Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2016) 24: 683 – 694

doi:10.3906/elk-1304-166

A distributed map animation framework for spatiotemporal datasets

Ahmet SAYAR*
Department of Computer Engineering, Faculty of Engineering, Kocaeli University, İzmit, Turkey

Received: 17.04.2013 • **Accepted/Published Online:** 16.01.2014 • **Final Version:** 05.02.2016

Abstract: Maps are an excellent way to present data that have spatial components. However, when the data being presented vary over time, a simple two-dimensional map ignores an important feature of the data. An animated map that shows a series of two-dimensional maps at successive points in time allows one to add a time dimension to the display of data. The current study proposes a distributed service-oriented architecture to create map animations from spatiotemporal datasets. We extend the open standards GIS web services definitions with a topic-based publish-subscribe paradigm, which best suits the animation requirements. The effectiveness of the technique is demonstrated in an exploratory data analysis of Turkey's earthquake seismic data records at the end of the paper.

Key words: Distributed systems, Geographic Information Systems (GIS), web services, animation, animated map, spatiotemporal data

1. Introduction

Vast amounts of data related to the earth are time-series and spatial in nature. The geological studies are mostly based on spatial data analysis. To understand geographical phenomena it is important to show the patterns changing over time. Spatial data are preferably represented and displayed as map layers. When you have the same layer in several different moments in time, it is better to display them as part of a movie. This is called time-series animation. It is a visualization technique ideally suited for the display and analysis of spatiotemporal and geographic datasets. Map animations enable scientific analyses and results to be understood not only by the scientist, but also by the public and policymakers from different domains and education levels. Animated maps can be interpreted more easily than their static representations.

Interoperability and distributed services are clear trends in today's Geographic Information Systems (GIS) [1]. Standards for interoperability proposed by distributed frameworks such as the Open Geospatial Consortium (OGC) offer advantages for data sharing, for combining software components, and for overlaying graphical outputs from different sources. The standardization efforts make distributed services widely accepted and used in many areas such as governmental agencies and educational institutions. However, standardization comes with costs.

Developing OGC-compatible GIS services (Web Map Service (WMS) [2] and Web Feature Services (WFS) [3]) as web services enables them to be discoverable and used in third-party distributed systems [4]. However, efficient data transportation capability still remains a challenge because web services are based on XML-based SOAP over HTTP protocol, while data to be processed are encoded in Geographic Markup language (GML) [5], which is an XML-based format. To overcome such problems in a distributed system framework requiring

*Correspondence: ahmet.sayar@kocaeli.edu.tr

large-scale XML-encoded geographic feature sets, we investigated the possibilities of using topic-based publish-subscribe paradigms (which are mostly used in P2P systems) for exchanging data payloads between web services. NaradaBrokering [6] is one of the well-known applications of that approach, enabling streaming data transport, reliable delivery, and recovery from network failures at the application level. NaradaBrokering is integrated with the system through a two-step communication protocol in which standard web service interfaces are used as a handshake protocol and the actual data are transferred over a publish-subscribe-based messaging system. This approach has some advantages over pure web services; it gets rid of the SOAP message creation overheads and even enables the creation of map images with partially returned data.

After developing an efficient data transfer protocol between standard GIS web services, we propose an animation web service extended from WMS. WMS has capabilities of animating temporally related map images one by one, dated in a vertical frame as a film. Images are created from the spatiotemporal data provided by WFS. The effectiveness of the technique is demonstrated in an exploratory data analysis of Turkey's earthquake seismic data records at the end of the paper.

The remainder of this paper is organized as follows: in Section 2 we present related works; Section 3 presents distributed service-oriented architecture for map animations; and Section 4 concludes the paper.

2. Related work

In the last decade renewed interest in animation has emerged due to technological developments. Because of these developments currently it is relatively easy and inexpensive to construct animations. This has led to an increase in the number, variety, and complexity of animations produced. One of the areas where animations have been successfully applied is map animations.

Map animations have been studied by various disciplines such as computer sciences, remote sensing, and pattern recognition. Most of the applications are central (i.e. desktop): data and services are physically located in the same machine and the analyses are carried out in the same place. Among these studies, [7] studied extracting the spatial pattern in time by visualization as an animation, [8,9] presented the technical aspects of the animated maps such as challenges and efficiency issues, and [10,11] presented case studies with spatiotemporal phenomena. These early works were on recognition of temporal changes in spatial datasets through animation, while our focus is creating animating web services that enable sharing and collaboration of animated spatial data among virtual organizations through distributed systems.

