click on Sites button to open the “Trusted sites” dialog. Here input the URL address of the web site created and click on Add to add to the list. Click Close.



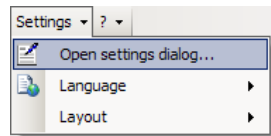
26. In the options dialog click on Custom level button to open the Security Settings dialog. Here enable the options as shown in the next picture. Click OK in every dialog and close Internet Explorer.



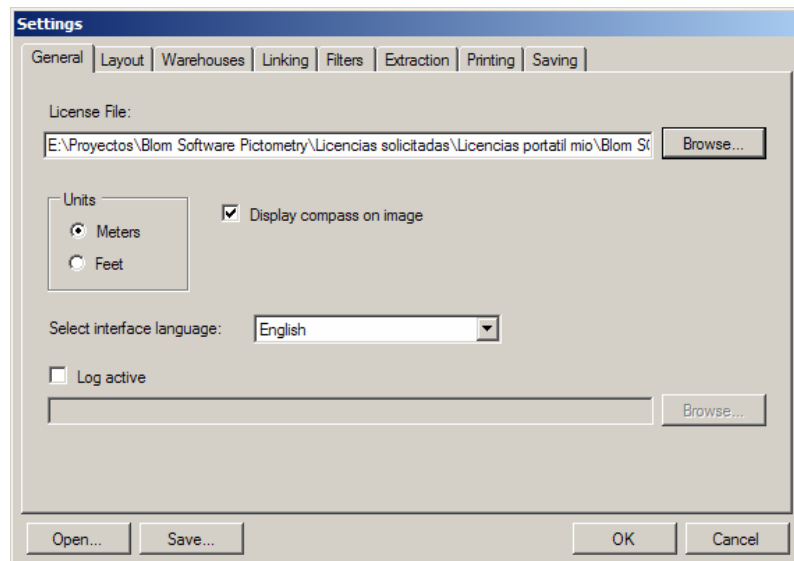
27. Open a Internet Explorer window and type URL `http://<server name>/<web app name>/`, where `<server name>` is the name of the server machine (or could be localhost if Internet Explorer is executing in the server), and `<web app name>` is the name given to the Geomedia web application, typed in the step 3 of the "Geomedia WebMap Publisher Application and Service Wizard". Press Enter.
28. Wait for moment until the map refreshes. Then click in the last command in the toolbar,

Blom Pictometry Viewer .

29. This opens the Blom Pictometry Viewer window. Click in the ActiveX control to activate it.
30. Select the menu Settings > Open settings dialog to open the Settings dialog.




31. In the dialog select the General tab. Click Browse to search for the license PLF file obtained from Blom. This license file is needed to execute the application and allow the access to the imagery. Click in the “Warehouses” and “Linking” tabs and ensure that the ActiveX have the settings modified by code. Click OK to save settings.

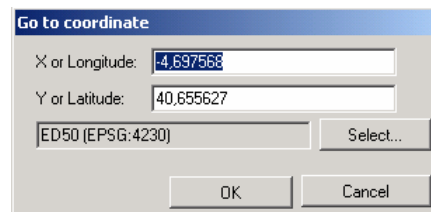


32. To open an image using de the map, click in the Geomedia map in any position and wait several seconds. Ensure that the current tool in the web application is the Blom



Pictometry Viewer tool. If not click again in the tool prior to click in the map. Blom Pictometry Viewer will load the image or images that cover the point selected in the map. Also the Geomedia map will show the extents (total and visible) of the images opened.

33. To open an image using a fixed coordinates, select the “Open a image in a coordinates” button  . The Go to coordinate dialog is opened.



34. Introduce a correct longitude or X and latitude or Y coordinates. Indicate the projection for this coordinates clicking in Select. By default, the last projection selected in the Settings dialog is offered here, but you can change it, so it is possible to ask for coordinates in any of the supported projections.

35. Click OK. The Blom Pictometry Viewer will open the best image for the point selected. If the layout selected includes several views, all of them will be opened with the best images that contains the point selected.
36. For a complete reference using Blom Pictometry Viewer, see the *Blom Pictometry Viewer v1.5 User Manual*.

