Distributed Map Animation Services for Spatiotemporal Datasets

A. Sayar

Key Words: *Distributed Systems; animation; GIS; animated map; spatiotemporal data.*

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 study presented in this paper proposes a distributed service-oriented architecture to create map animations from spatiotemporal datasets. We extend open standards' GIS web services definitions with topic-based publish-subscribe paradigm, which best suits to the animation requirements. The effectiveness of the technique is demonstrated on exploratory data analysis on Turkey's earthquake seismic data records at the end of the paper.*

1. Inroduction

Vast amounts of data related to 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 one has the same layer in several different moments along the time, it is better to display them as part of a movie. This is called time-series animation, which is a visualization technique ideally suited for the display and analysis of spatiotemporal and geographic data sets. Map animations enable scientific analysis and results to be understood not only by the scientist but also the public and policymakers from different domains and education level. Animated maps can be interpreted more easily than their static representations by the users.

Interoperability and distributed services are clear trends that today's Geographic Information Systems (GIS) [1] is taking. Standards for interoperability proposed by distributed frameworks such as the Open Geospatial Consortium (OGC) [2] offer advantages for data sharing, for combining software components and for overlaying graphical outputs from different sources. The standardization efforts cause distributed services to be widely accepted and used in many areas such as governmental agencies and educational institutions. However, standardization comes with its cost.

Developing OGC compatible GIS services (WMS [3] and WFS [4]) as web services enables them to be discoverable and used in third party distributed systems [5]. However, efficient data transportation capability still remains as a challenge, because of the fact that web services are based on XML based SOAP over HTTP protocol and data to be processed are encoded in Geographic Markup language (GML) [6] which is an XML-based format. To overcome such problems in a distributed system framework requiring large scale XML-encoded geographic feature sets, we have investigated the possibilities of using topic-based publish-subscribe paradigms (which is mostly used in P2P systems) for exchanging data payload between web services. NaradaBrokering [7] 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 to the system through two-step communication protocol in which standard web service interfaces are used as a handshake protocol and the actual data is transferred over publish-subscribe based messaging system. This approach has some advantages over pure web services. The proposed system gets rid of the SOAP message creation overheads, and even enables 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 have 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 on exploratory data analysis on 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 the related works. Section 3 presents distributed service oriented architecture for map animations. 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 it is nowadays 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 that 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) in which data and services are physically located in the same machine and the analyses are carried out in the same place. Among these studies, [8] studies extracting the spatial pattern in time by visualization as an animation, [9] and [10] present the technical aspects of the animated maps such as challenges and efficiency issues, [11] and [12] present their case studies with spatiotemporal phenomena. These early works are on recognition of temporal changes in spatial datasets through animation, but our focus is creating animating web services enabling sharing and collaboration of animated spatial data among the virtual organizations through the distributed systems.

