presented at: 10th AGILE International Conference on Geographic Information Science 2007, Aalborg, Denmark
published in: The European Information Society, Lecture Notes in Geoinformation and Cartography, Springer, 2007, 247-264

Increasing the Fitness of OGC-Compliant Web Map Services for the Web 2.0

Thomas Brinkhoff

Institute for Applied Photogrammetry and Geoinformatics (IAPG)
FH Oldenburg/Ostfriesland/Wilhelmshaven (University of Applied Sciences)
Ofener Straße 16/19, D-26121 Oldenburg, Germany
Thomas.Brinkhoff@fh-oldenburg.de
http://www.fh-oow.de/institute/iapg/personen/brinkhoff/

Abstract. Google Maps and Google Earth are very popular web mapping services but they are not open and not OGC-compliant. The question arises why this is not the case. One reason is that commercial data providers require a tight coupling between the provided maps and additional supplements like advertisements. Another answer to this question is the missing support of mashups that become more and more popular with using Google Maps. This paper presents several approaches to overcome these deficits of OGC-compliant web mapping services. The most promising solutions are integrated into an architecture for a Web 2.0 map server and are implemented by a prototype. Scalable Vector Graphics (SVG) is one essential technology within this architecture. By using such approach, the current need for closed systems like Google Earth or Google Maps may diminish.

Keywords: Web Mapping Services, OGC, Google Maps, Google Earth, Mashups, SVG.

1 Introduction

The first *Web Map Service* (*WMS*) implementation specification was published by the Open Geospatial Consortium (OGC) in the year 2000. The current version 1.3.0 [19] dates from 2006 and corresponds to *ISO 19128:2005* “Geographic Information – Web Map Server Interface”. During this period, the WMS has become the most popular web service standard for geospatial data. However, the most popular web services for geospatial data are the non-WMS services *Google Maps* (http://maps.google.com/) and *Google Earth* (http://earth.google.com/). Tom Poiker asks in [22] the question if Google Map is special. This paper deals with the question why Google Maps and Google Earth are not open OGC-compliant web mapping services.

One simple answer is that a company like Google does not (need to) care for standards. In case of *KML* (*Keyhole Markup Language*, [6]) – the geographic markup language of Google Earth – this answer may be the right reply: Using OGC's *Geography Markup Language* (*GML*) [15] as coordinate representation would fulfill the same requirements as the corresponding elements of KML. In the case of the WMS, the answer is more difficult and requires a more careful consideration of the requirements of commercial data providers as well as of the users of web mapping services.

Providing information and services in the Internet requires a return of investment. Because micro payments have not succeeded (and will not succeed in the near future) [24], *advertisements* are the dominant opportunity for earning money. In general, advertisements in the Web are successful if (and only if) they are connected with the user's interests. In other words, the advertisements must be related to the information the user is seeking or to the service the user is calling. In case of geospatial data and services, the location is the essential bridge. That means the map areas must be overlaid by textual or graphical information (the *marking* of the advertisement) and a *hyperlink* (pointing to the destination the user should visit). From the point of view of the information provider it is important that the map is *tightly coupled* with such *supplements*.

An important current trend is that web sites allow users to add their own information (for themselves and/or for other users). Prominent examples are *Wikipedia* (http://en.wikipedia.org/), *YouTube* (http://www.youtube.com/), and *OpenBC* (http://www.openbc.com/). Tim O'Reilly [20] has summarized this trend by the term *Web 2.0*. In case web mapping services, the adding of geospatial information by the user is an essential part of the Web 2.0 paradigm. Google Maps, for example, allows adding *markers* by using a simple API [7] and *Wikimapia* (http://wikimapia.org) requests the users to publish points (or areas) of interest. The result is often a *mashup* of two or more web services; a mashup is a web site or web application that seamlessly combines content from more than one source into an "integrated experience" [28]. The web page "America's most expensive Colleges" is a nice example for this approach (see Fig. 1). In case of Google Earth, the before-mentioned KML has the role of a data integrator. The Google Earth Community (http://bbs.keyhole.com/entrance.php?Cat=0) offers a large selection of KML files produced by the users of Google Earth.