With the advent of the Internet and advancements in distributed systems great dissemination possibilities have opened up; it has become easier to access data and processing services, and to couple them for application purposes. In the context of distributed system applications, map animations can be grouped into two types: client-centric and server-centric. MathWorks' mapping toolbox and GeoServer's animator toolbox can be given as examples of client-centric approaches. They connect to a remote WMS (OGC-compatible) and fetch the map images to create an animation. They build animations as a set of frames, and each frame is a separate WMS getMap call, similar to the others in the set, but with a different value in one of the parameters. Map images are stored in a local file system at the client's side, and animations are created as animated GIF or movies in AVI format. There are also a few related works on server-centric map animations. Köbben [12], Becker [13], and Esri's ArcGIS claim they are OGC-compatible, but they did not develop their services in accordance with the service-oriented architectures. Furthermore, they did not consider the issues of large-scale data transfers over the network for real-time animations. The animations they produce are mostly in the form of a moving window (bounding box) or zooming in and out.

The work presented in this paper is a server-centric approach. The proposed system not only supports zooming or moving animations, but also animates spatiotemporal changes in large-scale feature collections for a specific bounding box. To do that, we take data rendering and overlaying issues into consideration. In other words, spatiotemporal feature data collections are overlaid on satellite base maps. In the proposed system the output is streaming data published through real-time protocol (RTP). This enables animations to be played in collaborative and Grid [14] environments.

3. Architecture: streaming map movies

This paper proposes a distributed service-oriented architectural framework for binding map-based geodata animations to the distributed services by adopting and implementing OGC's WFS and WMS services in accordance with publicly available standards. OGC also has a discussion paper [15] on animations as an extension to WMS. It discusses how WMS specifications can be extended to allow animations that move in space over time. In this case, the only parameter changing is the bounding box in successive map images. We focus on overlay layers that are rendered from feature data collections represented as GML and provided by WFS. We take both geometrical and nongeometrical attributes of spatiotemporal datasets into account and animate their changing values over time. The map movies and animations are used interchangeably throughout the document. The map movies presented in this paper are similar to the ones shown in the weather news.

The work presented here looks into the possibilities of extending the web-based map services with time-series data as animated maps and provides a framework that enables the integration of animation services with distributed systems. For service-level interoperability (in terms of request and response types) and definition of animated layer descriptions we use OGC defined standards [2,3]; for data transfer we propose a novel approach based on integration of OGC specifications with web services [14] and topic-based publish-subscribe paradigms.

This section first explains the system components and their interactions to create a simple map image, and then elaborates on WMS and its capabilities of being an animation service. Finally, it presents a distributed architecture enabling the creation of map movies from time-series geographic data.

3.1. System components to create a map

The architecture is composed of distributed service components (Figure 1). These are basically WMS, WFS, and NaradaBrokering. NaradaBrokering is a data exchange protocol that enables topic-based publish-subscribe data transfers. We adopt and implement OGC's WFS and WMS as distributed web services [14]. Our earlier works on these are presented in [16–18] and [17], respectively.

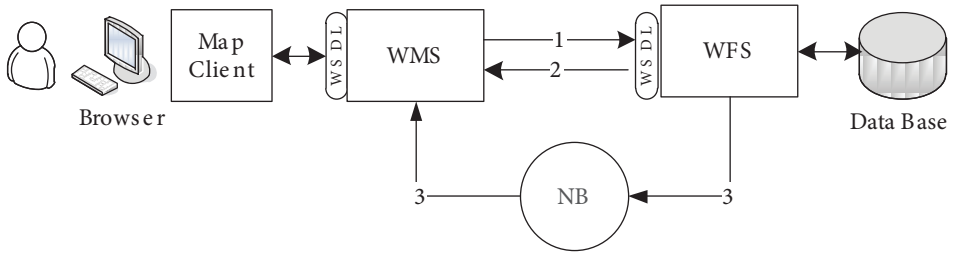

Figure 1. System components.

Like all computer networking standards, WMS standards specify how the interfaces of services should be defined. Service developers remain free to store or process data according to methods that suit their internal

needs and resources. Conforming to the WMS specifications enables the automatic overlaying of maps obtained from several different map servers, regardless of projection, earth coordinate system, scale, or digital format. Since the outcome is a digital image, it can be shown on an ordinary browser. Furthermore, one or more of these map images may well be the result of complex geoprocessing. This leads to important time and financial gains.