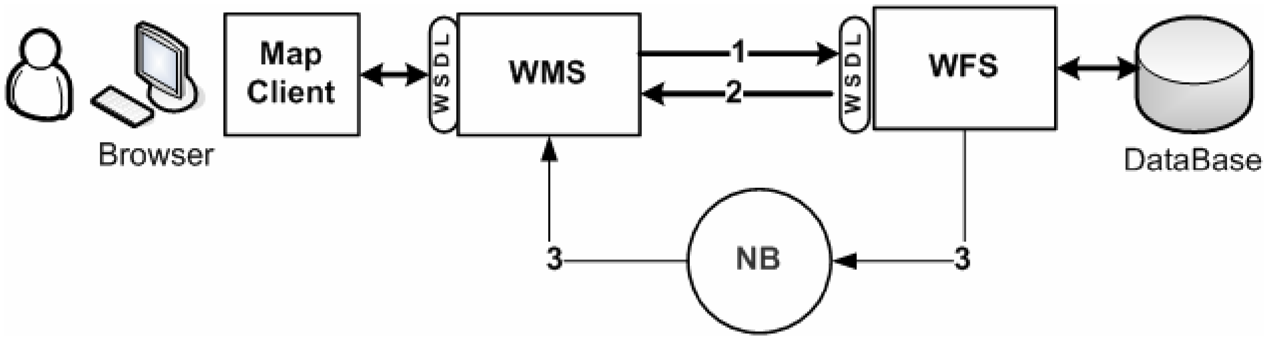


Figure 1. System Components

After the invent of the internet and advancements in distributed systems there has been great dissemination possibilities, and it has become easier to access data and processing services and coupling them for the application purposes. In the context of distributed system applications, map animations can be grouped into two, these are client-centric and server-centric. MathWorks's mapping toolbox [13] and GeoServer's animator toolbox [14] can be given as examples of client-centric approaches. They connect to 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 into local file system at client 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. Kцbben [15], Becker [16] and Esri's ArcGIS [17] claim they are OGC compatible, but they do not develop their services in accordance with the service oriented architectures. Furthermore, they do not consider the issues of large scale data transfer over the network for the real time animations. The animations they produce are mostly in the form of 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 considerations. In other words, spatiotemporal feature data collections are overlaid on satellite base maps. In the proposed system the output is streaming data published through RTP protocol. This enables animations to be played in collaborative and Grid [18] environments.

3. Architecture: Streaming Map Movies

This paper proposes a distributed service oriented architectural framework for binding map based geo-data animations to the distributed services by adopting and implementing OGC's WFS and WMS services in accordance with the publicly available standards. OGC also has a discussion paper [19] 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. On the other hand, we focus on overlay layers which are rendered from feature data collections represented as GML and provided by WFS. We take both geometrical and non-geometrical attributes of spatiotemporal data sets into account and animate their changing values over time. We produce map animations in the form of streaming map videos similar to ones you see in the weather cast web sites.

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 enabling integration of animation services to the distributed systems. In the proposed framework, for the service level interoperability, in terms of request and response types, and definition of animated layer descriptions we use OGC defined standards [3, 4], but for the data transfer we propose a novel approach based on integration of OGC specifications with web services [18] 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, and finally presents a distributed architecture enabling creation of map movies from time-series geographic data. Figure 4 shows the illustration of the general architecture.

3.1. System Components to Create a Map

The architecture is composed of distributed service components (see *figure 1*). These are basically WMS, WFS and NaradaBrokering. NaradaBrokering is a data exchange protocol enabling topic-based publish-subscribe data transfer. We adopt and implement OGC's Web Feature Services (WFS) and Web Map Service (WMS) as distributed web services [18]. Our earlier works on these are presented in [20-22] and [21].

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 enable automatically overlay maps obtained from several different map servers, regardless of projection, earth coordinate system, scale or digital format. Since the outcome is digital image it can be shown on any ordinary browser. Furthermore, one or more of these map images may well be the result of complex geo-processing. This leads to important time and financial gains.

There are three basic types of requests that can be asked to WMS implementation. These are GetCapabilities request, 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. [23] and [21] preset the implementation details of WMS through some applications.

WFS allow 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 accept the queries which 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 resultsets. 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 resultset containing full geometry and feature attributes.

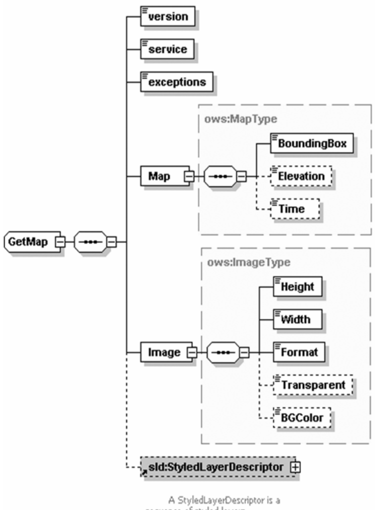
WFS define the standard communication protocol as HTTP Get/Post methods and it has some limitations when they are used in scientific computations requiring processing and transferring of large scale data sets. Furthermore, we sometimes need to use WFS in coordination with some other web services in distributed applications. That requires WFS to be discovered and coupled with the 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 Pattern Informatics [24] application, and [21] 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 as a challenge, because of the fact that web services are based on XML based SOAP over HTTP protocol. In order to overcome such problem in the proposed distributed system framework requiring large scale XML-encoded geographic feature sets, we have investigated the possibilities of using topic-based publish-subscribe paradigms (which is mostly used in P2P systems) for exchanging data payload between web services. *Figure 1* illustrates how the proposed architecture works to produce a map image from geographic features. WFS using NaradaBrokering are called streaming WFS. When the NaradaBrokering is used, WFS are 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 send the IP and topic to which the results will be streamed. The clients (WMS) subscribed to the same topic can receive the streams. WFS use MySQL in background and stream the results row by row (consider the relational tables) instead of waiting the calculation of the result set to be completed. The streaming enables I/O and CPU jobs to be overlapped and ends up with performance gains. The performance results are presented in our earlier work [21].