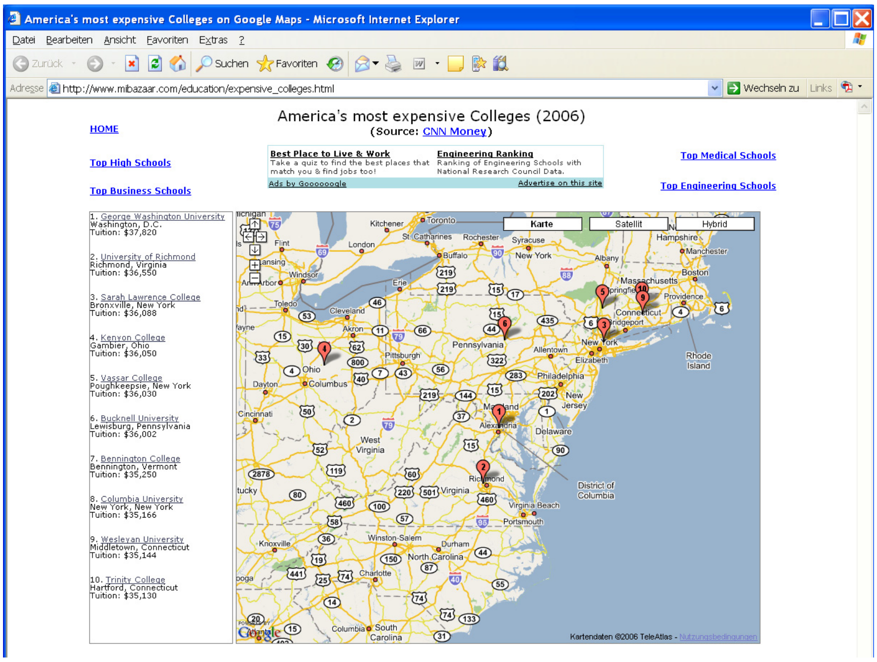

Fig. 1. Example of a mashup combining CNN Money data with Google Maps (http://www.mibazaar.com/education/expensive_colleges.html).

We can summarize that for the success of web mapping services two requirements are important: 1.) for the providers, the possibility to couple the map tightly with textual or graphical supplements and corresponding hyperlinks, and 2.) for users, the opportunity to add their own geospatial information.

In the following, several approaches for achieving these two objectives are presented and discussed. The result of this analysis is an architecture that fulfills the requirements and is conform to the current WMS standard. Consequently, it would be possible for a provider like Google to supply its services as WMS services; the current need for closed systems may diminish.

The rest of paper is organized as follows: The next section compares standardized OGC web services with non-standardized web mapping services. Section 3 presents several approaches for coupling information with map elements. The forth section deals with the question how WMS services can be enhanced to support mashups. Section 5 proposes an architecture that allows the server-side supplement of information and delivers WMS images that are prepared for mashups. Related work is discussed in Section 6. The paper concludes with a summary and an outlook to future work.

2 Web Services

In this section, standardized OGC web services are compared with non-standardized web mapping services.

2.1 OGC and ISO Web Services

The most important web data service specifications are the OCG *Web Feature Service (WFS)* [16] for providing features with *vector geometries* and the OGC *Web Coverage Service (WCS)* [17] for providing *grid coverages*, i.e. raster data. By using these services, a client gets (more or less) original data. The WFS geometries are encoded by (a subset) of GML. This allows sophisticated applications. However, these services are not suitable for open Internet applications because data providers typically do not want to give vector data to anonymous users. Therefore, WFS and WCS are more appropriate for in-house Intranet solutions or they are used by other web services as standardized data sources.

For open web applications, the portrayal *Web Map Service* (*WMS*) [19] (ISO 19128:2005) is more suitable. It provides *maps* that are compiled, resized and styled according to the users' requests. The maps are mostly represented as *raster images*. Thus, the maps are not provided by using their original coordinate system. The WMS consists of two obligatory operations `GetCapabilities` and `GetMap` and of the optional operation `GetFeatureInfo`. `GetCapabilities` offers information about the capabilities of the service and the provided layers. `GetMap` delivers the maps and `GetFeatureInfo` returns a document containing the attribute values of the feature(s) that are located at a specified pixel position. Figure 2 illustrates these three operations by an UML sequence diagram.

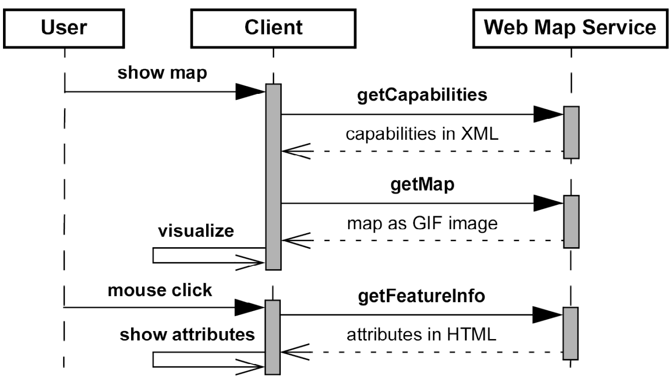

Fig. 2. A typical sequence of operations for using a Web Map Service.

The `GetMap` operation allows overlaying several layers of a WMS service. Furthermore, it is possible that a WMS server fulfills an operation by requesting layers from other web map servers (*cascading WMS*).

2.2 Non-standardized Web Mapping Services

Google Maps and Google Earth are typical representatives of non-standardized web mapping services.

Using *Google Earth* requires specific client software. This client communicates with the Google Earth server by using a non-open protocol that transports the requests of the clients as well as the resulting maps. The maps (satellite and aerial images) are streamed. In consequence, the quality of the maps increases during the streaming process. In addition, predefined layers (e.g. locations of cities and of streets) can be requested from the server. The system is closed: no export functionality is offered and the Google Earth server cannot be used by a WMS client. Instead, the user has the possibility to load *KML files* for overlaying geospatial data [7]. KML can be regarded as simplified mixture of GML and *Styled Layer Descriptors* (*SLD*) [14]. KML allows adding simple points, line strings and polygons with textual descriptions and hyperlinks as well as adding raster overlays that may provided by a WMS. The KML documents are interpreted and visualized by the Google Earth client. KML files can be exchanged between Google Earth users. Figure 3 shows an example.