There are three basic types of requests with a WMS implementation: the GetCapabilities request, the GetMap request, and the GetFeatureInfo request. The first two are compulsory for OGC compliance and the last is optional. The GetCapabilities request is used by the clients to ask for the capabilities of the service to create valid successive queries such as GetMap and GetFeatureInfo. The service capabilities are about the available layers, the projection system available for the layers, supported output formats, etc. GetMap is issued to ask for an actual map. GetFeatureInfo is used to determine attribute values in the underlying data. [19] and [17] present the implementation details of WMS through some applications.

WFS allows clients (mostly WMS) to access/manipulate spatial data (called features) through its standard service interfaces. It does that by hiding the heterogeneity of the types and interfaces of the local data stores (file systems or databases). WFS accepts queries that are encoded in OGC's Common Query Language. These are GetCapabilities, DescribeFeatureType, and GetFeature. GetCapabilities queries the WFS service to determine available options to create successive valid queries. DescribeFeatureType retrieves the XML schema to allow the WFS client to parse the result sets. GetFeature performs the actual query; parameters such as bounding box and any other filters should be passed in, as appropriate, and the WFS service then returns a GML result set containing full geometry and feature attributes.

WFS defines the standard communication protocol as HTTP Get/Post methods and it has some limitations when they are used in scientific computations requiring processing and transferring of large scale datasets. Furthermore, we sometimes need to use WFS in coordination with other web services in distributed applications. That requires WFS to be discovered and coupled with other services easily. Web service technologies help us get over these problems. Our web service-based WFS application is used and tested with distributed data analysis tools in the Pattern Informatics [20] application, and [17] shows our initial works and the performance analyses.

Developing WMS and WFS as web services enables them to be discoverable and used in third-party distributed systems. However, efficient data transportation capability still remains a challenge because web services are based on XML-based SOAP over HTTP protocol. In order to overcome such problems in the proposed distributed system framework requiring large-scale XML-encoded geographic feature sets, we investigated the possibilities of using topic-based publish-subscribe paradigms (mostly used in P2P systems) for exchanging data payloads between web services. Figure 1 illustrates how the proposed architecture works to produce a map image from geographic features. WFS using NaradaBrokering is called streaming WFS. When the NaradaBrokering is used, WFS is still queried with standard SOAP messages (requests) (arrow 1 in Figure 1). However, the responses are published (i.e. streamed) to an NB topic as they become available (arrow 3 in Figure 1). Arrow 2 shows the subscription stage. WFS sends the "IP" and "topic" to which the results will be streamed. The clients (WMS) subscribed to the same topic can receive the streams. WFS uses MySQL in the background and streams the results row by row (consider the relational tables) instead of waiting for the calculation of the result set to be completed. The streaming enables the I/O and CPU tasks to be overlapped and ends up with performance gains. The performance results are presented in our earlier work [17].

The final outcome of the system (Figure 1) is a map image. WMS clients request that image through the WMS getMap service interface. Once WMS gets this request, it creates corresponding getFeature requests to

WFS to get the feature data in GML format. After getting the data, WMS checks and extracts all the geometry elements such as points, line-strings, polygons, etc., and converts them into appropriate image formats. WMS developers can use any kind of graphics tools to create map images from those geometric features of data.

3.2. WMS as an animation server

We think that WMS is presently the most suited candidate for building a map server that provides animated maps in accordance with domain standards and web service standards. Although the standardization for animations is not mature yet [15], WMS specifications offer an appropriate standard for the sharing of data containing a temporal dimension.

WMS creates static maps in pictorial formats from geodata provided by WFS and stores them in memory as an image array. Map animation is basically showing those images dated in a vertical frame one by one as a movie. If WMS is capable of providing animation services for a layer (spatial datasets), then some attributes can be added under this layer definition in its capabilities file. These attributes are defined in the dimension tag (" < Dimension name = "time" > ") (details are given below). This tag is defined in WMS standards. WMS has the same interface for providing both static map images and map movies. For animations clients use GetMap services with a different set of parameters. The schema and an example of a GetMap request are illustrated in Figure 2 and Figure 3, respectively. See how "format" and "time" variables are set to obtain map movies instead of static still map images.

