AN OVERVIEW OF THE SEQUOIA 2000 PROJECT

Michael Stonebraker

Sequoia 2000 Technical Report 91/5
Computer Science Division, EECS Department
University of California
Berkeley, Ca. 94720

Abstract

Achieving the goals of the U.S. Global Change Research Program will depend not only on improved measurement systems, but also on improved data systems that will allow scientists to manipulate the resulting large-scale data sets and climate system models, as well as compare model results with observations. New modes of research, especially the synergistic interactions between observations and model-based simulations, will require massive amounts of diverse data to be stored, organized, accessed, distributed, visualized, and analyzed. Computer scientists and environmental researchers at several UC campuses are collaborating to address these challenges. Refinements in computing — specifically involving storage, networking, file systems, extensible data base management, and visualization — will be applied to specific Global Change applications.

We have named this project **Sequoia 2000,** after the giant trees of the Sierra Nevada, the largest organisms on the Earth's land surface.

1. INTRODUCTION

1.1. Background

One of the most important challenges that will confront the scientific and computing communities during the 1990s is the development of models to predict the impact of Global Change on the planet Earth [CPM91]. Among the specific issues that must be faced are:

- the "greenhouse effect" associated with increasing concentrations of carbon dioxide, methane, chlorofluorocarbons, and other gases;
- ozone depletion in the stratosphere, most notably near the South Pole, resulting in a significant increase in the ultraviolet radiation reaching the Earth's surface;
- a diminishing supply of water of suitable quality for human uses;
- deforestation and other anthropogenic changes to the Earth's surface, which can affect the carbon budget, patterns of evaporation and precipitation, and other components of the Earth System;
- a pervasive toxification of the biosphere caused by long-term changes in precipitation chemistry and atmospheric chemistry;
- biospheric feedbacks caused by the above stresses and involving changes in photosynthesis, respiration, transpiration, and trace gas exchange, both on the land and in the ocean.

Environmental Scientists, typically in Earth Sciences departments, are addressing these concerns and report considerable shortcomings in current information systems that impair their ability to make research progress on these problems [CEES91]. Specifically, they report the five shortcomings indicated in the next subsection.

1.2. Shortcomings of Current Information Systems

1) Current storage management system technology is inadequate to store and access the massive amounts of data required.

The data requirements in these problem areas are massive. They entail storing a variety of satellite imagery, e.g. Thematic Mapper (TM), Landsat Multispectral Scanner (MSS) and Advanced Visible and Infrared Imaging Spectrometer (AVIRIS) data, vector maps such as USGS topographic maps, as well as experimental data sets. In aggregate, the five Earth Sciences groups that are participating in Sequoia 2000 have about 100 Terabytes of information that they would like to store.

While it is possible, in theory, to build a magnetic disk system with this capacity, its cost would be prohibitive. A much more cost-effective solution would incorporate a multi-level storage hierarchy that uses not only magnetic disk but also one or more tertiary memory devices.

Current system software, including file systems and data base systems, offers no support for this type of multi-level storage hierarchy. Moreover, current tertiary memory devices (such as tape and optical disk) are exceedingly slow, and hardware and software must mask these long access delays through sophisticated caching, and increase effective transfer bandwidth by compression techniques and parallel device utilization. None of the necessary support is incorporated in currently available commercial systems.

2) Current I/O and networking technologies do not support the data transfer rates required for browsing and visualization.

Examination of satellite data or output from models of the Earth's processes requires that we **visualize** data sets or model outputs in various ways. A particularly challenging technique is to **fast-forward** satellite data in either the temporal or spatial dimension. The desired effect is similar to that achieved by the TV weather forecasters who show, in a 20-second animated summary, a sequence of images collected from a weather satellite over a 24-hour period.

To do such visualization in real time places severe demands on the I/O system to generate the required data at a usable rate. Additionally, severe networking problems arise when investigators are geographically remote from the I/O server. Not only is a high bandwidth link required that can deliver 20-30 images per second (i.e. up to 600 Mbits/sec), but also the network must guarantee delivery of required data without pauses that would degrade real-time viewing. Current commercial networking technology cannot support such "guaranteed delivery" contracts.

