TimeMapper — generating animated SVG from a WMS to visualise moving object data

Barend Köbben[1], Timothée Becker[2], and Connie Blok[1]

[1]ITC – University of Twente, Faculty of Geo-Information Science and Earth Observation, Enschede, The Netherlands (`kobben@itc.nl`); [2]Laboratoire d'étude des Transferts en Hydrologie et Environnement (LTHE), Grenoble, France

Abstract. Within a larger aim of improving automated vector animated mapping, the main objective of this research was to look into the possibility of combining two technologies: distributed geo–webservices and animated, interactive vector maps. TimeMapper was developed as prototype for an OGC–compliant Web Map Service implementation that serializes spatio–temporal data from a database backend as Scalable Vector Graphics. The SVG is used in a web browser to show animated maps with a built–in advanced user–interface. This interface allows the user to interact with both the spatial and the temporal dimensions of the data. The potential and limitations of the TimeMapper WMS were explored in a prototype using Antarctic iceberg movement data. The main features that were implemented are three types of temporal legend, a time–slider and a speed control, all combined with standard mapping interface features, such as zoom, pan and layer switching. The prototype can be tested on the TimeMapper website [10].

Keywords: spatio–temporal data, animated maps, geo–webservices, SVG, SMIL, WMS, SDI$^{\text{light}}$

1 Motivation and objective

The motivation for the TimeMapper research was the improvement of automated animated vector mapping, specifically by looking into the possibilities of the loose coupling of distributed geo–webservices with animated, interactive vector maps. It can be seen as an effort to bridge the subject matter of two research groups at ITC (the Faculty of Geo–Information Science and Earth Observation of the University of Twente). On the one hand the Spatial Data Infrastructure Technologies group that deals with the methods and techniques for the design, implementation, maintenance and exploitation of service–oriented, high–volume spatial data systems; On the other hand the Spatio–Temporal Data Integration & Visualization group, whose subjects include geo-visual analytics, time, visual representations, and animation. In this group, we are especially interested in animation, as interactive animated mapping has been pointed out (in [1], among others) as the only technique to be generically applicable to visually analyze the dynamic nature of real world phenomena.

The reason we want to combine animated mapping with distributed geo–webservices is simple: There are many technologies available for producing interactive animations, but none (that we're aware of) that facilitate the production of vector animated maps, *automatically* and *directly*, from spatio–temporal data to a format suitable for internet dissemination.

'Automatically' means that the animated maps will be generated from the spatio-temporal data by the system "working by itself with little or no direct human control" (which is how 'automatic' is defined in [7]). It has to be noted that this automation will nót include the cartographic decisions as to what type of animated map to use for different data–types and -instances. That type of automation would make a system "produce results otherwise done by hand, or (...) simulate human (...) action" (from the more extensive definition in [19]). As in most current GIS and SDI systems, this level of automation of cartography has not been achieved in our set–up, and the link between data- and visualisation-type has to be made by a human, setting up the appropriate parameters beforehand.

By 'directly' we mean that the data can be used in the form(at) it was stored in originally, without conversion or pre-processing needed for the purpose of visualisation. This is important because we want our system to fit in an interoperable SDI context, where it would be an SDI node, as defined in [6]: "a single system node in an SDI network, which archetypically has data, catalog or portrayal services (...)". In such a context, TimeMapper would be just another service, that should be able to consume data from any other SDI node. In such a service-oriented, distributed architecture, the visualization layer should be loosely coupled to the other layers of the system. For this, OGC's Web Map Services (WMS and related specifications, see [16, 17]) are especially useful. These geowebservices are designed to take their input from a variety of (distributed) sources and generate output meant for internet dissemination. Current WMS implementations commonly use so-called 'picture' formats, such as GIF and JPEG. But the specification also allows for 'graphic element' formats, vector formats with added possibilities, such as scripting and animation. At the time of our research, two graphic element formats were widely used to produce vector animations for the internet: Scalable Vector Graphics (SVG) and Adobe Flash. SVG was our choice, firstly because its specification allows the use of SMIL (Synchronized Multimedia Integration Language), a declarative XML–based language well suited for building data driven animations; And secondly, because it is an Open Standard (governed by the W3C, see [20]), whereas Flash is a proprietary technology.