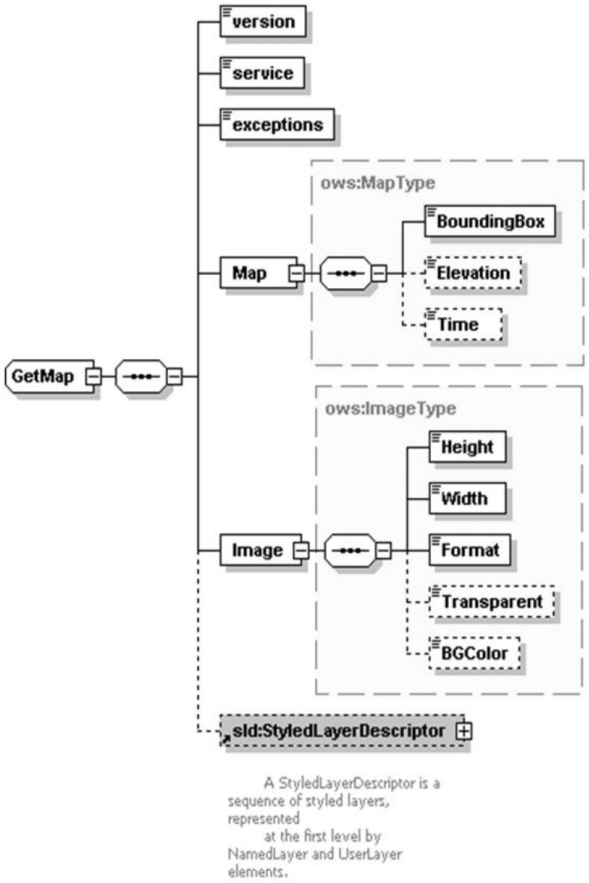

Figure 2. GetMap Request schema to create streaming map movies.

```
<?xml version="1.0" encoding="UTF-8"?>
<GetMap xmlns="http://www.opengis.net/ows">
    <version>1.1.1</version>
    <service>wms</service>
    <exceptions>application_vnd_ogc_se_xml</exceptions>
    <Map>
        <BoundingBox ts="">-124.85,32.26,-113.56,42.75</BoundingBox>
        <Elevation>5.0</Elevation>
        <Time>01-01-1987/12-31-1992/P1Y</Time>
    </Map>
    <Image>
        <Height>400</Height>
        <Width>400</Width>
        <Format>video/mpeg</Format>
        <Transparent>true</Transparent>
        <BGColor>0xFFFFFF</BGColor>
    </Image>
    <ns1:StyledLayerDescriptor version="1.0.20" xmlns:ns1="http://www.opet/sld">
        <ns1:NamedLayer>
            <ns1:Name>Nasa:Satellite</ns1:Name>
            <ns1:Description>
                <ns1:Title>Nasa:Satellite</ns1:Title>
                <ns1:Abstract>Nasa:Satellite</ns1:Abstract>
            </ns1:Description>
        </ns1:NamedLayer>
        ....
        <ns1:NamedLayer>
            <ns1:Name>World:Seismic</ns1:Name>
            <ns1:Description>
                <ns1:Title>World:Seismic</ns1:Title>
                <ns1:Abstract>World:Seismic</ns1:Abstract>
            </ns1:Description>
        </ns1:NamedLayer>
    </ns1:StyledLayerDescriptor>
</GetMap>
```

Figure 3. A sample GetMap request for WMS to create streaming map movies.

Since we developed WMS as a web service, we created a schema for getMap requests in accordance with the standard URL definitions. These requests are inserted into SOAP messages to invoke animation web services. Figure 2 shows the standard schema for getMap requests and some of the standard parameters and their possible values. Figure 3 is an instance of the standard getMap request created over the schema (see Figure 2).

According to the standards, queries for multidimensional data objects are described with the < Dimension > tag. Time is one of the dimension names defined in the WMS capabilities file. If the time dimension is defined for a layer, then clients can make requests for specific time intervals and periodicity values to create movies. The number of requests to WFS to get feature data for the specific time intervals and the number of frames for the movie change according to the parameter values given in the time parameter in the getMap request (Figure 3). For example, if a WMS provides a time-series data layer listed in its capabilities file, it puts a time dimension element as displayed below under the specific layer tag.

< Dimension name = "time" units = "ISO8601" default = "2000-08-22" >

2001-04-10/2010-10-11/P1Y

< / Dimension >

One layer might be available in multiple disjoint time intervals and those intervals might have different periodicities. At that time, WMS adds additional lines to the time dimension element as displayed below.

< Dimension name = "time" units = "ISO8601" default = "2000-08-22" >
2001-04-10/2010-10-11/P1D
1990-01-01/1998-08-22/P1Y
< / Dimension >

In the parameter values given as examples above, the first date defines the starting date and the second date defines the end date of the available data collection. The last value (ex. "P1D") defines the periodicity of data collection. According to the last value in the time parameter, WMS cuts the time into multiple values and for each time interval it makes a request to WFS to get feature data in GML. See the successive requests (req_i) in Figure 4; gml_i represents corresponding responses from WFS to the requests req_i; img_i are images created from corresponding gml_i.

