Computers & Geosciences 35 (2009) 1802–1816

Contents lists available at ScienceDirect

Computers & Geosciences

journal homepage: www.elsevier.com/locate/cageo

Building GML-native web-based geographic information systems

Chia-Hsin Huang [a,b,*], Tyng-Ruey Chuang [a], Dong-Po Deng [a], Hahn-Ming Lee [a,c]

[a] Institute of Information Science, Academia Sinica, Taipei 115, Taiwan
[b] Department of Electronic Engineering, National Taiwan University of Science and Technology, Taipei 106, Taiwan
[c] Department of Computer Science and Information Engineering, National Taiwan University of Science and Technology, Taipei 106, Taiwan

A R T I C L E I N F O

Article history:
Received 31 March 2008
Received in revised form
6 November 2008
Accepted 20 November 2008

Keywords:
GML
Web-based GISs
SVG
XQuery
DOM
SAX
RDBMS

A B S T R A C T

Disaster response systems are designed to facilitate decision-making based on large amounts of heterogeneous geographic information. Most geographic information systems (GISs) use relational databases to manipulate information efficiently. However, they suffer from interoperability issues because they need to expend significant effort mapping heterogeneous geographic information, which may have complicated structures, into relational data models, and vice versa. Geography Markup Language (GML) is regarded as a standard for expressing, storing, and exchanging geospatial data, and has been applied to help solve interoperability problems. Interestingly, no GIS has been built on native XML/GML technologies so far. There are two possible reasons for this: current XML processors are incapable of processing geospatial information, and they are inefficient in manipulating large XML documents. In this paper, we resolve these two difficulties and move forward to realizing GML-native web-based geographic information systems.

1. Introduction

People living in Taiwan are always under threat from natural disasters, such as typhoons, earthquakes, landslides, and floods. Typhoons, for example, often bring huge amounts of rain and cause severe floods and landslides. These natural hazards demonstrate the need for efficient disaster response systems. Such systems are data-intensive and time-sensitive with intolerance of delays in decision-making and response operations. In addition, the systems need to exchange and process heterogeneous data among multiple government departments and institutes. Efficiency and interoperability are thus crucial concerns. Geospatial information systems (GISs) have played key roles in emergency response operations (Mansouriana et al., 2006). Current GISs address the performance issue, but they pay relatively less attention to the interoperability issue. Hence, it emerges the need to utilize appropriate technologies to resolve these issues and develop robust GISs (Lu et al., 2007).

To manipulate large volumes of geographic information efficiently, many GISs[1] use relational database management systems (RDBMSs) with geospatial extensions (Corcoles and Gonzalez, 2002). Taking advantage of mature indexing techniques, DB-driven GISs are able to retrieve geographic information efficiently. Many GISs have supported eXtensible Markup Language (XML) (Bray et al., 2006) and Geography Markup Language (GML) (Cox et al., 2003) for importing and exporting information. Nevertheless, the interoperability issue still arises because GIS and DBMS software providers normally use their own relational schemes and file formats. One system may not understand the data generated by another, so it cannot reuse the data directly. This also results in some delays in transforming data formats. Another difficulty DB-driven GISs encounter is to express and manipulate structural information in relational data models (Zhu et al., 2007). Real world phenomena are naturally complicated, and relational data models often fail to convey the real structures of geographic features. Moreover, in order to support domain knowledge or improve the discovery of information, applications may want to organize geographic information to comply with some geographic categorization systems or ontologies, which may also possess complicated structures (Raskin and Pan, 2005). For example, there are over 50 relational tables in the relational data model of Alexandria Digital Library Gazetteer,[2] a distributed digital library with collections of geo-referenced materials. It would require a significant effort to import/export and maintain

* Corresponding author.
E-mail addresses: D9002101@mail.ntust.edu.tw, jashing@iis.sinica.edu.tw (C.-H. Huang), trc@iis.sinica.edu.tw (T.-R. Chuang), dongpo@iis.sinica.edu.tw (D.-P. Deng), hmlee@mail.ntust.edu.tw (H.-M. Lee).

[1] Such as ESRI's ArcGIS, Autodesk's MapGuide, and Oracle's MapViewer, as well as open source RDBMS with geospatial extensions, such as MySQL/GIS-extensions and PostgreSQL/PostGIS.

[2] http://www.alexandria.ucsb.edu/gazetteer/ContentStandard/dbmodel/main.jpeg

0098-3004/$ - see front matter © 2009 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cageo.2008.11.009

information in such a complicated data model. The above disadvantages create barriers to developing GISs and disaster response systems.

GML, on the other hand, is more flexible and natural in expressing geographic information; hence it has rapidly emerged as an open standard for modeling, transporting, and storing such information. GML encodes geospatial as well as non-geospatial information in XML. Thus, GISs using GML technologies are able to alleviate the interoperability problem (Shekhar et al., 2001). It is also natural and easier to manipulate geographic information in GML documents directly, rather than importing the information into DBMSs. Furthermore, as the Web is a popular medium for information exchange now, the need to exchange geographic information seamlessly over the Web also arises. Open Geospatial Consortium's (OGC) Web Processing Service (WPS) (Schut, 2007) defines a standardized interface that facilitates the publishing of geospatial processes, and the discovery of and binding to those processes by clients. It is expected that, by using XML/GML technologies rather than DB technologies as the core of GISs, the GISs will be better able to support such standards.