The **objective** of this research was therefore *to design an animated mapping system which can generate SVG SMIL animations, automatically and directly, from spatio–temporal data.* The scope of vector animated mapping is broad and we therefore started by developing a prototype system for one specific type of real-world phenomenon: the movement of objects in geographic space. To achieve our objective, we looked into extending the capacity of our existing SVG–based Web Mapping System, RIMapperWMS, described in 3. Because RIMapperWMS is part of our SDI^{light} approach, we briefly discuss that first in section 2. We explain the extensions necessary and their set–up and introduce our prototype, called TimeMapper, in more detail in section 4. We evaluate the prototype in section 5 and based on that draw some conclusions and give an outlook on further work (6).

2 The SDI^{light} approach

The SDI^{light} approach has been used for several years now in our teaching and research at ITC, and has been discussed in detail elsewhere (in [12], among others). In short, it is a down–to–earth approach towards Spatial Data Infrastructures, using open standards whenever available, and open source solutions where possible. This approach provides researchers, students and our project partners with a platform for relatively simple, lowcost, yet powerful ways of sharing data amongst various stakeholders. The main components of the platform are firstly a spatial database back end (we most often use the object-relational DBMS PostgreSQL and its spatial extension PostGIS). Secondly, there are middleware web applications that interface with the database back end and with each other, and fulfill tasks such as delivering maps, data and processing services. We use existing applications (MapServer and GeoServer), but also develop our own components, of which RIMapperWMS and TimeMapper are examples. And finally there are open source desktop clients (such as QuantumGIS and ILWIS), as well as browser–based clients (such as OpenLayers and the SVG GUI part of TimeMapper), that enable access to the maps and data.

SDI^{light} is intended as "the place where we can show fellow researchers, consultants and students as well as possible users (such as GIS and Cartography departments in developing countries) that the things we teach can be made to work quite quickly, in a relatively simple and low–cost setup. This approach has been functioning well for some five years now." [12, p.30]

3 RIMapperWMS

Based on the SDI^{light} approach, several software development projects have been undertaken, and RIMapper, described in more detail in [11], was one of those. The software started out as the visualisation part of an urban risk management system, hence the name, an acronym for Risk Indicator Mapper.

RIMapperWMS was a further development to make the software compliant with OGC's Web Map Service (WMS) specifications [16]. Compared to existing WMS implementations it stands out firstly because it serves its maps in the Scalable Vector Graphics (SVG) format, and secondly because RIMapperWMS can produce SVG output with a built–in Graphical User Interface (GUI), allowing the data to be disseminated to any SVG-capable application, without the need for a separate WMS client. RIMapperWMS is a set of Java servlets and classes that can be deployed in any J2EE compatible servlet container. The WMS is configured using the database back end. This database also includes the spatial and attribute data to be mapped, stored according to the OGC 'Simple Features for SQL' specification [15]. Since the description in [11], the software has been further developed to version 1.0, and is now a fully functional version 1.1.1 Basic Web Map Service. The GUI has been improved by including a layer switcher and an info tool, and is now draggable. A transcoder has been added for the ability to output PNG and JPEG. Also, URLs to other WMS services now can be used as input data, and the resulting map layers will be included as `<image>` elements in the SVG. This way so-called cascaded WMS layers can be offered.

RIMapperWMS is published under an open source license and the components, their source, as well as some documentation can be found on the website [9].

4 The TimeMapper prototype

We designed a prototype system, called TimeMapper, to add output of animated SVG and associated GUI extensions to RIMapperWMS. To focus on animated mapping of movement of objects in geographic space, we chose the visualisation of the dynamics underlying the calving and movement of icebergs. The data used is the U.S. National Ice Center (NIC) Antarctic icebergs dataset.

4.1 Test case data: Antarctic icebergs

This data consists of movement data, gathered since 1978, of 301 icebergs (adding up to 15737 records at the time of writing) that are described on and can be downloaded from the NIC website [14]. Besides the iceberg position (in latitude and longitude) and time of recording, the dataset contains basic descriptive information (name, size, et cetera). The temporal resolution is quite irregular, ranging from 1 day between consecutive positions, to up to 50 days. The iceberg dataset of the NIC has been used by various researchers with various objectives. Many of them have described the importance of visualising the dynamics of movement as well as calvings or splits, appearances and disappearances, of the icebergs. For example, Schmittner et al. [18] used these as climate change indicators, and Benn et al. [4] looked at the relationship with external factors (e.g. winds and ocean currents). To be useful for visualising the properties and relations of the data, the TimeMapper prototype would have to be able to show animated point symbols (depicting icebergs), together with relevant topographic data (to give an appropriate spatial reference) as well as relevant other data (such as ocean currents).

