Scalable Data Collection Infrastructure for Digital Government Applications *

Hanan Samet Egemen Tanin

Department of Computer Science
Center for Automation Research
Institute for Advanced Computer Studies
University of Maryland at College Park

hjs@cs.umd.edu egemen@cs.umd.edu

Leana Golubchik

CS and EE-S Departments
Integrated Media Systems Center
Information Sciences Institute
University of Southern California

leana@cs.usc.edu

ABSTRACT

Highlights are presented of a project on the development of a scalable data collection infrastructure of digital government applications. The approach takes advantage of both the distributed nature of the data and the distributed ways of interacting with it. The approach is three-parted having theoretical, systems, and application components. On the theoretical and systems levels, work has continued on the development of the BISTRO system. On the applications level, work has continued on the investigation of the use of peer-to-peer networks in a number of domains in which the proper handling of spatial data is important.

Categories and Subject Descriptors

C.2 [**Computer-communications Networks**]: Distributed Systems—*Distributed databases*; D.2.8 [**Software Engineering**]: Metrics—*complexity measures, performance measures*; H.2.8 [**Database Management**]: Database Applications—*Spatial databases and GIS*

General Terms

Algorithms, Measurement, Performance, Security

1. INTRODUCTION

In this paper we describe highlights of the project titled "Scalable data collection infrastructure for digital government applications" under the auspices of the Digital Government Research Program of the National Science Foundation. Our research is focussed on taking advantage of the distributed nature of data and the interaction with it. Our efforts have been directed at both the systems/theoretical and applications levels. On the systems and theoretical levels, we have continued our development of the BISTRO system (Section 2). On the applications level, we have been investigating the use of peer-to-peer networks in a number of domains in which the proper handling of spatial data plays an important role (Section 3).

*This work was supported in part by the US National Science Foundation under Grant EIA-00-91474.

2. BISTRO

Hotspots are a major obstacle to achieving scalability in the Internet; they are usually caused by either high demand for some data or high demand for a certain service. At the application layer, hotspot problems have traditionally been dealt with using some combination of increasing capacity, spreading the load over time and/or space, and changing the workload. Previous classes of solutions have been studied in the context of applications using one-to-many, many-to-many, and one-to-one communication. However, to the best of our knowledge there is no existing work, except ours on making applications using many-to-one communication scalable and efficient; existing solutions simply use many independent one-to-one transfers. This corresponds to an important class of applications, whose examples include digital government tasks such as submission of income tax forms to IRS. We proposed Bistro, a framework for building scalable and secure wide-area digital government upload applications.

Briefly, the Bistro upload architecture works as follows. Given a large number of clients that need to upload their data by a given deadline to a given destination server, the Bistro architecture breaks the upload problem into three steps. Step 1, which is the timestamp step, must be accomplished prior to the deadline for clients to submit their data to the destination server. In this step, each client sends to the server a message digest of their data and in return receives a secure timestamp ticket from the destination server as a receipt indicating that the client made the deadline for data submission. The purpose of this step is to ensure that the client makes the deadline without having to transfer their data which is significantly larger than a message digest and might take a long time to transfer during high loads which are bound to occur around the deadline time. It is also intended to ensure that the client (or an intermediate bistro used in Step 2) does not change their data after receiving the timestamp ticket. All other steps can occur before or

after the deadline. Step 2 is the transfer of data from clients to intermediate hosts, termed bistros. This results in a low data transfer response time for clients. Since the bistros are not trusted entities (unlike the destination server), the data is encrypted by the client prior to the transfer. Step 3 is the collection of data by the destination server from the bistros. The destination server determines when and how the data is collected in order to avoid hotspots around the destination server. Once the destination server collects all the data, it can decrypt it, recompute message digests, and verify that no changes were made to a client's data (either by the client or by one of the intermediate bistros) after the timestamp ticket was issued. A summary of main advantages of this architecture is: (1) hotspots can be eliminated around the server because the transfer of data is decoupled from making of the deadline, (2) clients can receive good performance since they can be dispersed among many bistros and each one can be direct to the best bistro for that client, and (3) the destination server can minimize the amount of time it takes to collect all the data since now it is in control of when and how to do it (i.e., Bistro employs a server pull).