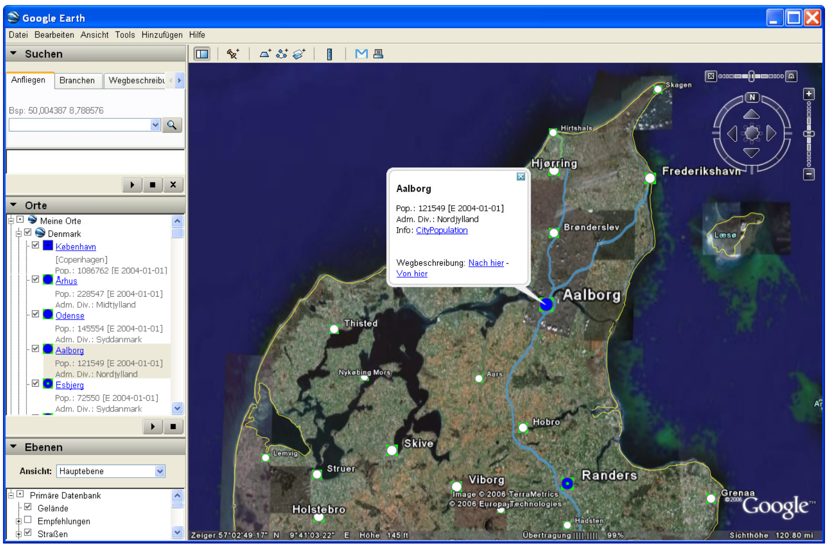

Fig. 3. Example for overlaying Google Earth by KML symbols for cities.

Google Maps is based on a JavaScript library [8]. Therefore, Google Maps uses the web browser as client. Its API supports the addition of user-defined overlays; these overlays may consist of markers (like in Fig. 1), of line strings, or of raster tiles. The last-mentioned class allows the addition of WMS maps. The added objects are interpreted and visualized by the client, i.e. by the web browser using the JavaScript library.

2.3 Comparison

The presentation in the previous subsections demonstrates the essential differences:

1. Non-standardized web mapping services may use OGC web map services but cannot typically be used as WMS (e.g., by a cascading WMS).
2. The presented non-standardized web mapping services allow the addition of geospatial objects on the client site whereas the OGC WMS may combine layers on the server site.
3. Non-standardized web mapping services are closed systems. Therefore, a tight coupling of maps with other geospatial objects is possible.

The following sections demonstrate how these differences can be overcome by OGC-compliant WMS services.

3 Coupling of Supplements

As mentioned in the introduction, the information coupling for maps consists of two steps: 1.) to overlay the map by a textual or graphical *supplement* and 2.) to integrate a *hyperlink* pointing to the destination the user should visit. Because the first task can be easily performed by any web map server, the following presentation and assessment of information-coupling techniques concentrate on the second step.

3.1 GetFeatureInfo

The standard technology for adding alphanumerical information to a WMS map is the use of the `GetFeatureInfo` operation. It retrieves alphanumerical information for a given image position (see Fig. 2) and the client **can** display this information typically by opening an additional window or by using a reserved frame of the main window. The call of the `GetFeatureInfo` operation and the visualization of the result are completely managed by the client that depicts the WMS map. Therefore, such a coupling is not tight. It needs no efforts to avoid it. A further disadvantage is that each `GetFeatureInfo` operation requires two data transfers between the client and the server.

3.2 HTML Image Maps

Another common technique for coupling raster maps (in special) and images (in general) is the use of *image maps*. They have been introduced by HTML 3.2 and are also element of XHTML 1.0 [31][34]. Image maps allow defining areas of an image that are connected with a hyperlink. These areas may be circles, rectangles or polygons. In case of WMS applications, the image map must be separately provided by the WMS server. The image map **can** – if the server provides such operation – be additionally requested by a client. Again, the coupling is loose. Another disadvantage of HTML image maps is their high storage consumption.

3.3 Image Integration

An obvious and effective solution would be the integration of link-sensitive areas directly into the images. However, none of the common image file formats – BMP, PNG, GIF and JPEG – supports such a feature [12]. The definition of specific extensions – e.g., by using the extensibility of the *TIFF file format* [1] – would allow the incorporation of hyperlinks and corresponding link-sensitive areas. However, specific extensions require specific clients or plug-ins. They currently do not exist and it can not be expected that such file formats and plug-ins will be common in the near future.