3) Current visualization software is too primitive to allow Global Change researchers to render the data returned for useful interactive viewing on a user workstation.

Global Change researchers would like, for example, to roam through an AVIRIS "image cube," displaying any three of the 224 spectral bands in RGB color, while at the same time displaying information, perhaps graphically, about all 224 bands. Today, each scientist must develop a substantial amount of device-specific and dataset-specific display and rendering code to perform such functions. Even after development, such code faces substantial performance problems if there is not enough space to buffer the information from an entire dataset locally.

4) Current data base systems are inadequate to store the diverse types of data required.

Earth Scientists require access to the following disparate kinds of data for their remote sensing applications:

- Point Data for specific geographic points.
- Vector Data. Topographic maps are often organized as polygons of constant elevation (i.e. a single datum applying to a region enclosed by a polygon).
- Raster Data. Many satellite and aircraft remote sensing instruments produce a regular array of point measurements. The array may be 3-dimensional if multiple measurements are made at each location.
- Text Data. Global Change researchers have large quantities of textual data including computer programs, descriptions of data sets, descriptions of results of simulations, technical reports, etc. that need to be organized for easy retrieval.

Current commercial relational data base systems (e.g. DB 2, RDB, ORACLE, INGRES, etc.) are not good at managing these kinds of data. During the last several years a variety of next generation DBMSs have been built, including IRIS [WILK90], ORION [KIM90], POSTGRES [STON90], and Starburst [HAAS90]. The more general of these systems appear to be usable, at least to some extent, for point, vector, and text data. However, none are adequate for the full range of needed capabilities.

5) It is extremely difficult to share the objects noted above with other interested researchers.

Most of the data objects that Earth Scientists wish to store are ones that they also wish to share with other researchers. Effective sharing of these classes of data objects requires an on-line, distributed **repository** that could **catalog** available objects, and then provide browsing support to a scientist. We call this **the electronic repository,**and it consists of software capabilities for indexing and browsing an object base built into and on top of a DBMS. Such software is not currently available.

1.3. Sequoia 2000 Participants

This paper describes the SEQUOIA 2000 research program at the University of California, which is executing a coordinated attack on these issues. This project is supported by Digital Equipment Corp as its flagship research project of the 1990's, replacing Project Athena at MIT, which was the corresponding project in the 1980's.

The participants in this project include Computer and Information Scientists, Global Change researchers, policy making groups in governmental bodies and industrial participants. The Computer and Information Scientists are located in the Computer Sciences Division and the School of Library and Information Studies at Berkeley, the Computer Sciences Department at San Diego and the San Diego Supercomputer Center. Their mission is to carry out a collection of research studies and build prototypes that will be described.

The Earth Scientists are from the Center for Remote Sensing and Environmental Optics (CRSEO) at Santa Barbara, the Department of Atmospheric Sciences at UCLA and the Scripps Institution of Oceanography in San Diego. They will use the prototype systems to be discussed and give feedback and research guidance to the Computer Scientists.

In addition, the State of California Department of Water Resources (DWR) and the U.S. Geologic Survey are project participants. Not only can they use the results of Global Change research, but also they have the above-mentioned five problems with current information systems and will be users of the prototype systems being developed.

Lastly, in addition to Digital Equipment Corporation, there are other industrial sponsors, including TRW and Hewlett Packard. They will also offer feedback and guidance on the SEQUOIA 2000 research plan.

In the remainder of this paper we give an overview of the Sequoia 2000 research plan. There are three companion papers which talk respectively about tertiary memory management, network management, and browsing access to the repository. Hence, these topics will only be casually mentioned. Instead we concentrate on the other aspects of Sequoia 2000, including file systems and visualization in this paper.