4.2 Extending the RIMapper system

To extend the RIMapperWMS system to achieve the mapping of temporal data as SMIL SVG animations, we addressed the following points:

– Firstly, on the database side, we determined how to store the temporal components of the data, compliant with relevant standards.

– Secondly, we designed SVG SMIL animation types we could use to represent the real–world dynamics.
– Thirdly, we needed to develop an algorithm for converting the temporal component of the data from database format to the internal SMIL format within SVG.
– Fourthly, we determined, based on animated mapping literature and our experiments, which user–interface elements and which interactivity we would offer to the user, and how those could be implemented in an SVG environment.
– Finally an appropriate architecture for the overall system was designed.

4.3 Storing the temporal component of the data

For this, we simply followed OGC specifications, which are based on the ISO 8601 standard. This defines the encoding of time stamps and time periods in single strings, following the format `ccyy-mm-ddThh:mm:ss.sssZ`: Up to 14 digits specify century, year, month, day, hour, minute, seconds, and optionally a decimal point followed by zero or more digits for fractional seconds, with non–numeric characters to separate each piece. All times should be expressed in Coordinated Universal Time (UTC) as indicated by the suffix Z (for "zulu"). Thus the timestamp `2009-10-04T14:50:58Z` indicates October 4th 2009 at 2:50 pm and 58 seconds.

4.4 Designing the SVG SMIL animations

In geographic theories on change, different types of change are usually recognized. Based on Bloks work [5], Andrienko et al. [2] proposed three categories of change: existential change (e.g. appearance/disappearance), change of spatial properties (e.g. location, shape) and change in attribute (e.g. land-cover change). These changes occur in the of geographic objects, usually divided into into polygons, lines and points. In our present test–case, we focus on *point* objects, i.e. the icebergs, and their changes in *position* over time.

To show the movement of the objects, two main types of animations were used, *stepwise* animations and *linearly interpolated* animations. Such animations are easy to declare using SVG SMIL. The following code shows an example of an animation for the step–wise movement of a circle symbol, depicting iceberg A22B:

```
<circle id="IB_A22B" cx="0" cy="600" r="25" >
  <animate id="XanimIB_A22B_0" attributeName="cx"
     repeatCount="none" fill="freeze" begin="0s"
     from="0" to="200" dur="2s" calcmode="discrete" />
  <animate id="YanimIB_A22B_0" attributeName="cy"
     repeatCount="none" fill="freeze" begin="0s"
     from="600" to="550" dur="2s" calcmode="discrete" />
</circle>
```

The first `animate` element will lead to movement along the x-axis, the second along the y-axis. The `begin` attribute tells the animation when to start and the `dur` attribute sets its duration and therefore its apparent speed. The `calcMode` attribute is set to "discrete", which means we will see a stepwise animation. In brief, the effect of these two animations is that the object is going to jump from 0 to 200 on the x–axis and from 600 to 550 on the y–axis after a count of 2 seconds. For a linearly interpolated animation, the `calcmode` attribute would have been set to "linear", and then the movement would be gradual over the course of 2 seconds.

In our moving–object prototype, the interpolated animation type is offered to the users for them to be able to appreciate the movement of the objects taken individually or in small groups. In other words, we are interested in the dynamics of the trajectories. To help visualise this, we added animated tracks. SVG SMIL doesn't have a built–in way of animating growing line segments. However, members of the SVG community have developed a workaround by animating the `stroke-dasharray` attribute of lines. The result can be seen in the screen–dump in figure 1 as the thin lines trailing the circles depicting the icebergs.