3.4 SMIL Areas

The *Synchronized Multimedia Integration Language (SMIL)* [36] allows the XML-based integration of multimedia objects. One of its features is the integration of image maps with multimedia objects. The semantics of the `area` element of SMIL 2.1 is the same as it is for HTML image maps: It specifies a spatial portion of a visual object that can be selected as the destination of a link. Figure 4 shows an example.

```xml
<smil xmlns="http://www.w3.org/2005/SMIL21/Language">
  <head>
    <meta name="title" content="SMIL Image Map with a WMS Image" />
    <meta name="author" content="Thomas Brinkhoff" />
    <layout>
      <root-layout id="Welcome" backgroundColor="gray" width="720" height="720" />
      <region id="map" width="700" height="700" left="10" top="10" z-index="2" />
    </layout>
  </head>
  <body>
    <img src="http://www.mapserver.niedersachsen.de/freezoneogc/mapserverogc?
      VERSION=1.1.1&REQUEST=GetMap&LAYERS=DUEKN1000&STYLES=&SRS=EPSG:31467&
      BBOX=3400000,5850000,3500000,5950000&WIDTH=700&HEIGHT=700&FORMAT=png" title="WMS Map"
      dur="60s" region="map">
      <area shape="rect" coords="5,5,50,50" title="super shop" href="http://www.supershop.com" />
      <area shape="rect" coords="60,5,100,50" title="better shop" href="http://www.bettershop.com" />
    </img>
  </body>
</smil>
```

Fig. 4. Example for the `area` element in SMIL documents.

Using SMIL, the WMS `GetMap` would return a SMIL document. Therefore, the client requires a SMIL interpreter for displaying the map and its image map. One essential disadvantage of SMIL is that it does not support different coordinate systems, i.e. it works only with image coordinates. This is a disadvantage for the case where geospatial user information might be added (see Section 5).

3.5 SVG Links

The OGC Web Map Service also supports "graphic element formats" that "constitute a scale-independent description of the graphic elements to be displayed" [19]. Graphic element formats include *Scalable Vector Graphics (SVG)* [35] and the *Web Computer Graphics Metafile (WebCGM)* [33]. The following discussion will concentrate on SVG because it is becoming more and more popular for displaying maps [3][13][21][26].

The World Wide Web Consortium (W3C) has proposed SVG as their XML-based representation of vector graphics. It supports not only vector shapes but also allows embedding of raster images by using the `image` element. Furthermore, *XLinks* [32] can enclose shape elements for linking external resources. Figure 5 illustrates this approach by a small example.

```xml
<?xml version="1.0" encoding="iso-8859-1" standalone="no" ?>
<svg xmlns="http://www.w3.org/2000/svg" xmlns:xlink="http://www.w3.org/1999/xlink" width="800"
  height="814" viewBox="0 0 800 814" zoomAndPan="disable">
  <image x="0" y="0" width="800" height="814" xlink:href="Japan.png" />
  <a xlink:href="../Japan-Saga.html">
    <path title="Saga" d="M 145 507 L 147 507 L 148 507 L 149 507 L 151 507 L 152 506 L 153 507
      L 157 508 L 158 508 L 159 509 L 160 510 L 161 510 L 162 509 L 163 509 L 165 508 L 166 511
      L 165 512 L 164 512 L 164 513 L 163 513 L 163 514 L 162 515 L 161 515 L 160 516 L 159 516
      L 158 518 L 158 520 L 157 519 L 156 520 L 156 519 L 154 518 L 152 518 L 153 519 L 152 520
      L 150 522 L 149 520 L 149 523 L 150 524 L 151 524 L 151 525 L 152 526 L 152 527 L 153 527
      L 152 528 L 147 527 L 145 526 L 144 525 L 140 523 L 140 522 L 141 522 L 141 521 L 140 520
      L 136 519 L 136 518 L 135 517 L 134 515 L 133 514 L 135 513 L 136 513 L 136 514 L 137 514
      L 138 515 L 137 514 L 137 513 L 137 512 L 137 511 L 136 512 L 136 510 L 135 509 L 137 510
      L 138 509 L 137 509 L 135 508 L 135 507 L 136 506 L 137 507 L 137 505 L 138 503 L 139 504
      L 140 504 L 141 503 L 142 504 L 142 505 L 141 506 L 142 507 L 143 507 L 143 508 L 145
      507" />
  </a>
  <!--  and so on   -->
</svg>
```

Fig. 5. Example for images and links in SVG documents.

The example shows that *SVL links* allow a tight coupling between raster images and hyperlinks in one document. Furthermore, it is possible to represent the markers of the links as vector shapes, which increases the quality of drawing and allows a client-based high-lighting of the marker below the mouse cursor. In addition, the content of the raster image may be partially or completely replaced by vector shapes.

4 Technologies for Mashups

Especially the support of mashups requires the possibility that users easily can add their own geospatial data to maps that are provided by web mapping services. In this section, techniques are evaluated that can be used for this purpose.

4.1 WMS Mashup Service

A straightforward solution is to set up a WMS server *("WMS mashup service")* that combines user data sets (in a predefined document format) and maps from other WMS servers. Such a mashup service is a variant of a cascading WMS. A GML-based XML format would be suitable as document format for user data. In this document, the geometries are described in world coordinates. It is task of the mashup server to transform them into graphical elements with image coordinates. Also the styling of the graphical elements is duty of the mashup server. For this purpose, SLD instructions must be provided by the user. Figure 6 illustrates all processing steps.