2. THE SEQUOIA 2000 RESEARCH PLAN

Sequoia 2000 is organized around an interconnected collection of hardware, file system, DBMS, networking, visualization, and repository projects. We discuss each in turn, concentrating on issues not covered in the companion papers [KATZ92, FERR92, CHEN92].

2.1. Hardware Concepts

The Global Change research groups wish to store approximately 100 Tbytes of data at reasonable cost and in a reasonable footprint. Obviously, a critical aspect of a storage management system subsystem of this size will be its support for managing a complex hierarchy of diverse storage media [RANA90]. Our goal is to build such a storage system, denoted BIGFOOT, capable of this level of capacity and with sufficient speed to support the traffic which will be directed to it. To gain experience and to support our client community in the immediate future, we are putting in place an "off the shelf" prototype with a capacity around 10 Terabytes. This system, called BIGFOOT I, should be operational in early 1992.

As noted in the companion paper in this volume [KATZ92], we are studying two different techniques to achieve high performance. Our first technique is based on the ideas developed in RAID technology [PATT88], a new way to construct high bandwidth, high availability disk systems based on arrays of small form factor disks [KATZ89]. High bandwidth comes from striping data across many disk actuators and harnessing this inherent parallelism to dramatically improve transfer rates. We are investigating applying these techniques to systems which include tertiary memory, and are considering striping in a wide variety of ways as noted in [KATZ92].

Besides striping, a second method for improving the transfer rate (and incidentally the capacity) of the storage system is compression [LELE87, MARK91]. Depending on the type of data present, size reductions from a factor of 2 - 30 have been reported. However, the compression technique which must be applied is specific to the type of data involved, and there is likely to be high leverage from hardware support. Lastly, when the user of a storage object (the client) is at a different location from the server storing the object, it is evident that the principle of "just in time" decompression should apply. Hence, decompression should be delayed as long as possible, and should be done only when a user program cannot operate on compressed data. Clearly, it does not make sense for the I/O path in the server to perform decompression, so that the networking system must immediately recompress the data for transmission over a long-haul network.

An important aspect of our research is an investigation of how hardware support for compression and decompression should be embedded into client and server data paths.

2.2. File Systems

We are exploring three different approaches to file systems which will run on the BIGFOOT prototypes. First, we will import a commercial off-the shelf systems (COTS) to run in our environment. Not only will this provide fallback software, in case both of our other approaches fail, but also, it will provide a benchmark against which to measure the systems we now discuss.

The second system that we are building, denoted Jaquith, is a migrating enhancement which can be added to any file system. It assumes the existence of a disk-based file system for the machine being used. To this it adds a tertiary memory file system, oriented toward tape jukeboxes. Hence, each file is stored contiguously on a tape cartridge, and only the last file on a tape can be extended. Others must be extended by a complete copy operation. Lastly, migration software is included which decides which tape files should be cached in the disk-based file system. This software is all non-kernal resident, and can be easily ported to any environment and used with any file system. Our software has a similar architecture to many commercial packages including Unitree and Epoch.

Our third file system is especially innovative, and does not use the the traditional organization of storage management, indicated in Figure 1. Here, the bottom level is one or more storage managers. These subsystems, usually implemented in device drivers, make a secondary storage device appear like a collection of fixed length blocks, addressed by block number. On top of these storage manager, an operating system has one or more file systems, which map the abstraction of a **user file** onto this storage system. User files are byte-addressible variable length, byte strings, that can be hierarchically organized into directories. User programs then can address files using the file system. It is clear that one or more storage managers are desirable. Moreover, the argument is made in [MULL91] that several file systems are desirable, and the file system switch being implemented in OSF/1 supports this concept.

Many traditional data base systems, e.g. INGRES and ORACLE, run using the model of Figure 1 by mapping their data structures into files supported by the file system. For example, INGRES maps each relation into an operating system file. However, a second model is used by some DBMS vendors, e.g. Sybase as noted in Figure 2. Here, the same storage manager is present as in Figure 1. However, the DBMS vendor abandons the OS-provided file system in favor of a "home-brew" one. This file system need only implement the notion of "flat" files, because a relational DBMS has no need for hierarchical addressing of storage objects. Then, it maps DBMS objects into the flat file system it provides, typically by alloting one relation to each file.