Final outcome of the system (*figure 1*) is a map image. WMS clients request that image through WMS's getMap service interface. Once WMS get this request, they create corresponding getFeature requests to WFS to get the feature data in GML (Geographic Markup Language) format. After getting the data, WMS check and extract all the geometry elements such as points, line-strings, polygons etc., and converts them into appropriate image formats. WMS developers 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 are presently the most suited candidate for building a map server providing with the animated maps in accordance with the domain standards and web service standards. Although the standardization for animations is not mature yet [19], WMS's specifications offer an appropriate standard for



Parameters and sample values
VERSION = 1.3.1
REQUEST = GetMap
LAYERS = earth
CRS = CRS:64
BBOX = -90,-45,90,45
WIDTH = 800
HEIGHT = 400
TIME=2008-6-01/2010-6-01/P1Y
FORMAT= video / mpeg

Figure 2. GetMap Request schema to be able 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. Sample GetMap request for WMS to create streaming map movies

the sharing of data containing a temporal dimension.

WMS create static maps in pictorial formats from geo-data provided by WFS and store 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 a WMS is capable of providing animation services for a layer (spatial data sets), then, some attributes are added under this layer definition in its capabilities file. These attributes are defined in dimension tag („<Dimension name="time">"), and details are given below. This tag is defined in WMS standards. WMS have same interface for both providing static map images and map movies. So, for animations, clients use GetMap services with different set of parameters. For example, clients set the „format" and „time" variables to appropriate formats as shown in *figure 3*.

Since we developed WMS as web services, 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 multi-dimensional data objects are described with <Dimension> tag. Time is one of the dimension name defined in WMS capabilities file. If time dimension is defined for a layer then clients can make requests for a specific time intervals and periodicity value to create movies. The number of requests to WFS to get feature data for the specific time intervals and number of frames for the movie change according to the parameter values given in time parameter in 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 add additional lines to 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, first date defines the starting date and second date defines the end date of the available data collection. The last value (ex. „P1D") defines the periodicity of data collection. According to last value in parameter time, WMS cut 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 represent corresponding responses from WFS for the requests req_i. img_i are images created from corresponding gml_i.

User interaction with the system is achieved through browser based WMS clients (*figure 5*). MapClient predefines the animation format in a particular style, define in what date ranges and in what time slices animation is needed. The appropriate query is created (e.g. *figure 3*) and sent to WMS thorough its getMap web service interface. Once WMS get this query, it creates successive queries (req_1, req_2, .., req_n) based on the time parameter in getMap request. Each of those queries is responded 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 a part of an image-array and displayed in a vertical sub-window simulating a camera film.

3.3. Publishing Animated Map Images

WMS (see 3.2) are access points for the distributed systems or any other clients to use map animation services. The first step in an animation is the creation of series of temporally related successive map image, which is explained in the previous chapter. The second step is playing these still map images as an animation, and explained in this chapter.

WMS need 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 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, there are different approaches according to application requirements. In the Internet world, widely used and well-known approach is using animated GIF89a files. The browser will be enough to open the animation, but the users do not have any control on the image such as stopping, pausing or play-backing the animation. The animated gif will play only in one direction. There is no way to make it play backward. PROC GMAP [25] is an example of gif animations. The Java Media Framework (JMF)