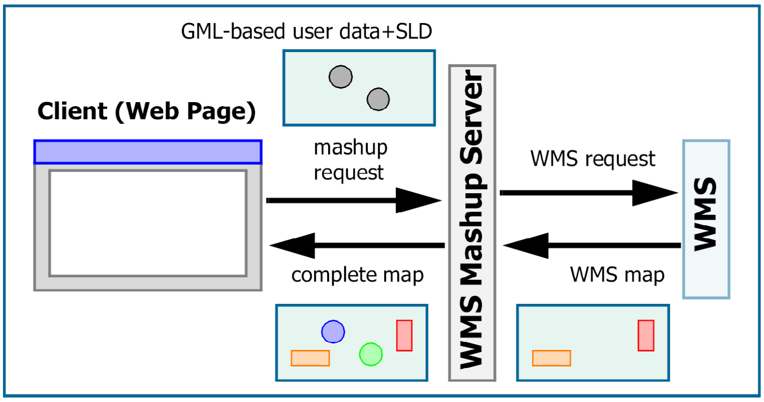

Fig. 6. Processing steps of a WMS mashup service: 1.) The WMS Mashup Server gets a) a WMS request, b) the address of the WMS server, c) a GML-based description of the user data, and d) SLD styling information for the user data. 2.) The WMS request is passed to the WMS server that provides the WMS map. 3.) The WMS map and the transformed user data are combined to the complete map that is delivered to the client.

Note that this WMS mashup service differs from the concept of "user-defined layers" as they are introduced in the SLD specification [14] because user-defined layers require that the user data is provided by a remote OGC service. However, this cannot be expected for performing mashups.

One disadvantage of this approach is that users are required to deliver their data to a third-party's server (or to establish their own WMS mashup server). Furthermore, GML and SLD are not simple document formats that any user can handle.

4.2 SVG Mashups

The use of SVG as image format allows *client-based mashups*. This is possible 1.) because SVG supports JavaScript and 2.) because SVG allows different coordinate systems in different layers of an SVG document.

JavaScript is required for offering operations that allow the adding of user geometries. In order to have a similar functionality as Google Maps [7], the following functions should be offered:

- `function addMarker (x, y, id, title, type, mclass, css)`
- `function addPolyline (points, id, title, mclass, css)`
- `function addTile (x, y, width, height, url, id, mclass, css)`

These functions describe the location of the geometries in world coordinates, define a `title` that can by displayed as tooltip for points and lines, the source (`url`) of the raster image and styling information by the parameters `type`, `mclass` and `css`. `type` defines the type of a marker, `mclass` selects a predefined styling by giving a CSS class name and `css` defines an individual styling by giving the corresponding *Cascading Style Sheet (CSS) declaration* [30]. In contrast to SLD, most web designers are familiar with CSS. The identifier `id` is necessary for removing or changing the user geometries by other JavaScript functions.

The graphical elements for user data are stored in a separate SVG layer (*mashup layer*) using world coordinates. The other layer (*server layer*) contains the WMS data provided by the WMS server in image coordinates. The server layer consists of a raster image or of SVG shapes. This depends of the value of the WMS parameter `FORMAT`. Figure 7 depicts a corresponding SVG document and Figure 8 the resulting image.

The main advantage of this approach is that the user data is completely handled by the client. Another plus is that the request of the WMS is (almost) unaffected by this enhancement. The only difference is that if, e.g., a GIF image is requested by the client (probably because this is the preferred format of the WMS **server**), an SVG image is delivered. Therefore, it would be reasonable to add a parameter `SVGWRAPPING` that indicates that a client accepts the wrapping of the original image format by SVG. Furthermore, SVG mashups perfectly harmonize with the SVG information coupling presented in Section 3.5.

```
<svg xmlns="http://www.w3.org/2000/svg" xmlns:xlink="http://www.w3.org/1999/xlink" width="700"
  height="700">
  <title>SVG mashup</title>
  <!-- The mashup functions  -->
  <script type="text/ecmascript">
    <![CDATA[    ]]>
  </script>
  <!-- Prefined styles  -->
  <style type="text/css">
    <![CDATA[
            use.largecity {fill: red; fill-opacity: 0.5; stroke: #000000; stroke-width: 0.1; }
            use.middlecity {fill: red; fill-opacity: 0.5; stroke: #000000; stroke-width: 0.1; }
            use.smallcity {fill: white; fill-opacity: 0.5; stroke: #000000; stroke-width: 0.1; }
            text.largecity {stroke: black; stroke-width: 0.04px; font-size: 1px; text-anchor: middle; }
            text.middlecity {stroke: yellow; stroke-width: 0.03px; font-size: 1px; text-anchor: middle; }
            text.smallcity {stroke: white; stroke-width: 0.02px; font-size: 1px; text-anchor: middle; }

    ]]>
  </style>
  <!-- Predefined marker types  -->
  <defs>
    <rect id="square" x="-0.5" y="-0.5" width="1" height="1" />
    <circle id="circle" cx="0" cy="0" r="0.5" />
    <polygon id="triangle" points="-0.5,0.4 0.5,0.4 0,-0.4" />
  </defs>
  <!-- Predefined supplements  -->
  <defs>
    <g id="fhoow">
      <a xlink:href="http://www.fh-oow.de">
        <image x="0" y="0" width="150" height="32" xlink:href="logo.gif" />
      </a>
    </g>
  </defs>
  <!-- The server layer  -->
  <svg width="100%" height="100%" viewBox="0 0 700 700">
    <image x="0" y="0" width="700" height="700" xlink:href="http://wms.jpl.nasa.gov/wms.cgi?
      REQUEST=GetMap&VERSION=1.1.1&LAYERS=global_mosaic&
      STYLES=&SRS=EPSG:4326&BBOX=7.5,52.5,9,54& WIDTH=700&HEIGHT=700&FORMAT=image/jpeg" />
  </svg>
  <!-- The server supplement layer (empty)  -->
  <svg width="100%" height="100%" viewBox="7.5 -54 1.5 1.5">
    <g id="supplementgroup" transform="scale(1 -1)" />
  </svg>
  <!-- The mashup layer  -->
  <svg id="mashups" width="100%" height="100%" viewBox="7.5 -54 1.5 1.5">
    <g id="mashupgroup" transform="scale(1 -1)">
      <use xlink:href="#circle" transform="translate(8.2275,53.1375) scale(0.05)" title="Oldenburg"
        class="middlecity" />
      <use xlink:href="#square" transform="translate(8.8075,53.0761) scale(0.075)" title="Bremen"
        class="largecity" />
      <use xlink:href="#circle" transform="translate(8.1081,53.5397) scale(0.05)" title="Wilhelmshaven"
        class="middlecity" />
      <use xlink:href="#circle" transform="translate(7.9981,53.1844) scale(0.0375)" title="Bad Zwischenahn"
        class="smallcity" />
      <use xlink:href="#circle" transform="translate(7.9256,53.2581) scale(0.0375)" title="Westerstede"
        class="smallcity" />
    </g>
  </svg>
</svg>
```