Our main research activities within the Bistro framework have been along the above described three steps. Moreover, this year, in addition to focusing on performance and security issues, our efforts have also included research directions on fault tolerance issues related to the entire Bistro framework. That is, the security mechanisms in the Bistro upload protocols guarantee integrity and privacy of the data being upload. However, to improve the performance characteristics of our scheme, it is still desirable to provide mechanisms and policies for ensuring that data will not have to be retransmitted due to losses or temporary unavailable which could occur due to failures or malicious behavior of various system components.

To this end, our work focuses on augmenting our current Bistro architecture with appropriate fault tolerance and redundancy mechanisms and policies, where the amount of redundancy and degree of fault tolerance depends on the application and the reliability characteristics of the system components. Several interesting questions arise in this context, including (a) which fault tolerance schemes should be used in Bistro, (b) how much redundancy in communication bandwidth and temporary storage is needed, and (c) what is the effect on the performance of the upload system when fault tolerance protocols are introduced. Our goal in this work is to maintain comparable performance to that of a system without fault tolerance mechanisms and to reduce the overhead attributed to fault tolerance mechanisms (such as storage and network bandwidth overheads) as much as possible. Specifically, during the server pull process of the data collection step, bistros can be unavailable due to failures, or they can be malicious, i.e., they might intentionally corrupt data. This degrades system performance since the destination server needs to ask for retransmissions. As a result, we need to provide an appropriate fault tolerance protocol. In this work, we develop such a protocol which employs erasure codes in order to make the data uploading process more reliable. We develop analytical models to study reliability and performance characteristics of this protocol, and we derive a cost function to study the tradeoff between reliability and performance in this context.

3. PEER-TO-PEER NETWORKS

Peer-to-peer (P2P) networks are becoming increasingly popular as a powerful means for data exchange. They are quite scalable and easy to deploy. Although P2P networks attract significant interest, methods for accessing complex data such as spatial data on P2P networks are still at their infancy. Several future P2P applications like P2P job-employee seeker networks and P2P virtual cities will need to answer queries involving spatial data, i.e., data with locational components. Frequently more than one location is associated with a spatial object. Hence, it commonly differs from conventional point data in that the objects may also have extent. For example, lakes can be represented as spatial objects whose extent is defined by the space that they occupy. There are many ways to describe such objects including their boundary, the locations that make up their interior, their minimum axis-aligned bounding box, etc. Thus a spatial description is more than just a longitude and a latitude value. Retrieving such complex data, given a query, on a dynamic network is a non-obvious task. There is no central server or administration that the data is stored. Hence, classical indexing methods and querying algorithms cannot be easily applied. Yet, efficient solutions to querying and locating spatial data on P2P networks can enable many government as well as other public domain networked applications. For example, from a digital government point of view, government agencies can form ad hoc virtual environments where they present and exchange their data without dedicated servers or can help the general public to exchange this data among themselves using existing, mostly unused, computational resources available at users' machines.

This year we focussed our efforts on the use of online virtual-worlds that has great potential for government, industry, and in general public domain applications. Recent developments of 3D virtual-world gaming has attracted many participants, various terrain-modelling based collaborative applications on a highly distributed scale can be considered. Yet, online multi-participant virtual-worlds do not easily scale because the communications and computations are centered around some dedicated servers. We have developed a distributed quadtree index for representing spatial object that has been found to perform well. This index is a variant of the MX-CIF quadtree (also known as a loose quadtree) which has found use in many applications ranging from VLSI design, spatial databases, and games. It is based on representing objects by their minimum enclosing quadtree blocks. It is a variant of order-preserving hashing. It is an image-based representation in contrast to object-based representation such as an R-tree. This means that it can be used with operations that involve more than one dataset which are in registration. In addition, we have addressed the issue of finding nearest neighbors in such an environment. This is a basic query that is the central component of any application that involves searching in a domain that involves data with a locational component. Our work makes use of a best-first algorithm developed by our group that has been proved to be I/O optimal. The advantage of this algorithm is that it can be run in an incremental manner so that the number of neighbors that are desired need not be known in advance, and thus the algorithm need not be restarted from scratch if additional neighbors are needed.