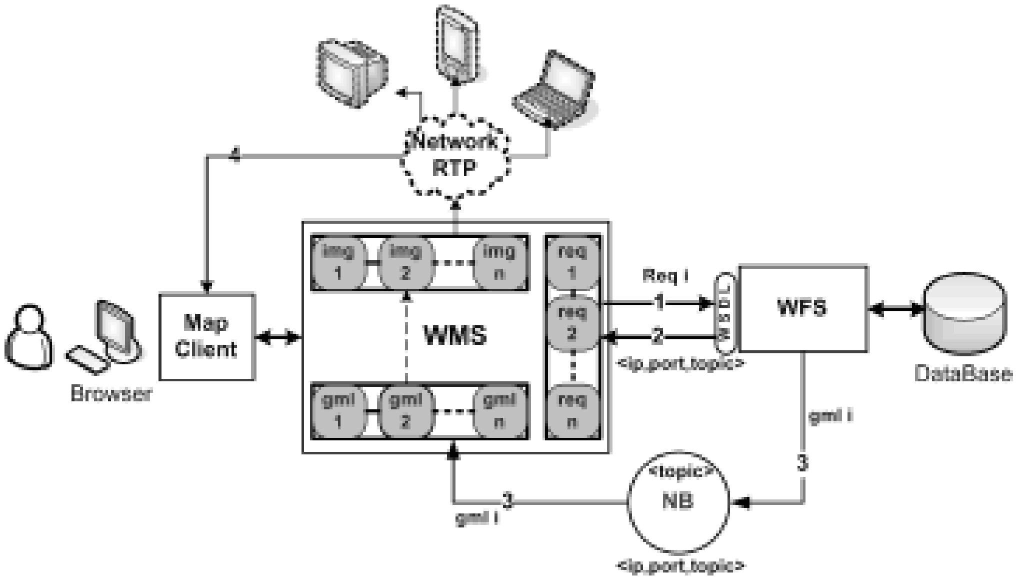


Figure 4. Detailed Streaming Map Movies architecture

[26] and Quicktime [27] provide with other approaches to assemble individual frames into an animation. JMF uses RTP sessions and need 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. It only specifies the interfaces in terms of standard queries and output formats. In the proposed framework, we preferred to use IP multicast approach with JMF technologies.

An animation is produced as video streams. Map images are converted into sequence of video streams and published to an Real Time Protocol (RTP) session [28]. 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/), but they are supposed to support H.263 and H.261 formats. The produced streams are published to multicast or unicast RTP sessions. Video streams can be delivered to a variety of platforms such as RealPlayer, Polycom and Access Grid [29]. 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 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 OneEarth project, http://onearth.jpl.nasa.gov. The second layer is Turkey's earthquake seismic data recorded from 1992 to 2005. Those feature data sets 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), date/time and some other minor attributes. Queries to create maps from those data sets are done based on those attributes. There are also some other parameters need to be set during the publication of contents of the streams. Frames (map images) are updated at every 0.3 seconds, and the frame rate is 5 frames per second. From our experience,

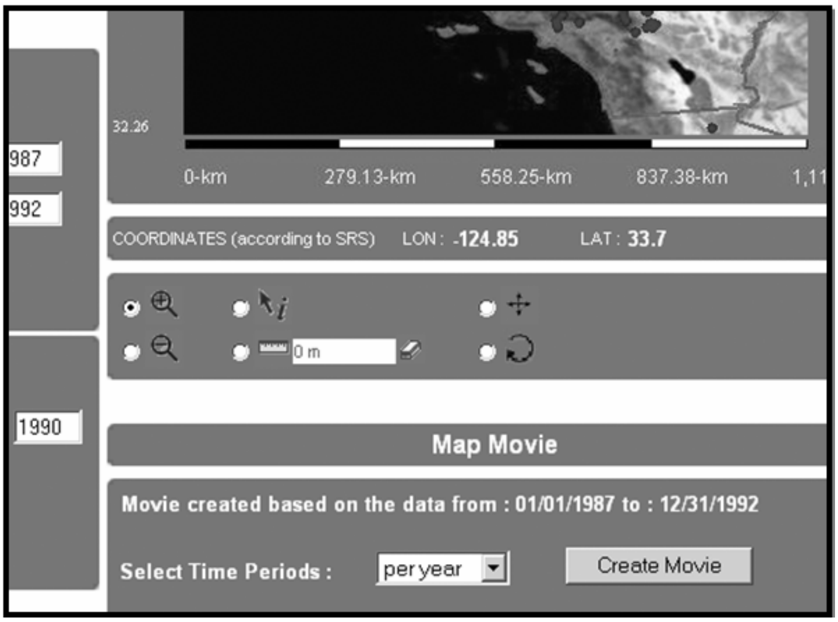


Figure 5. User interface (WMS Client) for streaming map movies

this parameter set for configuration is sufficient but they can be changed according to the application requirements. Depending on the underlying network and characteristics of data and servers used, the parameter values change to get best results. This also explains why update and frame rates are different.

Let the table keeping the earthquake seismic data be called Earthquake-Seismic, and let it have the following attributes: X: Longitude value of seismicity of earth, Y: latitude values of seismicity on earth, Magnitude: the power of the seismicity changing from 0 to 10, Year: in which seismic event occurred, Month: in which seismic event occurred, Day: in which seismic event occurred.