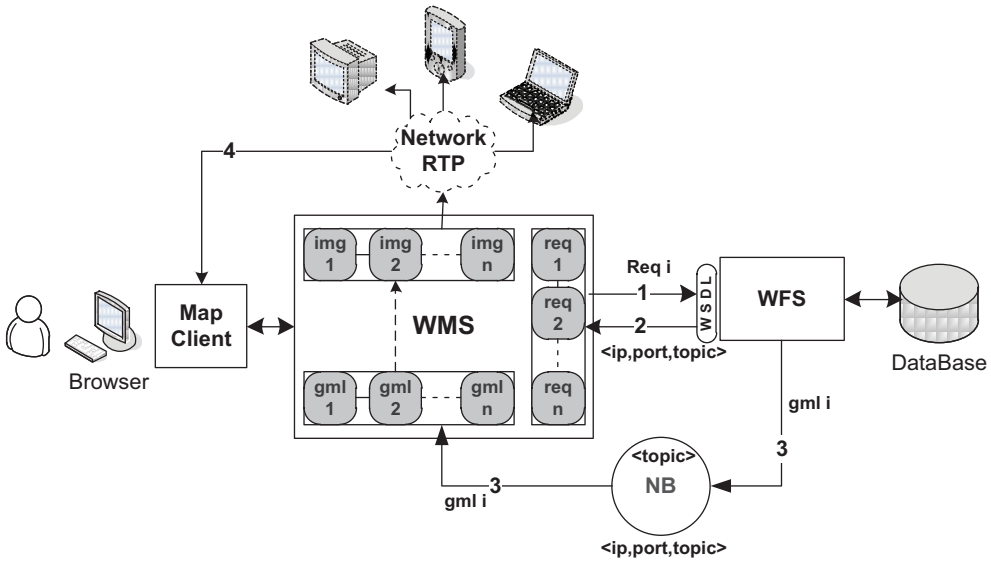

Figure 4. Detailed streaming map movies architecture.

User interaction with the system is achieved through browser-based WMS clients (Figure 5). MapClient predefines the animation format in a particular style and defines in what date ranges and in what time slices the animation is needed. The appropriate query is created (Figure 3) and sent to WMS thorough its getMap web service interface. Once WMS gets this query, it creates successive queries (req_1, req_2, ..., req_n) based on the time parameter in the getMap request. Each of those queries is responded to with gml (gml_1, gml_2, ..., gml_n). For each gml a still map is created. Every still map corresponding to a time slice is stored in memory as part of an image-array and displayed in a vertical subwindow simulating a camera film.

3.3. Publishing animated map images

WMS (see Section 3.2) is an access point for distributed systems or any other clients to use map animation services. The first step in an animation is the creation of a series of temporally related successive map images. The second step is playing these still map images as an animation. The proposed architecture is presented in Figure 4.

WMS needs successive map images for a period of time in order to be able to create an animation. The time period and the periodicity of the movie frames are defined by a parameter called "time" in the GetMap

request (Figure 3). The number of frames to be played depends on the time period and periodicity of the time intervals. To assemble the individual frames into an animation, different approaches can be used according to application requirements. In the Internet world a widely used and well-known approach is using animated GIF89a files. The browser will be enough to open the animation, but the users have no control of the image such as stopping, pausing, or playing back the animation. The animated GIF will play only in one direction. There is no way to make it play backwards. PROC GMAP is an example of GIF animations. The Java Media Framework (JMF) and Quicktime provide other approaches to assemble individual frames into an animation. JMF uses RTP sessions and needs a Java virtual machine installed on the client machines. That is used on Java-based applications. OGC standards do not specifically and clearly define how to transfer and display animation. They only specify the interfaces in terms of standard queries and output formats. In the proposed framework, we preferred the IP multicast approach with JMF technologies.

An animation is produced as video streams. Map images are converted into a sequence of video streams and published in a RTP session [21]. RTP sessions are formalized as < IPAddress, PortNumber > pairs. There are various video stream formats. The framework uses H.263 and H.261, which are well-known and widely used formats. Those animated video streams can be played and displayed on video-conferencing systems and collaborative environments such as AccessGrid (http://www.accessgrid.org/), which are supposed to support H.263 and H.261 formats. The produced streams are published in multicast or unicast RTP sessions. Video streams can be delivered to a variety of platforms such as RealPlayer, Polycom, and Access Grid [22]. The published video streams can also be displayed by any client building his own custom system and services to display the map video streams. The easiest way to display the map movie stream is connecting to RTP sessions by using a JMF Client.