In fact, XML/GML technologies have been used to develop GML-native processors, like GQuery (Boucelma and Colonna, 2004), which has been applied to a geographic mediation system (Essid et al., 2004). Unfortunately, generic XML processors, including GQuery, are unable to process large documents efficiently. This is an inherent weakness of generic XML processors. Interestingly, the GML specification (Cox et al., 2003) does not address the performance issue. This might imply that GML is not designed for use in performance-sensitive or online applications. Our goal in this paper is to enhance the performance of the XML-based processors in order to realize practical GML-based Web GISs.

We briefly review two XML processing models: the Document Object Model (DOM) (Hors et al., 2004) and a streaming model, namely Simple API for XML (SAX[3]). Most standard XML processors, such as XPath (Clark and DeRos, 1999) and XQuery (Boag et al., 2007), use DOM (a tree structure) to represent XML documents in the memory. DOM typically needs an internal memory space that is five times the size of the external documents. It is also a time consuming task to build a DOM tree for a large document. In contrast, the streaming model views an XML document as a data stream. A streaming parser scans a document and triggers corresponding events while XML elements are visited. The main advantage of using a streaming processing model is that it consumes a small amount of memory space. However, unlike DOM, a streaming model cannot access documents randomly and cannot be used for evaluating user queries directly. To sum up, standard XML processing models are insufficient for manipulating large documents, which are common in GISs. Therefore, to realize practical GML-based Web GISs, this performance issue must be addressed.

In this paper, we develop GML-based Web GISs, in which geographic information is stored and manipulated in GML. We now briefly summarize this work and our previous effort toward our goal. We have introduced an XML prefilter (Huang et al., 2005), and applied it to an XQuery processor to support geospatial information processing (Huang et al., 2006a). In this work, the XML prefilter has been extended in the following areas. We model the efficiency of the XML prefilter. The model guides us to optimize the prefilter. Another improvement is that we develop an efficient index engine so that the prefilter can manage indexes better.

[3] http://www.saxproject.org/

We summarize our contributions to the GIS field as follows.

- Our work shows that it is possible to use XML/GML as the core technologies in developing GML-based Web GISs. To the best of our knowledge, this is the first work that addresses and realizes GML-based Web GISs.
- We develop two GML-native processors by adapting, respectively, a streaming-based XML parser and an XQuery processor to support the processing of geospatial information.
- We present an XML/GML prefilter, a tiny indexing and search engine, used to increase the performance of generic XML/GML processors, and apply it to improve the performance of the GML-native processors.
- We have successfully developed a GML-based Web GIS prototype by replacing the database component of a DB-driven Web GIS prototype with the enhanced GML processors.
- We conduct intensive experiments on the performance of the prefilter and the GML-native processors. The results show that the prefilter improves the performance of GML-native processors by two to tenfold. We also observe that the GML-based Web GIS performs as well as the DB-driven approach on most queries.
- We propose two optimization mechanisms, namely simplifying queries and caching intermediate results, for the prefilter. By enabling the mechanisms, the enhanced GML-based Web GIS is able to outperform the DB-driven Web GIS.
- We provide the source code of software packages mentioned in the paper for public access.[4] Users just need to download the GML-native processors and prepare GML documents to test and realize a simple GML-based Web GIS. They can also use the APIs of the prefilter to enhance their XML-based processors.

2. GML data models and queries

GML is based on XML and is used to encode geospatial information. GML version 3 provides several objects for describing geography, including features, coordinate reference systems, geometry, topology, time, units of measure, and generalized values (Cox et al., 2003). Applications can extend or restrict these GML objects to fit their requirements. In this paper, we focus on a data model of geographic features.

As defined in the GML specification, a geographic feature is "*an abstraction of a real world phenomenon*; *it is a geographic feature if it is associated with a location relative to the Earth*." A feature may have a set of properties that present its states. Each property is represented as a triple of name, type, value. Naturally, GML uses a hierarchical structure to express geographic features. Specifically, a set of features is aggregated as a feature collection, which can also be viewed as a feature. A geographic data model is a collection of geographic features, feature types, and general integrity constraints.

2.1. GML instance

Fig. 1 shows a fragment of a GML instance comprising several types of geographic features, such as rivers and roads, of a small island, called *Lanyu*. Each geographic feature is enclosed by a *FootPrint* element with an *id* attribute. A FootPrint is followed by a *boundedBy* element that contains two points (minX, minY) and (maxX, maxY) in a feature's geometry. The boundedBy elements are used to aid to accelerate query processes. We explain this

[4] http://www.iis.sinica.edu.tw/~jashing/prefiltering/