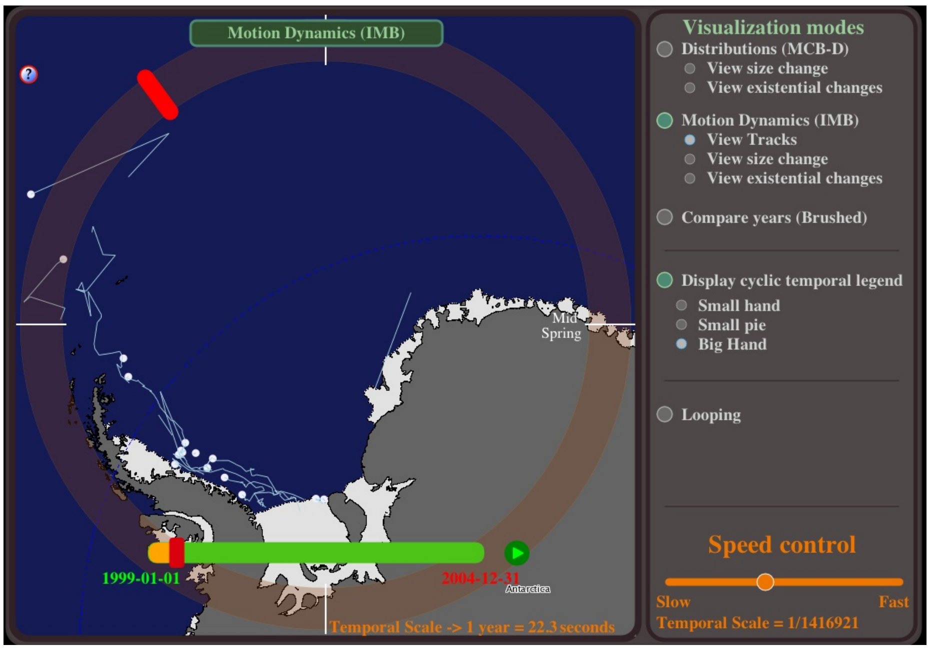


Fig. 1. Prototype of interactive animation of 19 icebergs in the Weddel Sea (Antarctica) from January 1, 1999 to December 31, 2004. Interactive original can be loaded from the TimeMapper website [10]

4.5 Converting the temporal dimension of the data

In the two previous sections, we first saw the way that the data is stored in the database and then, how time is used in SMIL animations. We therefore needed to find a way to go from real–world time in the database to display time in the animations. Three steps effectuate the necessary conversion. These three steps are illustrated in figure 2.

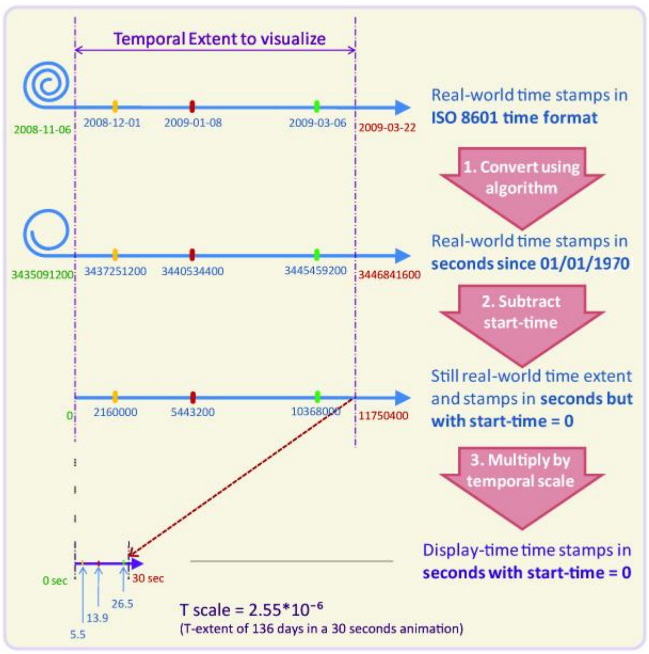

Temporal Extent to visualize
2008-11-06
2008-12-01
2009-01-08
2009-03-06
2009-03-22
Real-world time stamps in ISO 8601 time format
1. Convert using algorithm
3435091200
3437251200
3440534400
3445459200
3446841600
Real-world time stamps in seconds since 01/01/1970
2. Subtract start-time
0
2160000
5443200
10368000
11750400
Still real-world time extent and stamps in seconds but with start-time = 0
3. Multiply by temporal scale
0 sec
30 sec
5.5
13.9
26.5
Display-time time stamps in seconds with start-time = 0
T scale = 2.55*10⁻⁶
(T-extent of 136 days in a 30 seconds animation)

Fig. 2. Steps in converting from real-world time to display time.

The first step is to convert the ISO 8601 time stamps to a single unit time format. For this, we converted them to a `timeinseconds` value, i.e. the number of seconds since a fixed starting point. This starting point is arbitrary, we use the so–called *epoch* 1/1/1970, because Java(script) as well as SQL use this in their `Date` functions.