The quality of the streams depends on some configurable parameters such as video format, frame rate, and update rate. These parameters are set at the creation time, depending on the data and the application specific requirements.

Use case scenario – animating earthquake seismic records:

Figure 5 shows the user interface (which is actually a map client in Figure 4) for animating time-series datasets. The images for the use case scenario animation are 2-layer map images. The first layer (base-map) is Landsat imagery of Turkey, provided by the NASA OnEarth project (http://onearth.jpl.nasa.gov/). The second layer is Turkey's earthquake seismic data recorded from 1992 to 2005. Those feature datasets are kept in a database and served to the system through standard WFS web service interfaces. The base map is held still and the action played out upon it is obtained by animating the second layer. Earthquake seismic data have some major attributes such as magnitude, location (x and y coordinates), and date/time, and some other minor attributes. Queries to create maps from those datasets are done based on those attributes. There are also some other parameters that need to be set during the publication of the contents of the streams. Frames (map images) are updated at every 0.3 s, and the frame rate is 5 frames/s. From our experience, this parameter set for configuration is sufficient, but they can be changed according to the application requirements. Depending on the underlying network and the characteristics of data and servers used, the parameter values can be changed to get the best results. This also explains why update and frame rates are different.

Let us call the table containing the earthquake seismic data Earthquake-Seismic. It has the following attributes: X: longitude values of the seismicity of earth; Y: latitude values of the seismicity of earth; Magnitude: the power of seismicity changing from 0 to 10; Year: in which seismic event occurred; Month: in which seismic event occurred; Day: on which seismic event occurred.

An SQL query to get seismic datasets whose magnitude values are between 7 and 10 and that occurred in the bounding box (–124.85, 32.26, –113.36, 42.75) is given below.

Select Latitude, Longitude, Magnitude from Earthquake-Seismic where –124.85 < X < –113.36 & 32.26 < Y < 42.75 & 7 < Magnitude < 10

However, the data are provided with a standard service interface (called getFeature) in a standard format (GML) by WFS. Both query (getFeature) and response (GML) are in the form of XML; they are created according to the standard schema. A small part of the standard getFeature query corresponding to the above SQL is given below (showing just the magnitude constraints).

```
< wfs:GetFeature outputFormat = “GML2” xmlns:gml = ”http://www.opengis.net/gml >
    ……..
    < wfs:Query typeName = "global_hotspots" >
        < ogc:Filter >
            < ogc:PropertyIsBetween >
                < ogc:Literal > Magnitude < / ogc:Literal >
                < ogc:LowerBoundary >
                    < ogc:Literal > 7 < / ogc:Literal >
                < / ogc:LowerBoundary >
                < ogc:UpperBoundary >
                    < ogc:Literal > 10 < / ogc:Literal >
                < / ogc:UpperBoundary >
            < / ogc:PropertyIsBetween >
        < / ogc:Filter >
    < / wfs:Query >
    ……
< / wfs:GetFeature >
```

The response to getFeature query will be GML. It carries geometrical points (x, y coordinates) and nongeometrical attributes (magnitudes) of seismic data. WMS renders the GML and overlays it on the base map. This image is going to be a frame in the movie stream.

By applying the architecture on real-world earthquake seismic data records, we investigated the seismicity characteristics of the selected regions and got the answers to the following questions:

- How often do earthquakes occur?
- Which regions have the higher possibilities of facing an earthquake?
- Where are the safest places in terms of earthquakes?
- Is there any pattern of earthquake events?
- What are the periodicities of earthquake events of magnitudes greater than a certain value?

Different queries need different sets of parameters. These parameters can be given through the user interfaces shown in Figure 5. Map tools enable users to choose a specific region, starting and ending dates for animation (time interval), and the periodicity defining the number of frames for a specific time interval.