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

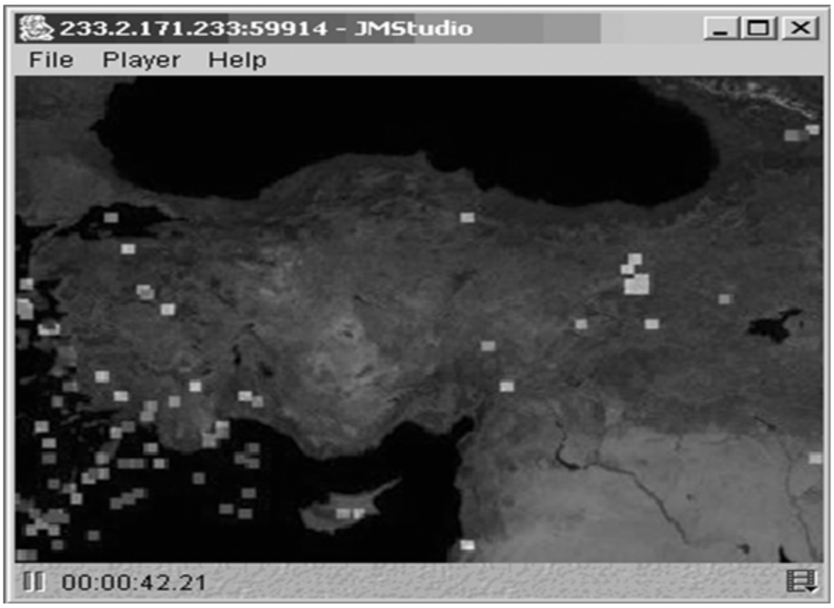


Figure 6. Displaying Map Movie with JMF Studio

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

However, the data is provided with 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 as 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 non-geometrical 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 move stream.

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

- How often does the earthquake occur?
- Which regions have the higher possibilities of facing an earthquake?
- Where are the safest places in terms of the earthquakes?
- Is there any pattern of the earthquake happenings?
- What are the periodicities of earthquake happenings whose magnitude greater than some values?

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

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 Java virtual machine (JVM) installed on it. As it is shown in the *figure 6*, light boxes show 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.

4. Conclusion and Future Work

The work presented here has looked into the possibilities of extending the web based map services with time-series data as animated maps, and introduced a framework enabling integration of animation services to the distributed systems. At the core of the framework there is a WMS. It is actually an access point for the 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 the framework, GIS web services uses standard interfaces for the handshake, the actual data is 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 a satellite map images and playing them successively with the predefined time intervals, we have determined the characteristics of the earthquake happenings for specific regions on a map. The proposed framework and the approach have been proved to be an effective tool 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 platforms and integrated to the third party distributed system applications.

In the proposed system, movie streams (for map animations) are created on demand from the archived data sets. 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 them to be searched. These parameters might be „temporal data layers from which movie streams are created", „frame rates", „starting-ending time/dates of the animation" and „periodicity of the data frames".