The difference between Figure 1 and Figure 2 is that unneeded pieces of the file system are discarded. Moreover, the implementation of the flat file system in Figure 2 is optimized for the

|l|l|.

_	_
top level	user programs
_	_
next level	one or more file systems
_	_
bottom level	one or more storage managers
_	_

Traditional Operating System Storage Manager
Figure 1

|l|l|.

_	_
top level	user programs
_	_
next level	DBMS-oriented flat file systems
_	_
bottom level	one or more storage managers
_	_

Second DBMS-oriented Model
Figure 2

large objects typically found in DBMSs rather than a mix of object sizes, such as found in traditional file systems. Lastly, the DBMS file system can support "ordered force out" and "synchronous write" so that DBMS supported transaction management can function correctly and efficiently.

POSTGRES has been written so it can function under either the model in Figure 1 or the model in Figure 2. Moreover, it has several notions for supporting large objects [STON92]. Hence, support for files can be provided as suggested in Figure 3. Here, a hierarchical file system is supported **on top of** the DBMS. The basic notion is to support two DBMS classes:

DIRCTORY (file-name, file-id, parent-file)
STORAGE (file-id, large-object)

The first DBMS class supports the directory structure, while the second simply supports storage of the bytes. Obviously, any implementation of large objects can be used to support the storage of files. Moreover, it is easy to specify the DBMS operations that must be performed to support

|l|l|.

_	_
top level	file support
_	_
next level	DBMS code
_	_
next level	DBMS-oriented flat file systems
_	_
bottom level	one or more storage managers
_	_

POSTGRES Support for Large Files
Figure 3

any particular file system calls.

This implementation has several advantages relative to the traditional implementation of files noted in Figure 1. First, files are data base objects, and therefore, transaction support is automatically provided. Hence, there is no need for the operating system file system to implement another transaction system. Second, it is likely that the DBMS implementation of large objects is optimized for big objects. Hence, higher performance on very large files is a distinct possibility. Our third file system, denoted Inversion, makes use of this architecture.

We are in the process of bringing up all three file systems in our environment, and a performance "bake-off" is planned. In the future, we are planning on other innovative tertiary memory file systems, including one based on ideas from the log structured file system (LFS) [ROSE91]. Moreover, we are experimenting with four and five level hierarchies, and migration algorithms that could take advantage of more than three levels.

2.2.1. Data Management Issues

In some environments it is desirable to use a DBMS rather than the file system to manage collections of large objects. Hence, we propose to extend the next-generation DBMS POSTGRES [STON90] to effectively manage Global Change data. There are three avenues of extension that we propose to explore.

First, POSTGRES has been designed to effectively manage point, vector and text data. However, satellite data are series of large multidimensional arrays. Efficient support for such objects must be designed into the system. Not only must the current query language be extended to support the time series array data that is present but also, the queries run by visualizers in **fast-forward** mode must be efficiently evaluated. This will entail substantial research on storage allocation of large arrays and perhaps on controlled use of redundancy. We are investigating decomposing a large multidimensional array into **chunklets** that would be stored together. Then, a fast-forward query would require a collection of chunklets to be accessed and then intersected with the viewing region. The optimal size and shape of these chunklets must be studied, as well as the number of redundant decompositions that should be maintained.

Second, POSTGRES has been designed to support data on a combination of secondary and tertiary memory. However, a policy to put historical data onto the archive and current data in secondary storage has been hard-coded into the current implementation. The rationale was that current data would be accessed much more frequently than historical data. While this may be true in many business environments, it will not be the case in Global Change research. Therefore, a more flexible way of dealing with the storage hierarchy must be defined that will allow "worthy" data to migrate to faster storage. Such migration might simply depend on the algorithms of the underlying file system discussed above to manage storage. However, the DBMS understands the logical structure of the data and can make more intelligent partitioning decisions as noted in [STON91].