4 Code description

The sample code provided shows how to integrate the Blom Pictometry Viewer v1.5 in an extension to the web application created using the Geomedia WebMap Publisher product.

Requirements to customize the code:

1. A configured Geomedia WebMap Publisher Web Application in the server.
2. .NET Framework 2.0 in the client
3. Blom Pictometry Viewer SDK 1.5 in the client (includes Pictometry SDK 2.7)
4. Local sample warehouse/es or NIW accesible server
5. Pictometry license
6. HTML and JavaScript editor (can be Notepad but a better one is recommended)

Files provided:

blompictometry.gif	Blom Pictometry Viewer icon
blompictometry.js	JavaScript code of the command
Viewer.htm	HTML webpage with the Viewer
Helpfile\blompictometry.htm	Web help of Blom Pictometry Viewer
Helpfile*.gif	Web help pictures

The code:

Viewer.htm

```

<html style="margin:0 0 0 0">
<head>
<title>Blom Pictometry Viewer v1.5</title>
<script type="text/jscript">
    function GoToCoord(x, y) {
        Object1.GoToCoord(x, y);
    }

    function TotalExtentChanged(ulx, uly, urx, ury, lrx, lry, llx, lly, ViewIndex) {
        window.opener.__gblompictometry_TotalExtentChanged(ulx, uly, urx, ury, lrx, lry,
llx, lly, ViewIndex);
    }

    function VisibleExtentChanged(ulx, uly, urx, ury, lrx, lry, llx, lly, ViewIndex) {
        window.opener.__gblompictometry_VisibleExtentChanged(ulx, uly, urx, ury, lrx,
lry, llx, lly, ViewIndex);
    }
    function LoadSettingsString(s) {
        Object1.LoadSettingsString(s);
    }
    function LayoutChanged() {
        window.opener.__gblompictometry_LayoutChanged();
    }
    var globalGoToCoord = this.GoToCoord;
    var globalLoadSettingsString = this.LoadSettingsString;
</script>
<script type="text/vbscript">
    sub Object1_TotalExtentChanged(ulx, uly, urx, ury, lrx, lry, llx, lly, ViewIndex,
Orient)
        TotalExtentChanged ulx, uly, urx, ury, lrx, lry, llx, lly, ViewIndex
    end sub

    sub Object1_VisibleExtentChanged(ulx, uly, urx, ury, lrx, lry, llx, lly, ViewIndex,
Orient)
        VisibleExtentChanged ulx, uly, urx, ury, lrx, lry, llx, lly, ViewIndex
    end sub

    sub Object1_LayoutChanged(layout)
        LayoutChanged
    end sub
</script>
</head>

```

```
<body>
  <object classid="clsid:35d4e4c7-910b-4df5-b52b-f8d8f7becea6" id="Object1" height="100%"
width="100%"></object>
</body>
</html>
```