The second step is necessary for the animations to start at the right time. The first animations will generally start soon after the beginning of the time extent chosen by the user, which in display–time is time 0. To achieve this, we subtract the start–time value from all time–stamps.

If the animations were viewed at this stage, they would last as long as the real–world events. The third step is scaling time, by multiplying all time–stamps by a ratio that we call the *temporal scale*. Just as in traditional cartographic spatial scale (the ratio between map distances and real–world distances), in animated cartography the temporal scale of an animation is the ratio between display–time and real–world time [8]. Most often, as in our prototype, time will be shrunk for the viewer to visualize in a short time the events which happened during a comparatively long period.

4.6 Developing the animated mapping user interface

Most animated mapping theorists agree that the user should be helped in their visualization task by two elements: temporal legends and ways to interact with the temporal dimension of the maps. The authors of [13], among others, state that temporal legends are necessary to understand a temporal animation. Three main types of temporal legends exist: digital clocks, time bars and cyclic temporal legends. All three of these legend types were developed for our prototype.

Various interactive functionalities were developed for the graphic user interface (GUI): to enable the user to choose between stepwise and interpolated animations, to add or remove interpolated tracks, and to choose which cyclic legend to view (if any). The mechanism behind these interactive choices is a series of javascript functions changing the attributes of the various SVG elements that make up the GUI and the map. In this paper, we do not go into details on the choices for and set-up of the various GUI elements. They were based on literature review and experiments (described in the MSc thesis of Becker [3]).

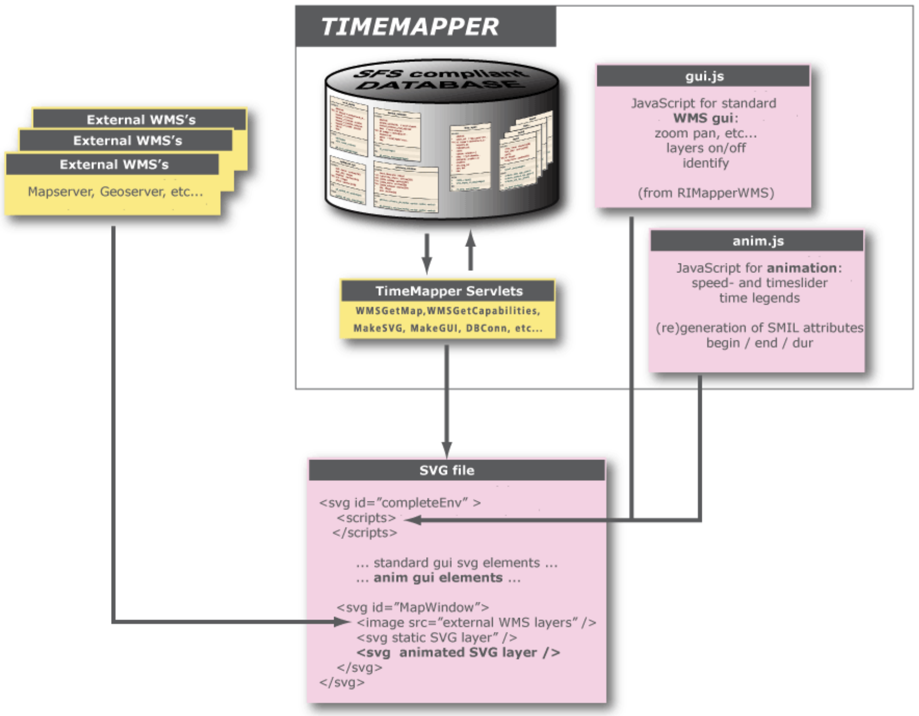
In figure 1 you can see various elements of the GUI: the *Big Hand* cyclic temporal legend has been chosen, and it is visible as a transparent circle over the whole map image, with a red bar that travels the full 360 degrees in one year of real–world time.

4.7 System architecture

To bind all elements together we used an overall system architecture that is, as mentioned before, basically an extension of RIMapperWMS (see section 3). We therefore give here only a relatively general overview, depicted in figure 3.