References

1. Peng, Z.-R. and M.-H. Tsou. Internet GIS: Distributed Geographic Information Services for the Internet and Wireless Networks. New Jersey, USA: John Wiley & Sons, 2003.
2. OGC. The Open Geospatial Consortium, Inc 1994 06/12/2008 [cited 02/14/2008]; The Open Geospatial Consortium, Inc]. Available from: http://www.opengeospatial.org/.
3. Beaujardiere, J.d.I. OGC Web Map Service Interface. Open GIS Consortium Inc. (OGC), 2006.
4. Vretanos, P. A. Web Feature Service 2.0 Interface Standard. Open Geospatial Consortium Inc, 2010.
5. Tanenbaum, A. S. and M.V. Steen. Distributed Systems: Principles and Paradigms, 2Ed., Prentice Hall 704, 2006.
6. Cox, S., et al. OpenGIS® Geography Markup Language (GML) Encoding Specification. Open Geospatial Consortium (OGC), 2003.
7. Pallickara, S. and G. Fox. NaradaBrokering: A Distributed Middleware Framework and Architecture for Enabling Durable Peer-to-Peer Grids. ACM/IFIP/USENIX, ACM: Rio Janeiro, Brazil, 2003.
8. Blok, C., et al. Visualization of Relationships between Spatial Patterns in Time by Cartographic Animation. Cartography and Geographic Information Science, 26 (2), 1999, 139-151.
9. Peterson, M.P. Interactive and Animated Cartography. Englewood Cliffs, NJ: Prentice Hall., 1995, 257.
10. Harrower, M. Tips for Designing Effective Animated Maps Cartographic Perspectives, 44, 2003, 63-65.
11. Harrower, M. Visualizing Change : Using Cartographic Animation to Explore Remotely-sensed Data. Cartographic Perspectives, 39, 2002, 30-42.
12. Frihida, A., D. J. Marceau, and M. Theriault. Extracting and Visualizing Individual Space-Time Paths: An Integration of GIS and KDD in Transport Demand Modeling. Cartography and Geographic Information Science, 31 (1), 2004, 30-42.
13. MathWorks. Mapping Toolbox 2012 [Cited 2012 March 03, 2012]. Available from: http://www.mathworks.com/products/mapping/.
14. GeoServer. GeoServer WMS Animator 2011 [Cited 2012 February 2, 2012]; Available from: http://docs.geoserver.org/stable/en/user/tutorials/animreflector.html#.
15. Kцbben, B. SVG and Geo Web Services for Visualization of Time Series Data of Food Risk. SVG Open 2008. Nьrnberg, Germany, 2008.
16. Becker, T. Visualizing Time Series Data Using Web Map Service Time Dimension and SVG Interactive Animation, in International Institute for Geo-Information Science and Earth Observation. University of Twente: Enschede, Netherlands,2009.
17. GoogleEarth. GoogleEarth. [Cited August 21, 2011], 2011.
18. Atkinson, M., et al. Web Service Grids: An Evolutionary Approach Concurrency & Computation: Practice&Experience, 17 (Number 2-4, February/April 2005), 2005, 377-389.
19. LaMar, E. Proposed Animation Service Extension. Open Geospatial Consortium Inc. (OGC), 2005, 23.
20. Pierce, M. E., et al. The QuakeSim Project: Web Services for Managing Geophysical Data and Applications. Pure and Applied Geophysics (PAGEOPH), 165 (3-4), 2008, 635-651.
21. Aydin, G., et al. Building and Applying Geographical Information Systems Grids. Concurrency and Computation: Practice and Experience, 20 (14), 2008, 1653-1695.
22. Aktas, M., et al. iSERVO: Implementing the International Solid Earth Research Virtual Observatory by Integrating Computational Grid and Geographical Information Web Services. Pure and Applied Geophysics (PAGEOPH), 163 (11-12), 2006, 2281-2296.
23. Sayar, A., M. Pierce, and G. Fox. Developing GIS Visualization Web Services for Geophysical Applications, in ISPRS Spatial Data Mining Workshop,Ankara, Turkey, 2005.
24. Tiampo, K. F., et al. Pattern Dynamics and Forecast Methods in Seismically Active Regions Pure and Applied Geophysics, 159 (10), 2002, 2429-2467.
25. Eberhart, M. Make the Map You Want with PROC GMAP and the Annotate Facility. Philadelphia Department of Public Health, 2008.
26. JMF. Java Media Framework 2011. [Cited 2008 09/01/2011]; Java Media Framework]. Available from: http://www.oracle.com/technetwork/java/javase/tech/index-jsp-140239.html.
27. Quicktime. 2011 [Cited; Available from: http://developer.apple.com/quicktime/.
28. Schulzrinne, H., S. L. Casner, and F. V. Jacopson. RTP: A Transport Protocol for Real-Time Applications. 2003.
29. Wu, W., et al. Design and Implementation of a Collaboration Web-Service System Neural, Parallel & Scientific Computations, 12 (3), 2004.

Manuscript received on 19.12.2011

Ahmet Sayar *has received his Ms degree in Computer Science from Syracuse University (Syracuse, NY,USA), and his PhD degree in Computer Science from Indiana University (Bloomington, IN, USA). During his PH.D. study he has worked at Los Alamos National Laboratory (New Mexico, USA) and Community Grids Laboratory (Indiana, USA) as a graduate research assistant. He is currently an Assistant Professor and Assistant Chair of Computer Engineering Department at Kocaeli University in Turkey. His current research interests are applications of Web technologies to problems in scientific computing, Web Services, Geographic Information Systems, and distributed spatial databases.*

Contacts:
Department of Computer Engineering, Kocaeli University
Izmit/Kocaeli, 41380 Turkey
tel. +90(532)357-3368
e-mail: ahmet.sayar@kocaeli.edu.tr