Fig. 7. Example of an SVG mashup after performing five additions. Note that instead of including the WMS image as a WMS link also the binary image data can be included by using a base64 coding.

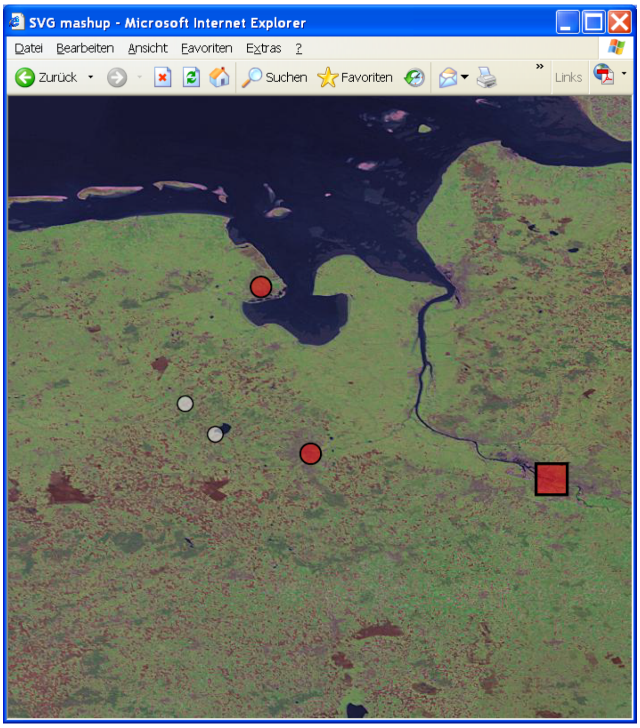

Fig. 8. Result of the SVG mashup of Figure 7.

5 Architecture

In order to evaluate the presented approaches for an information coupling and for mashups, an architecture has been designed which implements them on the top of existing OGC compliant web map servers. Figure 9 gives an overview of the architecture that supplements the request by using SMIL (Section 3.4) or SVG (Section 3.5) and / or computes SVG mashups (Section 4.2).

The presented architecture allows supplementing the map by using SMIL or SVG. In both cases a spatial database is requested with the query window of the original WMS `GetMap` request and the name(s) of the requested layer(s). These two criteria are used to select the suitable supplements that consist of geometry (in world coordinates), display information (text/icon), and hyperlink.

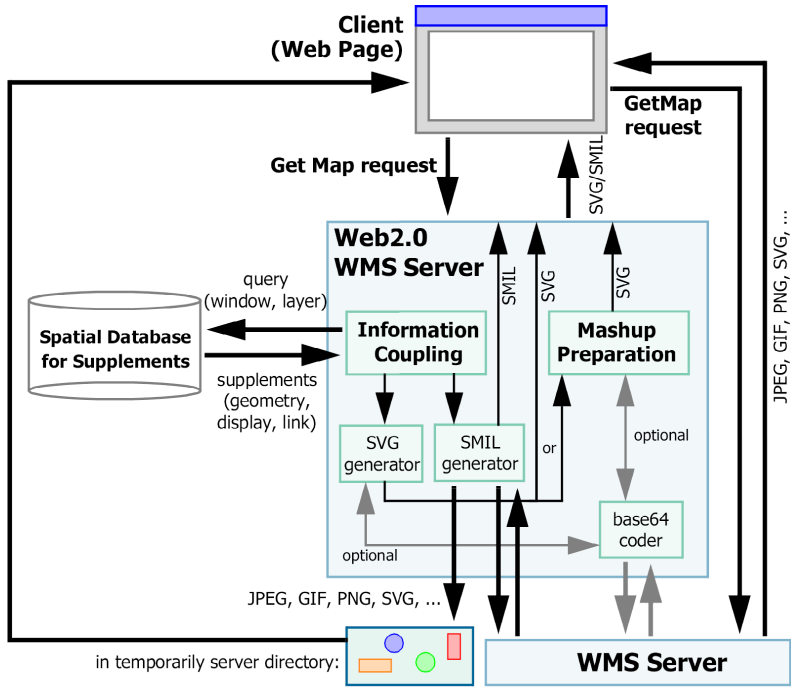

Fig. 9. Architecture for supporting supplements and SVG mashups.