The third area where we propose investigations concerns indexing. The conventional DBMS paradigm is to provide **value** indexing. Hence, one can designate one or more fields in a record as **indexed,** and POSTGRES will build the appropriate kind of index on the data in the required fields. Value indexing may be reasonable in traditional applications, but will not work for the type of data needed to support Global Change research. First, researchers need to retrieve images by their content, e.g. to to find all images that contain Lake Tahoe. To perform this search requires indexes on the result of a classification function and not on the raw image. Second, indexing functions for images and text often return a collection of values for which efficient access is desired [LYNC88]. For example, a keyword extraction function might return a set of relevant keywords for a document, and the user desires indexing on all keywords. In this case one desires **instance** indexing on the set of values returned by a function. We propose to look for a more general paradigm that will be able to satisfy all indexing needs of Global Change researchers.

2.3. Networking Hardware and Software

The University of California has donated funds to purchase T3 bandwidth to connect the Sequoia research sites, and we are in the process of constructing a long-haul network. We have several research ideas which we propose to explore on this network, as discussed in the companion paper [FERR92].

First, routers and switches are typically constructed from specialized hardware components. On the other hand, we expect general purpose RISC machines will be fast enough to serve this function on future networks without resorting to special purpose iron. As a result, Sequoia is planning to use conventional RISC computers as routers and switches. Initial measurements have been encouraging as indicated in [PASQ92].

Second, our network must carry primarily data traffic, composed entirely of large objects. As a result, the Asynchronous Transfer Mode (ATM), which is emerging as the preferred standard for the Broadband Integrated Services Digital Network (B-ISDN), may not be appropriate for the Sequoia network because of its small cell size (53 bytes). When network traffic is dominated by large image transmissions and video streams, a network optimized for large packets may be preferred.

Third, Sequoia users who are **fast-forwarding** data expect to see predictable response time to queries. Although they can buffer data at their client workstations, and then render the data at a smooth rate onto their screen, it would be preferable for the network (as well as the rest of the system) to be able to make guarantees about information delivery. In [FERR90] algorithms are presented which make such delivery guarantees, and we are in the process of migrating these algorithms to the Sequoia network.

Although image sequences require high bandwidth and low delay guarantees, these guarantees are often statistical in nature. The network, or even the workstation's operating system, can take advantage of this by conveniently dropping packets when necessary to control congestion and smooth network traffic. This is particularly relevant when one is fast-forwarding through a sequence of images; supporting full resolution might not be possible, and users might be willing to accept a lower resolution picture in return for faster movement.

One approach to this problem is hierarchical coding [KARL89], whereby a unit of information such as an image is decomposed into a set of ordered sub-images. A selected subset of these may be re-composed to obtain various levels of resolution of the original image. This gives the receiver the flexibility of making the best use of received sub-images that must be output by some deadline, and gives the network the flexibility of dropping packets containing the least important sub-images when packets must be dropped. We are also investigating the deployment of such "drop on the floor" protocols.

2.3.1. Visualization

Sequoia clients wish to visualize their data streams in ad-hoc ways on their workstations. Our project is taking three approaches to visualization environments. Our first approach is to have the Earth Sciences participants converge on a single package to be the preferred Sequoia visualization tool. Then, all Sequoia team members could focus on enhancing a single package, and substantial leverage would result. An effort to select such a preferred package is underway.

Our second approach recognizes that there are a large number of popular tools in this area, including AVS, Explorer, PV-wave, IDL, and SPAM. As such, convergence on a single preferred package may be difficult, and the real problem is to provide **inter-operability** between many tools. In this way, a file written by tool i in tool-i format should be convertable to the format understood by tool j, so that it can be used for a subsequent step of the same visualization sequence. We have designed a standard language for representing image files, a so-called **inter-lingua,** and are currently designing conversion tools that convert from popular file formats to and from our interlingua. Currently, more than 20 popular formats are understood, and more are under development [BAIL91].

