Bigfoot: An Earth Science Computing Environment for the Sequoia 2000 Project

James Frew

Abstract

The Sequoia 2000 Project is a large-scale collaboration between the Digital Equipment Corporation, the University of California, and several industrial partners and government agencies, for developing new computing environments for global change research. The primary focus of the Project is to develop solutions for massive data storage, data access, data analysis and visualization, and wide-area networking. These solutions are embodied in ‘Bigfoot,’ a computing environment the Project is building at the Berkeley campus. Bigfoot comprises 10 terabytes of tertiary storage supporting commercial and experimental file systems, managed by an extended relational database management system supporting Earth science data types and operations. Bigfoot is linked to a complex of private high-speed local and wide area networks, running both standard and experimental protocols. Research projects are extending Bigfoot to couple directly to general circulation models, and to provide visualization, full-text retrieval, and geographic information system capabilities.

Introduction

This chapter discusses ‘Bigfoot,’ the computing environment developed for the Sequoia 2000 Project at the University of California (UC). It begins with a very brief overview of the Sequoia 2000 Project, followed by a discussion of the computing problems that Bigfoot was designed to address. The architecture of the current implementation of Bigfoot is then described in some detail.

What is the Sequoia 2000 Project?

The Sequoia 2000 Project is a three-year collaboration between: computer and Earth scientists at UC; the Digital Equipment Corporation (DEC); other industrial sponsors; and several state and federal agencies. The overall goal of Sequoia 2000 is to further the state of the art in data management for global change research. The Project has developed several interpretations of this goal, owing largely to the diversity of the Sequoia 2000 community.

For example, Sequoia 2000 computer scientists view the Project as an engineering effort to develop a data system capable of storing, retrieving, and manipulating several terabytes of heterogeneous Earth science data (one terabyte = one trillion bytes). The Earth scientists are viewed as ‘clients,’ providing system specifications and testing the various prototypes.

Sequoia 2000 Earth scientists view the Project as an opportunity to apply massive data management resources to previously intractable problems. Since they are geographically distributed throughout California, they also use the Sequoia 2000 networked computing environment as a ‘collaboratory,’ furthering the kinds of interdisciplinary research essential to global change studies (National Research Council Committee on Global Change, 1990).

Finally, the Sequoia 2000 industrial and governmental partners view the Project as an opportunity to share new technologies. Sequoia 2000 is a source of new ideas for products and services, as well as a sophisticated community of beta-testers for products and services that have reached the development stage.

Who is the Sequoia 2000 Project?

The Sequoia 2000 Project is organized around a partnership between DEC and UC. DEC is Sequoia 2000’s primary sponsor, contributing about $5 million per year in direct funding and equipment credits, plus four full-time staff positions in engineering and management. Sequoia 2000 is DEC’s ‘flagship’ external research project, the successor to Project Athena (Champine, 1991), but with a much different focus: Athena concentrated on workstations and local-area networks, whereas Sequoia 2000 concentrates on data management and wide-area networks.

Five UC campuses are involved in Sequoia 2000. The Project is headquartered in the Computer Science Division at UC Berkeley. Most of the computer science and engineering activities are concentrated at Berkeley, with significant contributions from the Computer Science Department at UC San Diego and from the San Diego Supercomputer Center. Earth scientists participating in Sequoia 2000 are affiliated with the Scripps Institution of Oceanography at UC San Diego; the Atmospheric Sciences Department at UC Los Angeles (UCLA); the Center for Remote Sensing and Environmental Optics at UC Santa Barbara; and the Department of Land, Air, and Water Resources at UC Davis.

Several government agencies sponsor Sequoia 2000. Most of them (California State Resources Agency, National Aeronautics and Space Administration, National Oceanic and Atmospheric Administration, US Geological Survey) face data management challenges that could possibly be addressed by early adoption of some Sequoia 2000 technologies. Others (US Army Corps of Engineers) are integrating their own data management technologies into Bigfoot.

Besides DEC, there are several industrial sponsors of Sequoia 2000. They currently include Epoch Systems Inc., Hewlett-Packard Co. (HP), Hughes Aircraft Co., MCI, Metrum Information Storage, North Carolina Supercomputer Center, PictureTel Corp., Research Systems Inc. (RSI), Science Applications International Corp. (SAIC), Siemens Corporate Research Inc., and TRW. Besides contributing direct financial support or products, the industrial partners provide invaluable feedback on the viability of the technologies being developed by Sequoia 2000.