In case of SMIL, the WMS request is performed and the coordinates of the geometry of the supplement are transformed into image coordinates. Then, the supplement information can be drawn into the WMS image and the resulting image is stored in a temporarily directory. Finally, the SMIL document is computed using the transformed geometry and a link to the supplemented image. A mashup is not supported for SMIL.

In case of using SVG for the information coupling, the processing is simpler: depending on the request, an SVG document is computed that consists of the base64-coded WMS image (requested from the WMS server) or of a link to the WMS request. In both cases, SVG link and path elements are computed; in contrast to Figure 5, the world coordinates are not transformed in order to improve performance. The resulting SVG document is returned to the client or further processed by the “Mashup Preparation” unit.

The “Mashup Preparation” unit extends the server layer, which contains the WMS request or the base64-coded WMS image (and the possible supplements), by (a) the mashup functions, (b) by the prepared styles and marker types, and (c) by an empty `svg` element for the mashup layer. The resulting SVG document is returned to the client.

A prototype of the system was implemented using deegree2 [11] as web map server, Postgres 8.1/PostGIS 1.1 [23] as spatial database system for the supplements, and Apache Tomcat 5.5 [2] as servlet container. The functionality is programmed by Java 5 servlets using JTS 1.7 [9] as geometry representation and GeoTools 2.2 [4] in combination with the EPSG database 6.8 [8] for coordinate transformations.

Figure 10 shows a simple web client displaying a map that was produced by the presented architecture. It supports adding, modifying and removing towns by calling the respective functions of the SVG image. The towns are displayed by a symbol and a label. The “Fachhochschule” icons and links were added by the server using the supplement database.

Fig. 10. Example for a map that contains supplements and supports mashups.

6 Related Work

The presented work takes the successful web services Google Maps and Google Earth as starting point of the considerations. In contrast to OGC or ISO specifications, the process of their development can only be comprehended by taking the final results, i.e. the documentation of the Google Earth API [6] and of KML [7]. Publications about theses services present (the success of) these services (e.g. [25] [29]), but do not treat the question, how such services can be also realized by standardized web services. The author is not aware about publications dealing with the **tight** coupling of links with WMS maps or with SVG mashups for standard-compliant web mapping services.

The presented WMS Mashup Service can be considered as an instance of a Web Processing Service [18]. As already discussed in Section 4.1, "user-defined layers" as they are defined in the SLD specification [14] differ from the concept of a WMS Mashup Service.

7 Conclusions

This paper presented an architecture that supports a tight coupling of supplemental information with WMS results and extensions that allows users easily to add their own geospatial objects to WMS images. This work was motivated by the observations that data providers often require such supplements (e.g. for advertisements) to finance their services and that users (including web pages based on web mapping services) often want to integrate several data sources ("mashups").

The investigation favored SMIL areas and SVG links for a tight information coupling. Furthermore, a WMS Mashup Service and SVG mashups were proposed for integrating graphical objects. For pursuing both objectives, the combination of SVG links and SVG mashups performed best. SMIL however does not support mashups and the WMS Mashup Service requires a new web service with rather complicated parameters.

Using the proposed approach, it is possible for geospatial data providers to supply their data by an open WMS service; the current need for closed systems like Google Earth or Google Maps may diminish.

Further work consists of increasing the robustness and performance of the current prototype implementation. This is necessary before – as second step – it can be made available for the public. The current approach of selecting supplements by a query window and the name(s) of the requested layer(s) is rather simple and should be replaced by a more elaborated algorithm. Also the placement of the supplemented texts and icons is improvable. The design of the current mashup functions was oriented at the functions of Google Maps. The number of the mashup functions can be increased by considering the functionality of KML. Because of the powerfulness of SVG, it can be expected that most of the relevant KML functions can be provided. An ongoing task is to consider the varying functionality of different SVG viewers.

Acknowledgments. Most of the infrastructure mentioned in Section 5 was set up for the project “OK-GIS: Offenes Katastrophenmanagement mit freiem GIS” together with Jürgen Weitkämper and Christian Rolfs. That project is funded by the German Federal Ministry for Education and Research (BMBF). This support is gratefully acknowledged.