Our third visualization effort concerns the architecture of such tools. Currently, visualization tools perform two separate functions, namely they provide **data rendering** and **data manipulation.** Hence, it is possible in many systems to overlay two images data manipulation) and then place the result on the screen (rendering). However, we expect the images that are to be combined will be stored in the POSTGRES DBMS. As such, POSTGRES has capabilities to overlay two data base images. Therefore, data manipulation should not be done by a visualization tool but should be **factored** out of such tools and relegated to the DBMS. As a result, the visualization tool need only perform fairly simple image rendering. In the process of extending the POSTGRES DBMS to meet Sequoia needs, we are co-incidentally building the data manipulation portion of a visualization system. We expect to design and implement a complete visualization tool, which will simply pass all data manipulation issues to the POSTGRES DBMS, for it to support.

2.4. The Electronic Repository

The electronic repository required by Global Change researchers includes various data sets, simulation output, programs, and documents. For repository objects to be effectively shared, they must be **indexed,** so that others can retrieve them by content. Moreover, effective user interfaces must be built so that a researcher can **browse** the repository for desired information.

We must address the issue of how to index the non-textual data of Global Change researchers. For example, they wish to find all instances of "El Nino" ocean patterns from historical satellite data. This requires indexing a region of spatial data and a region of time according to an imprecise (fuzzy) classification. One approach will be to use existing thesauri of geographical regions and place names that include the cartographic coordinates of the places. Researchers may also need to create their own classifications that can be used to select and partition the data.

We must also index computer programs in the repository. We will take two approaches in this area. First, we will index the documentation that is associated with a program using traditional techniques. In addition, because we have the source code available, we can also index program variables and names of called functions. This will allow retrieval, for example, of all repository programs that include the variable ‘‘drought_level’’ or the ones that call the subsystem SPSS.

Finally, mature techniques exist for indexing and retrieving textual documents based on automatic and manual indexing using keywords, thesaurus terms, and classification schemes. Statistically-based probabilistic match techniques have been developed that present the "best-matching" documents to the user in response to an imprecise query [BELK87, LARS91]. These techniques must be extended to deal with indexing large collections of complete textual documents, rather than just collections of document surrogates (i.e. titles and abstracts).

Tools are also needed to allow users to **browse** this repository to find objects relevant to their work. Our approach is to use a graphical spatial paradigm. In this case, we would require users to furnish an **icon** which would represent their object. The repository can then be viewed as a collection of icons, for which we would attempt to build an organizing tool that would place them spatially in 2 or 3 dimensions. This tool would support movement through the space of icons, by simply panning geographically. A user who located an icon of interest could then **zoom** on the icon and receive increasing amounts of information about the object. For functions, the tool could capture relevant information from program documentation. For example, the first level might be the documentation banner at the top of the function and the second level might be the call graph of the function. At the finest level of granularity, the entire source code would be presented.

These techniques can be extended to apply to maps in two ways. First, a collection of maps of different resolutions could be organized hierarchically. Zooming into a map would then cause it to be replaced by a higher resolution map of the target area. This paradigm could be further extended to include a **time** dimension, through which a user would be able to "pan" forward or backward in time, thereby obtaining **time travel** with the same interface. Second, icons could be associated with geographic co-ordinates and represent information associated with a point or region. For example, a data set of manually measured snow depths could have an appropriate icon assigned, say a depth gauge. A join query could be run to construct a composite map with the icons overlaid on any particular map. Zooming into an icon would then cause the icon to be replaced by a window with the more detailed data. As our second approach to the repository, we expect to explore this "pan and zoom" paradigm, popularized for military ships by SDMS [HERO80]. Our last companion paper presents the initial design of our browsing system [CHEN92].

REFERENCES

[BAIL91] Bailey, M.J., ‘‘Scientific Visualization for a Large, Remotely-Distributed User Community,’’ *Graphicon Conference Proceedings,* pp. 11-26 (February 1991).