Blompictometry.js

```
//=====
//
var __blompictometry_window = null;
var __blompictometry_centerX, __blompictometry_centerY;
var __blompictometry_InnerBounds = new Array (null, null, null, null, null);
var __blompictometry_OuterBounds = new Array (null, null, null, null, null);
var __blompictometry_styles = null;
// __blompictometry_settings: string with initial settings
// To change this variable read the Blom Pictometry Viewer 1.5 Developer Manual
// The sample here sets the projection to WGS84Lat/Lon and the warehouses to use to one
with code ESAVLA.
// For example, to change projection to British OSGB the next code should be like this:
// var __blompictometry_settings = "LinkEPSG=27700\r\n" +
// "LinkEPSGName=OSGB 1936 / British National
Grid\r\n" +
// "LinkEPSGParams = +proj=tmerc +lat_0=49
+lon_0=-2 +k=0.999601 +x_0=400000 +y_0=-100000 +a=6377563,39600000 +b=6356256,90923729
+tows84=446.448,-125.157,542.060,0.1502,0.2470,0.8421,-20.4894 +units=m +no_defs
no_defs\r\n" +
// "Warehouse=C:\ESAVLA06-093-LIB\ESAVLA06-
093-WHS\ESAVLA-093-WHS.PIW\r\n" +
// "SelectedWarehouses=C:\ESAVLA06-093-
LIB\ESAVLA06-093-WHS\ESAVLA-093-WHS.PIW";
// The values for LinkEPSG, LinkEPSGName & LinkEPSGParams are taken from the "epsg" file
in the Blom Pictometry Viewer 1.5 install dir.

var __blompictometry_settings = "LinkEPSG=4326\r\n" +
"LinkEPSGName=WGS 84\r\n" +
"LinkEPSGParams = +proj=longlat +ellps=WGS84 +datum=WGS84
+no_defs\r\n" +
"Warehouse=http://virtual/NIW/Avila\r\n" +
"SelectedWarehouses=http://virtual/NIW/Avila";

function blompictometry_onclick() {
  var name="blompictometry";
  __blompictometry_OpenViewer();
  __blompictometry_window.globalLoadSettingsString(__blompictometry_settings);
  if (WaitingFor("map")) {
    WaitAlertFor("map");
  } else {
    // set coordinate units to storage
    top.frames["fraMap"].setCoordinateUnits(1);
    top.frames.fraMap.getPoint(__blompictometry_callback, name);
  }
}

function __blompictometry_callback(sIOBuf) {
  var iTmp = sIOBuf.indexOf( ":" );

  if (iTmp==-1) {
    alert(sIOBuf);
  } else {
    var X = sIOBuf.substring( 0, iTmp)-0; //Get X
    var Y = sIOBuf.substring( iTmp+1, sIOBuf.length-1 )-0; //Get Y
    // add temp redline
    top.frames["fraMap"].addPoint(X, Y, false);

    __blompictometry_centerX = X;
    __blompictometry_centerY = Y;

    __blompictometry_window.globalGoToCoord(__blompictometry_centerX,
__blompictometry_centerY);
  }
}

function __blompictometry_OpenViewer() {
  if (!(__blompictometry_window && !__blompictometry_window.closed)) {
    var w = 780;
    var h = 545;
    var t = getWindowY(h, 2);
    var l = getWindowX(w, 2);
    var specs =
"screenX="+l+",screenY="+t+",left="+l+",top="+t+",width="+w+",height="+h+",menubar=no,toolbar
```

```

=no,status=no,resizable=yes,scrollbars=yes";
    __blompictometry_window = window.open (appGmwebappURL +
"commands/blompictometry/viewer.htm", "BlomPictometryWindow", specs);
    if (!__blompictometry_window.opener) {
        __blompictometry_window.opener = this.window;
    }
    __blompictometry_window.focus();
}

function __blompictometry_TotalExtentChanged(ulx, uly, urx, ury, lrx, lry, llx, lly,
ViewIndex) {
    var shape = ulx + ":" + uly + ";" + urx + ":" + ury + ";" + lrx + ":" + lry + ";" +
llx + ":" + lly + ";" + ulx + ":" + uly;
    __blompictometry_OuterBounds[ViewIndex] = shape;
    __blompictometry_RefreshMap();
}

function __blompictometry_VisibleExtentChanged(ulx, uly, urx, ury, lrx, lry, llx, lly,
ViewIndex) {
    var shape = ulx + ":" + uly + ";" + urx + ":" + ury + ";" + lrx + ":" + lry + ";" +
llx + ":" + lly + ";" + ulx + ":" + uly;
    __blompictometry_InnerBounds[ViewIndex] = shape;
    __blompictometry_RefreshMap();
}

function __gblompictometry_LayoutChanged() {
    top.frames.fraMap.deleteAllRedlines(true);
    __blompictometry_InnerBounds = new Array (null, null, null, null, null);
    __blompictometry_OuterBounds = new Array (null, null, null, null, null);
}

function __blompictometry_RefreshMap() {
    top.frames["fraMap"].setCoordinateUnits(1);
    __blompictometry_styles = top.frames.fraMap.getRLStyles();
    __blompictometry_styles[7] = "fill:white;fill-opacity:0.5;stroke:red;stroke-width:2";
    top.frames.fraMap.setRLStyles(__blompictometry_styles);
    top.frames.fraMap.deleteAllRedlines(true);
    for (var n = 0; n < 5; n++) {
        if (__blompictometry_OuterBounds[n] != null) {
            top.frames.fraMap.addPolygon(__blompictometry_OuterBounds[n], true);
        }
        if (__blompictometry_InnerBounds[n] != null)
            top.frames.fraMap.addPolygon(__blompictometry_InnerBounds[n], true);
    }
}

var __gblompictometry_TotalExtentChanged = this.__blompictometry_TotalExtentChanged;
var __gblompictometry_VisibleExtentChanged = this.__blompictometry_VisibleExtentChanged;
//=====================================================