References

1. Adobe Developers Association (1992) TIFF, Revision 6.0, Final.
2. Apache Software Foundation: Apache Tomcat. http://tomcat.apache.org/
3. Brinkhoff T, Weitkämper J (2005) Mobile Viewers based on SVG$^{\pm geo}$ and XFormsGI. Proceedings 8th AGILE Conference on Geographic Information Science, Estoril, Portugal, pp 599–604
4. GeoTools Project: GeoTools – The Open Source Java GIS Toolkit. http://geotools.codehaus.org/
5. Gibson R, Schuyler E (2006) Google Maps Hacks. O’Reilly
6. Google Inc. (2006) Google Earth KML Documentation Version 2.0. http://earth.google.com/kml/
7. Google Inc. (2006) Google Maps API Version 2 Reference. http://www.google.com/apis/maps/
8. International Association of Oil & Gas Producers – Surveying & Positioning Committee: EPSG Geodetic Parameters. http://www.epsg.org/
9. JUMP Project: JTS Topology Suite (JTS). http://www.jump-project.org/project.php?PID=JTS&SID=OVER
10. Kraak M-J, Brown A (2001) Web Cartography. Taylor & Francis
11. lat/lon GmbH: deegree – Free Software for Spatial Data Infrastructures. http://www.deegree.org/
12. Miano J (1999) Compressed Image File Formats: JPEG, PNG, GIF, XBM, BMP. ACM Press
13. Neumann A, Winter AM, Neumann I (2000) Vector-based Web Cartography: Enabler SVG. carto:net (2000), http://www.carto.net/papers/svg/index_e.shtml
14. Open Geospatial Consortium, Inc. (2002): Styled Layer Description (SLD) Implementation Specification, Version 1.0.0, OGC 02-070
15. Open Geospatial Consortium, Inc. (2004) Geography Markup Language (GML) Implementation Specification, Version 3.1.1, OGC 03-105r1
16. Open Geospatial Consortium, Inc. (2004) Web Feature Service (WFS) Implementation Specification, Version 1.1.0, OGC 04-094

17. Open Geospatial Consortium, Inc. (2005): Web Coverage Service (WCS) Implementation Specification, Version 1.0.0, OGC 05-076
18. Open Geospatial Consortium, Inc. (2005) Web Processing Service (WPS) Discussion Paper, Version 0.4.0, OGC 05-007r4
19. Open Geospatial Consortium, Inc. (2006) Web Map Service (WMS) Implementation Specification, Version 1.3.0, OGC 06-042
20. O'Reilly T (2005) What is the Web 2.0? – Design Patterns and Business Models for the Next Generation of Software http://www.oreillynet.com/pub/a/oreilly/tim/news/2005/09/30/what-is-web-20.html
21. Peng Z-R, Zhang C (2004) The roles of geography markup language (GML), scalable vector graphics (SVG), and Web feature service (WFS) specifications in the development of Internet geographic information systems (GIS). Journal of Geographical Systems, Springer, 6:95–116
22. Poiker T (2006) Is Google Map Special? GEOconnexion International Magazine No. 9, p 50
23. Refractions Research: PostGIS. http://www.postgis.org/
24. Shirky C (2000) The Case Against Micropayments. O'Reilly Network http://www.openp2p.com/pub/a/p2p/2000/12/19/micropayments.html
25. Soutschek M (2006) Google Earth – Neuer Platzhirsch im Geo-Revier? GeoBIT 1/2, 8–15
26. Ueberschär N, Winter AM (2006) Visualisieren von Geodaten mit SVG im Internet, Wichmann-Verlag
27. Vatsavai RR, Shekhar S, Burk TE, Lime S (2006): UMN Map Server: A High Performance, Interoperable, and Open Source Web Mapping and Geo-Spatial Analysis System. 4th International Conference on Geographic Information Science, Münster, Germany. In: Lecture Notes in Computer Science, Springer, 4197:400–417
28. Wikipedia: Mashup (web application hybrid). http://en.wikipedia.org/wiki/Mashup_(web_application_hybrid)
29. Wilk C (2006) Welt in Händen – Arbeiten mit Google Earth und World Wind. iX 12, pp 50–62
30. World Wide Web Consortium (1998) Cascading Style Sheets Level 2 (CSS 2) Specification. W3C Recommendation
31. World Wide Web Consortium (1999) HTML 4.01 Specification. W3C Recommendation. http://www.w3.org/TR/1999/REC-html401-19991224
32. World Wide Web Consortium (2001) XML Linking Language (XLink) Version 1.0. W3C Recommendation. http://www.w3.org/TR/2001/REC-xlink-20010627/
33. World Wide Web Consortium (2001) WebCGM 1.0 Second Release. W3C Recommendation. http://www.w3.org/TR/2001/REC-WebCGM-20011217/
34. World Wide Web Consortium (2002) XHTML 1.0 The Extensible HyperText Markup Language (2nd Edition). W3C Recommendation. http://www.w3.org/TR/2002/REC-xhtml1-20020801
35. World Wide Web Consortium (2003) Scalable Vector Graphics (SVG) 1.1 Specification. W3C Recommendation. http://www.w3.org/TR/2003/REC-SVG11-20030114/

36. World Wide Web Consortium (2005) Synchronized Multimedia Integration Language (SMIL) 2.1 Specification. W3C Recommendation http://www.w3.org/TR/2005/REC-SMIL2-20051213/