[BELK87] Belkin, N. and Croft, W., ‘‘Retrieval Techniques,’’ *Annual Review of Information Science and Technology,* vol. 22, pp. 109-145 (1987).

[CEES91] Committee on Earth and Environmental Sciences, *Our Changing Planet: The FY 1992 U.S. Global Change Research Program,* Office of Science and Technology Policy, Washington, D.C. (1991).

[CHEN92] Chen, J. et. al., "The Sequoia 2000 Object Browser," (elsewhere in this proceedings).

[CPM91] Committee on Physical, Mathematical and Engineering Sciences, *Grand Challenges: High Performance Computing and Communications,* Office of Science and Technology Policy, Washington, D.C. (1991).

[FERR90] Ferrari, D. and Verma, D., "A Scheme for Real-time Channel Establishment in Wide-area Networks," *IEEE Journal on Selected Areas in Communications,* April 1990.

[FERR92] Ferrari, D. et. al., "Network Issues for Sequoia 2000," (elsewhere in this proceedings).

[HAAS90] Haas, L. et al., ‘‘Starburst Mid-Flight: As the Dust Clears,’’ *IEEE Transactions on Knowledge and Data Engineering* (1990).

[HERO80] Herot, C., ‘‘SDMS: A Spatial Data Base System,’’ *ACM TODS* (1980).

[KATZ89] Katz, R., et al., ‘‘Disk System Architectures for High Performance Computing,’’ *Proceedings of the IEEE, Special Issue on Supercomputing*

(December 1989).

[KARL89] Karlson, G. and Vetterli, M., ''Packet Video and its Integration into the Network Architecture,'' *IEEE Journal on Selected Areas in Communications,* vol. 8, pp. 380-390 (1990).

[KATZ92]

[KIM90] Kim, W. et al., ''Architecture of the ORION Next-Generation Database System,'' *IEEE Transactions on Knowledge and Data Engineering* (March 1990).

[LARS91] Larson, R., ''Classification Clustering, Probabilistic Information Retrieval and the Online Catalog,'' *Library Quarterly,* vol. 61 (April 1991).

[LELE87] Lelewer, D. and Hirschberg, D., ''Data Compression,'' *ACM Computing Surveys*, vol. 19 (September 1987).

[LYNC88] Lynch, C. and Stonebraker, M., ''Extended User-Defined Indexing with Application to Textual Databases,'' *Proceedings 1988 VLDB Conference,* Los Angeles (1988).

[MARK91] Markoff, J., ''A System to Speed Computer Data ,'' *New York Times,* p. C7 (January 23, 1991).

[MULL91] Muller, K. and Pasquale, J., "A High Performance Multi-structured File System Design," Proc. 13th ACM Symposium on Operating Systems Principles, Pacific Grove, Ca., October 1991.

[PASQ92] Pasquale, J. et. al., "Internet Throughput and Delay Measurements between Sequoia 2000 Sites", (in preparation).

[PATT88] Patterson, D. et al., ''A Case for Redundant Arrays of Inexpensive Disks,'' *Proceedings 1988 ACM-SIGMOD Conference on Management of Data*, Chicago, IL (1988).

[RANA90] Ranade, S. and Ng, J., *Systems Integration for Write-Once Optical Storage,* Meckler, Westport, CT (1990).

[ROSE91] Rosenblum, M., and Ousterhout, J., ''The Design and Implementation of a Log-Structured File System,'' submitted for publication, February 1991.

[STON90] Stonebraker, M. et al., ''The Implementation of POSTGRES,'' *IEEE Transactions on Knowledge and Data Engineering* (March 1990).

[STON91] Stonebraker, M., ''Managing Persistent Objects in a Multi-level Store,'' *Proceedings 1990 ACM SIGMOD Conference on Management of Data*, Denver, CO (1990).

[STON92] Stonebraker, M,. and Olson, M., "Large Object Support in POSTGRES," (in preparation).

[WILK90] Wilkinson, K. et al., ''The IRIS Architecture and Implementation,'' *IEEE Transactions on Knowledge and Data Engineering* (March 1990).