```

Explanation:

1. The Blom Pictometry Viewer contains just one ActiveX control, called BlomPictometryViewer. Its ClassID is {35d4e4c7-910b-4df5-b52b-f8d8f7becea6}. We have used the <object> tag in the Viewer.htm web page to insert the control, indicating the ClassID in the classid attribute of the <object> tag. It is needed to give the control an instance name using the id attribute so later we can reference the instance in the code.
2. When user clicks in the “Blom Pictometry Viewer” button in the Geomeia web application the blompictometry_onclick function of blompictometry.js is called. In this function we open the viewer and initialize the Blom Pictometry Viewer control calling the LoadSettingsString control method. As we need to call the ActiveX, that is embedded in other browser window, we use the trick of a global variable (globalLoadSettingsString) that stores a pointer to the ActiveX method.

In blompictometry.js

```

var __blompictometry_settings = "LinkEPSG=4326\r\n" +
"LinkEPSGName=WGS 84\r\n" +
"LinkEPSGParams = +proj=longlat +ellps=WGS84 +datum=WGS84
+no_defs\r\n" +
"Warehouse=http://virtual/NIW/Avila\r\n" +
"SelectedWarehouses=http://virtual/NIW/Avila";

```

```
function blompictometry_onclick() {
    var name="blompictometry";
    __blompictometry_OpenViewer();
    __blompictometry_window.globalLoadSettingsString(__blompictometry_settings);
    if (WaitingFor("map")) {
        WaitAlertFor("map");
    } else {
        // set coordinate units to storage
        top.frames["fraMap"].setCoordinateUnits(1);
        top.frames.fraMap.getPoint(__blompictometry_callback, name);
    }
}
```

In Viewer.htm

```
function LoadSettingsString(s) {
    Object1.LoadSettingsString(s);
}

var globalLoadSettingsString = this.LoadSettingsString;
```

- When the end user clicks in the map having the “Blom Pictometry Viewer” button selected the __blompictometry_callback is triggered. Here we get the x and y coordinates clicked and call GoToCoord method of the ActiveX control using the same trick of the global variable.

In blompictometry.js

```
function __blompictometry_callback(sIOBuf) {
    var iTmp = sIOBuf.indexOf( ":" );

    if (iTmp== -1) {
        alert(sIOBuf);
    } else {
        var X = sIOBuf.substring( 0, iTmp)-0; //Get X
        var Y = sIOBuf.substring( iTmp+1, sIOBuf.length-1 )-0; //Get Y
        // add temp redline
        top.frames["fraMap"].addPoint(X, Y, false);

        __blompictometry_centerX = X;
        __blompictometry_centerY = Y;

        __blompictometry_window.globalGoToCoord(__blompictometry_centerX,
        __blompictometry_centerY);
    }
}
```

In Viewer.htm

```
function GoToCoord(x, y) {
    Object1.GoToCoord(x, y);
}

var globalGoToCoord = this.GoToCoord;
```

- To draw the extents of the images we use two event handlers for both the TotalExtentChanged and VisibleExtentChanged events of the Blom PictometryViewer. These events trigger when a new image is loaded and when the visible portion of the image changes, respectively. As these events are from an ActiveX control, for better management of them we have used the trick of separating the event capture in a new VBScript block. In VBScript is very easy to capture ActiveX events. TotalExtentChanged and VisibleExtentChanged events send the same parameters: the four coordinates of the bounds (in the order needed to draw the bounds), a index of the view that trigger the event (useful for layouts with several views) and a string with the orientation that triggered the event (Ortho, North, South, West or East). To call the main window we have created the global variables __gblompictometry_TotalExtentChanged and __gblompictometry_VisibleExtentChanged in blompictometry.js, so here we call then through the window.opener object that references the main window.