The spatial database back end is used for storing both the configuration of the Web Map Services as well as the actual spatial and attribute data the maps are derived from. A template SVG file defines the setup of the map and its GUI: the necessary scripts for the basic WMS GUI as well as the animation GUI are loaded, and the map layers are defined as a set of nested SVG elements. The content of the map layers is not limited to animated SVG. External WMS layers can also be included, for instance in the example in figure 1, the land and ice shelf are (raster) layers from the Antarctic Cryosphere Access Portal OGC services (`http://nsidc.org/agdc/acap`). Additonally, one or more non–animated SVG layers can be retrieved from TimeMapper or RIMapperWMS services.

The application tier is a set of Java Servlets that can be deployed in any J2EE compatible servlet container. The servlets do recurring tasks like extracting OGC features and attribute data from the database, translating these into fragments of SVG, SVG SMIL and JavaScript, collecting and structuring these fragments into valid output and delivering this output to the clients. The web interfaces



```
<svg id="completeEnv" >
   <scripts>
  </scripts>

      ... standard gui svg elements ...
      ... anim gui elements ...

   <svg id="MapWindow">
      <image src="external WMS layers" />
      <svg static SVG layer" />
      <svg  animated SVG layer />
   </svg>
</svg>
```

Fig. 3. TimeMapper system architecture overview.

of the servlets comply to OGC's Web Map Service 1.1.1 standard [16]: Firstly *GetCapabilities*, returning an XML description of the WMS information content and acceptable request parameters and secondly *GetMap*, returning the map itself.

5 Evaluation of the prototype

With our prototype of TimeMapper, a series of three test-cases was generated, all depicting the same five year period (1999–2003), with an increasing number of icebergs displayed. The *small* test–case shows 3 icebergs, with a total of 218 separate occurrences. The *middle* test–case has 19 icebergs, all the icebergs present in the Weddell Sea region during the five years, accounting for 811 occurrences. The *big* test-case shows 99 icebergs, those are all icebergs in the whole Antarctic Ocean during the five years, which makes for 4825 occurrences.

For the moment, testing has only been done using the Opera web browser, as at the time it was the only web browser supporting the full SVG SMIL specification. Others user agents are catching up fast, and right now Mozilla Firefox and Webkitbased browsers (e.g. Safari and Chrome) support almost all of TimeMappers functionality. Microsoft recently announced that they will implement SVG natively starting with Internet Explorer 9, although it looks like SMIL animation will not be supported in the first version.

While testing the prototype with the small test-case, we found all the features working as planned. The speed setting system as well as the time-slider work particularly well. With this small test–case, the time-slider is very responsive. The user can interact with the temporal moment of the animation with hardly any response delay, creating a true time-brushing effect.

However, with larger test-cases, there are important responsiveness limitations, that mostly affect the speed setting mechanism and the time slider. Setting the speed of the animations with the middle test–case takes a few seconds, after which the animations themselves work fine. But although interacting with the time slider while the animation runs still works, it is too slow to constitute a usable exploratory tool. Finally, the big test–case takes a very long time (over 60 seconds) to load. Setting the speed of the animations and interacting with the time slider hardly works at all, although once it is finally running, the animation is still reasonably smooth.

In recent meetings at the SVG Open developers conference (`http://svgopen.org`), we discussed the performance issues with SVG implementors (from Opera, among others). The lack of performance in interactivity seems to be mainly due to the fact that the browser needs to manipulate large amounts of objects in the DOM tree that represents the SVG in memory. DOM manipulation is a recognized performance problem in current browser implementations. With our present animation set-up, each segment travelled by an object leads to two animation objects (one x and one y) per segment travelled. We plan to do some testing using an alternative coding method for the animation, using `keyTimes` and `KeyValues`. With these SMIL methods, the whole trajectory (i.e. the sum of the segments) travelled by an object would lead to only two animation objects and we feel this may improve the responsiveness of the animations to the time slider and the speed–setting mechanism.

6 Conclusion

We can conclude that combining distributed geo–webservices and animated, interactive vector maps in a service–oriented, distributed architecture is possible with the use of existing interface, service and data standards. We have built an OGC-compliant Web Map Service implementation that serializes spatio–temporal data from a database backend as animated Scalable Vector Graphics. The SVG is used in a web browser to show animated maps with a built-in advanced user-interface. This interface allows the user to interact with both the spatial and the temporal dimensions of the data.