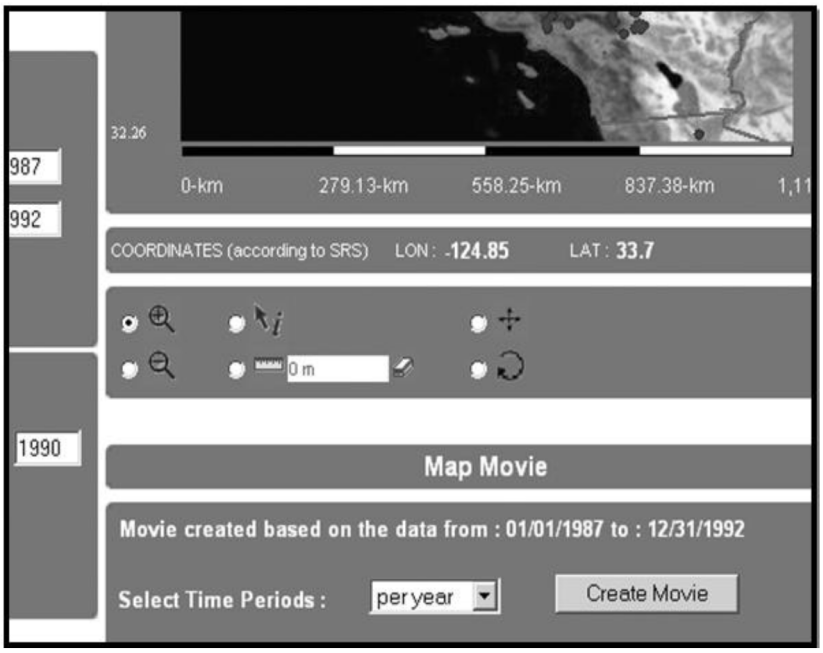

Figure 5. User interface (WMS Client) for streaming map movies.

Figure 6 is a JMF display tool called JMF studio. It subscribes to RTP sessions and displays the movie streams published by JMF. It might be deployed on any machine running on any platform as long as the machine has a Java virtual machine installed on it. As shown in Figure 6, the light boxes represent the earthquake seismic data records with the lowest magnitude values, and the dark boxes show the earthquake seismic records with the highest magnitude values for a specific date. Coloring is done according to the magnitude values of the seismic records.

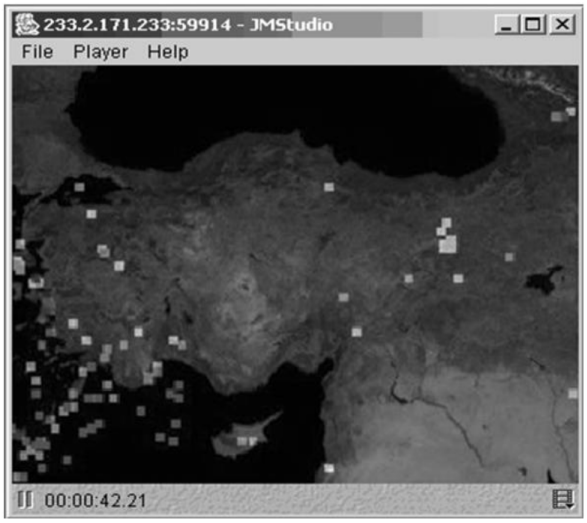

Figure 6. Displaying a map movie with JMF Studio.

4. Conclusion and future work

The work presented here examined the possibilities of extending the web-based map services with time-series data as animated maps and introduced a framework enabling the integration of animation services with distributed systems. WMS is at the core of this framework. It is actually an access point for distributed systems to use map animation services. We have also extended the standard GIS web service communications with the topic-based publish-subscribe communication approach. In this framework, GIS web services use standard interfaces for the handshake; the actual data are transferred over the P2P overlay network provided by NaradaBrokering. After the handshake, communicating peers start transferring the data through the agreed upon broker (IP) and topic (any string). This approach enables us to create map images for partially returned data and get rid of the SOAP message creation overheads.

The proposed architecture enables the exploratory data analysis of spatiotemporal datasets. By overlaying Turkey's earthquake seismic data records on satellite map images and playing them successively with predefined time intervals, we were able to determine the characteristics of earthquake events for specific regions on the map. The proposed framework and the approach have proven effective for such purposes. However, the framework needs to be enhanced with some quality of services. The proposed framework is developed with open standards and Java technologies. Therefore, it can easily be enhanced and extended for application specific purposes, deployed on any platform, and integrated to third party distributed system applications.

In the proposed system, movie streams (for map animations) are created on demand from the archived datasets. In the future we plan to enhance the system with the capability of archiving map animations. In that case, each archived map animation needs to be annotated with some parameters enabling it to be searched. These parameters might be "temporal data layers from which movie streams are created", "frame rates", "starting-ending times/dates of the animation", and "periodicity of the data frames".