In Viewer.htm

```
<script type="text/jscript">
    function GoToCoord(x, y) {
        Object1.GoToCoord(x, y);
    }
```

```

    }

    function TotalExtentChanged(ulx, uly, urx, ury, lrx, lry, llx, lly, ViewIndex) {
        window.opener.__blompictometry_TotalExtentChanged(ulx, uly, urx, ury, lrx, lry,
        llx, lly, ViewIndex);
    }

    function VisibleExtentChanged(ulx, uly, urx, ury, lrx, lry, llx, lly, ViewIndex) {
        window.opener.__blompictometry_VisibleExtentChanged(ulx, uly, urx, ury, lrx,
        lry, llx, lly, ViewIndex);
    }
</script>
<script type="text/vbscript">
    sub Object1_TotalExtentChanged(ulx, uly, urx, ury, lrx, lry, llx, lly, ViewIndex,
Orient)
        TotalExtentChanged ulx, uly, urx, ury, lrx, lry, llx, lly, ViewIndex
    end sub

    sub Object1_VisibleExtentChanged(ulx, uly, urx, ury, lrx, lry, llx, lly, ViewIndex,
Orient)
        VisibleExtentChanged ulx, uly, urx, ury, lrx, lry, llx, lly, ViewIndex
    end sub

    sub Object1_LayoutChanged(layout)
        LayoutChanged
    end sub
</script>

```

5. The map refresh is done through the `__blompictometry_TotalExtentChanged`, `__blompictometry_VisibleExtentChanged`. These functions call finally the `__blompictometry_RefreshMap` function that calls existing routines in the web application to draw graphics in the map.

In blompictometry.js

```

function __blompictometry_TotalExtentChanged(ulx, uly, urx, ury, lrx, lry, llx, lly,
ViewIndex) {
    var shape = ulx + ":" + uly + ";" + urx + ":" + ury + ";" + lrx + ":" + lry + ";" +
    llx + ":" + lly + ";" + ulx + ":" + uly;
    __blompictometry_OuterBounds[ViewIndex] = shape;
    __blompictometry_RefreshMap();
}

function __blompictometry_VisibleExtentChanged(ulx, uly, urx, ury, lrx, lry, llx, lly,
ViewIndex) {
    var shape = ulx + ":" + uly + ";" + urx + ":" + ury + ";" + lrx + ":" + lry + ";" +
    llx + ":" + lly + ";" + ulx + ":" + uly;
    __blompictometry_InnerBounds[ViewIndex] = shape;
    __blompictometry_RefreshMap();
}

function LayoutChanged() {
    window.opener.__blompictometry_LayoutChanged();
}

function __blompictometry_RefreshMap() {
    top.frames["fraMap"].setCoordinateUnits(1);
    __blompictometry_styles = top.frames.fraMap.getRLStyles();
    __blompictometry_styles[7] = "fill:white;fill-opacity:0.5;stroke:red;stroke-width:2";
    top.frames.fraMap.setRLStyles(__blompictometry_styles);
    top.frames.fraMap.deleteAllRedlines(true);
    for (var n = 0; n < 5; n++) {
        if (__blompictometry_OuterBounds[n] != null) {
            top.frames.fraMap.addPolygon(__blompictometry_OuterBounds[n], true);
        }
        if (__blompictometry_InnerBounds[n] != null)
            top.frames.fraMap.addPolygon(__blompictometry_InnerBounds[n], true);
    }
}

var __blompictometry_TotalExtentChanged = this.__blompictometry_TotalExtentChanged;
var __blompictometry_VisibleExtentChanged = this.__blompictometry_VisibleExtentChanged;

```