A use case for the visualization of the movements of Antarctic icebergs was developed and our test-cases show that the system works as planned. There were clear performance problems in the interactivity, which makes this particular implementation less useful for exploratory purposes, where time-brushing and similar interactive behaviour is needed. But for less demanding tasks, where the speed of the animation can be fixed and brushing is not needed, the system could be very useful. Besides that, we have identified the cause of the interactivity performance limitations, and feel we have pointers to solving them in the longer run. These results make us think that the TimeMapper system could be the starting point for a complete animated mapping system within a SDI context.

Our objective to design an animated mapping system which can generate SVG SMIL animations, automatically and directly, from spatio-temporal data, has thus been reached. Although our prototype only uses one specific data set, the employment of OGC standards and services in the system does make us confident that any other SDI-type of data could be used directly. It also does work automatically, albeit in the more narrow meaning explained in section 1. The broader sense of really fully automatic mapping from data, with cartographic design decisions included, still remains as an interesting research challenge that we hope to work on in the future. . .

References

1. Andrienko, N., Andrienko, G.: Exploratory analysis of spatial and temporal data: a systematic approach. Springer Verlag, Berlin, etc. (2006)
2. Andrienko, N., Andrienko, G., Gatalsky, P.: Exploratory spatio–temporal visualization: an analytical review. Journal of Visual Languages and Computing 14, 503–541 (2003)
3. Becker, T.: Visualizing Time Series Data Using Web Map Service Time Dimension and SVG Interactive Animation. Msc thesis, ITC, Enschede (2009)
4. Benn, D.I., Warren, C.R., Mottram, R.H.: Calving processes and the dynamics of calving glaciers. Earth-Science Reviews 82(3-4), 143–179 (2007)

5. Blok, C.A.: Dynamic visualization variables in animation to support monitoring of spatial phenomena. No. 328 in Nederlandse Geografische Studies, KNAG/Universiteit Utrecht, Utrecht, Enschede (2005)
6. de By, R.A., Lemmens, R., Morales, J.: A skeleton design theory for spatial data infrastructure. Earth Science Informatics 2(4), 299–313 (2009)
7. Fowler, H., Fowler, F., Sykes, J. (eds.): Concise Oxford Dictionary of Current English. Clarendon Press, Oxford, 6th edn. (1976)
8. Harrower, M., Fabrikant., S.: The role of map animation for geographic visualization. In: Dodge, M. (ed.) Geographic Visualization: Concepts, Tools and Applications., pp. 44–66. John Wiley and Sons (2008)
9. ITC: RIMapper project pages, `http://kartoweb.itc.nl/rimapper/`
10. ITC: TimeMapper project pages, `http://geoserver.itc.nl/timemapper/`
11. Köbben, B.: RIMapperWMS: a Web Map Service providing SVG maps with a built-in client. In: Fabrikant, S., Wachowicz, M. (eds.) The European Information Society - Leading the way with Geo-information, pp. 217–230. Lecture Notes in Geoinformation and Cartography, Springer-Verlag, Berlin, etc. (2007)
12. Köbben, B., de By, R., Foerster, T., Huisman, O., Lemmens, R., Morales, J.: Using the SDIlight approach in teaching a geoinformatics master. Transactions in GIS 14(s1), 25–37 (2010)
13. Kraak, M.J., Edsall, R., MacEachren, A.M.: Cartographic animation and legends for temporal maps: exploration and/or interaction. In: Proceedings of the 18th ICA International cartographic conference. vol. 1, p. 8. International Cartographic Association, Stockholm, Sweden (1997)
14. National Ice Center: Antarctic icebergs, `http://www.natice.noaa.gov/products/iceberg/`
15. OGC: OpenGIS Simple Features Specification for SQL Revision 1.1. Tech. Rep. OGC 99-049, Open Geospatial Consortium (1999)
16. OGC: Web Map Service Implementation Specification (1.1.1). Tech. Rep. OGC 01-068r3, Open Geospatial Consortium (2002)
17. OGC: Web Map Server Implementation Specification 1.3.0. Tech. Rep. 06-042, Open Geospatial Consortium (2006)
18. Schmittner, A., Yoshimori, M., Weaver, A.J.: Instability of glacial climate in a model of the ocean-atmosphere-cryosphere system. Science Express 295(5559), 1489 –1493 (2002)
19. Simpson, J., Weiner, E. (eds.): Oxford English Dictionary. Clarendon Press, Oxford, 2nd edn. (1989)
20. World Wide Web Consortium (W3C): Scalable Vector Graphics (SVG) 1.1 Specification, `http://www.w3.org/TR/SVG11/`