References

[1] Peng ZR, Tsou MH. Internet GIS: Distributed Geographic Information Services for the Internet and Wireless Networks. Hoboken, NJ, USA: Wiley, 2003.

[2] Beaujardiere JDL. OGC Web Map Service Interface. Vol. 06-042. Open GIS Consortium Inc. (OGC) Specification, 2006.

[3] Vretanos PA. Web Feature Service 2.0 Interface Standard. Vol. 09-025r1 and ISO/DIS 19142. Open Geospatial Consortium Inc. (OGC) Specification, 2010.

[4] Tanenbaum AS, Steen MV. Distributed Systems: Principles and Paradigms. 2nd ed. Upper Saddle River, NJ, USA: Prentice Hall, 2006.

[5] Cox S, Daisey P, Lake R, Portele C, Whiteside A. OpenGIS®Geography Markup Language (GML) Encoding Specification. Vol. 02-023r4. Open Geospatial Consortium (OGC), 2003.

[6] Pallickara S, Fox G. NaradaBrokering. A Distributed Middleware Framework and Architecture for Enabling Durable Peer-to-Peer Grids. In: Middleware'03 Proceedings; 16–20 June 2003; Rio Janeiro, Brazil. New York, NY, USA: ACM. pp. 41–61.

[7] Blok C, Kobben B, Cheng T, Kuterema AA. Visualization of relationships between spatial patterns in time by cartographic animation. Cartogr Geogr Inform 1999; 26: 139–151.

[8] Peterson MP. Interactive and Animated Cartography. Englewood Cliffs, NJ, USA: Prentice Hall, 1995.

[9] Harrower M. Tips for designing effective animated maps. Cartographic Perspectives 2003; 44: 63–65.

[10] Harrower M. Visualizing change: using cartographic animation to explore remotely-sensed data. Cartographic Perspectives 2002; 39: 30–42.

[11] Frihida A, Marceau DJ, Theriault M. Extracting and visualizing individual space-time paths: an integration of GIS and KDD in transport demand modeling. Cartogr Geogr Inform 2004; 31: 30–42.

[12] Köbben B. SVG and Geo Web Services for visualization of time series data of flood risk. In: SVG Open'08; 26–28 August, 2008; Nürnberg, Germany. pp. 9–12.

[13] Becker T. Visualizing time series data using web map service time dimension and SVG interactive animation. MSc, University of Twente, Enschede, the Netherlands, 2009.

[14] Atkinson M, DeRoure D, Dunlop A, Fox G, Henderson P, Hey T, Paton N, Newhouse S, Parastatidis S, Trefethen A et al. Web service grids: an evolutionary approach. Concurr Comp-Pract E 2005; 17: 377–389.

[15] LaMar E. Proposed Animation Service Extension. Vol. 06-045r1, p. 23. Open Geospatial Consortium Inc. (OGC) Specification, 2005.

[16] Pierce ME, Fox GC, Aktas MS, Aydin G, Qi Z, Sayar, A. The QuakeSim Project: web services for managing geophysical data and applications. Pure Appl Geophys 2008; 165: 635–651.

[17] Aydin G, Sayar A, Gadgil H, Aktas MS, Fox GC, Ko S, Bulut H, Pierce ME. Building and applying geographical information systems grids. Concurr Comp-Pract E 2008; 20: 1653–1695.

[18] Aktas M, Aydin G, Donnellan A, Fox G, Granat R, Grant L, Lyzenga G, McLeod D, Pallickara S, Parker J et al. iSERVO: Implementing the international solid Earth research virtual observatory by integrating computational grid and geographical information web services. Pure Appl Geophys 2006; 163: 2281–2296.

[19] Sayar A, Pierce M, Fox G. Developing GIS visualization web services for geophysical applications. In: ISPRS Spatial Data Mining Workshop, Commission II WD/2; 24–25 November 2005; Middle East Technical University. Ankara, Turkey: ISPRS. pp. 21–28.

[20] Tiampo KF, Rundle JB, McGinnis SA, Klein W. Pattern dynamics and forecast methods in seismically active regions. Pure Appl Geophys 2002; 159: 2429–2467.

[21] Schulzrinne H, Casner SL, Jacopson FV. RTP: A Transport Protocol for Real-Time Applications. Internet Standard Specifications (RFC 3550), 2003.

[22] Wu W, Fox GC, Bulut H, Uyar A, Altay H. Design and implementation of a collaboration web-service system. Neural, Parallel & Scientific Computations 2004; 12: 301–496.