Shortcomings of Current Computing Environments

The development of Bigfoot has been driven by the inability of current computing environments to accommodate the demands of Sequoia 2000 Earth scientists, particularly in the following areas (Stonebraker, 1993):

- storing huge datasets
- accessing huge datasets

- analyzing and visualizing complex data
- connecting remote investigators and data

This section illustrates these shortcomings and some of their implications.

Storing huge datasets

Sequoia 2000 Earth scientists must routinely deal with datasets far too large to keep online in their local computing environments. To understand the magnitude of this problem, consider a computing environment consisting of a file server with 10 gigabytes of disk storage, on a network of 10 workstations, each with one gigabyte of local storage (one gigabyte = one billion bytes). In 1993, this is a reasonable ‘off-the-shelf’ scientific computing system.

Now consider some datasets typical of those used by Sequoia 2000 investigators:

Table 1 goes here

Only the smallest of these may be kept online in the typical computing environment specified; the others must be stored offline and processed in (relatively) small pieces. The larger the dataset, the more physical volumes it will consume (e.g. 3000 gigabytes = 600 8mm tapes), and thus the more human intervention (fetching, mounting and unmounting volumes) that will be required to process the dataset. The delays introduced by requiring a human operator in the middle of a processing sequence can profoundly discourage attempts to process huge datasets.

Accessing huge datasets

Not only are Earth science datasets often huge, but they are of many different types. Some of the data types used by Sequoia 2000 Earth scientists are:

Table 2 goes here

A typical analysis may involve combining data of several types (Davis, 1992). Furthermore, the criteria by which the data are selected may involve both metadata (static attributes of the entire dataset; e.g.

date and time of acquisition) and the data values themselves. For example, consider the variety of data types and manipulations necessary to satisfy a request for ‘all ocean color values for the melting margin of the ice pack beneath the Antarctic ozone hole for the dates nearest 15 October 1991.’

To extract a portion of a large dataset using current computing environments requires mounting offline physical volumes and then scanning them to extract a region of interest. If the organization of the desired subset is a transposition of the dataset (e.g. multispectral values extracted from a sequence of single-band images) then the entire dataset may have to be mounted and scanned. If several datasets are involved, the search time multiplies, with additional overhead if the results have to be merged.

Of course, locating and gaining access to large datasets in the first place can be a considerable challenge. Datasets are usually organized logically into files and physically into volumes. The file and volume divisions may, but need not, correspond to some logical properties of the dataset (e.g. single image). The point is that file and volume names describe only how the data are stored, not any attributes relevant to the data themselves. Yet in most current environments these are the only metadata available for large datasets.

Collectively, these data access restrictions make it hard to find data of interest and difficult to extract them from the dataset. Many analyses that are conceptually simple are never begun because it is so difficult to obtain the relevant portions of huge datasets.

Analysis and visualization

Earth science data analysis in current computing environments typically involves the successive application of several barely-compatible tools, each of which may use a different data format, different units, etc. Users must spend much time both converting data as they flow from one tool to the next, and tracking the sequence of tools applied to each dataset. For many Earth scientists, this bookkeeping is a major share of their data analysis activities.

Visualization tools are becoming a critical part of the Earth scientist’s analytical toolkit. Yet current visualization tools have two distinct shortcomings. First, they are large, slow, cumbersome software systems. This is partly because visualization is an inherently complex activity, but also because visualization packages must perform substantial data management activities (e.g. keeping track of files). In any event, the complexity of current visualization software makes it difficult to use effectively. Many Earth scientists must employ specially trained programmers to operate their visualization software.

A second shortcoming of many current visualization tools is that they are output-only − the screen, film recorder, or whatever is viewed as a data sink. This makes these tools suitable for preparing publication graphics (certainly an important activity!) but less useful for the kind of interactive processing necessary to refine, or even direct, an analysis sequence. For example, if a visualization tool shows that a general circulation model (GCM) is predicting boiling-water surface temperatures over North America, then it would be desirable to use the visualization tool to probe the executing GCM and discover which parameter was causing the (apparent) error.

The current state of data analysis and visualization causes much time to be wasted ‘gluing’ incompatible stand-alone tools together. There are no automatically-maintained audit trails to keep track of how each dataset is modified by specific tools. Furthermore, since most visualization is done in what amounts to a ‘batch’ mode, problems or opportunities arising early in the analysis sequence cannot be visually detected until the entire analysis is complete.

Connecting remote investigators and data

Sequoia 2000 Earth scientists are currently based at 4 UC campuses, the furthest distant of which are separated by several hundred miles. Internet connections between these sites, while adequate for electronic mail and remote logins, are too slow and erratic to be used for sustained high-volume data transfers (Pasquale, 1991). Insufficient network capacity significantly impedes the scientists’ ability to share both computing resources and data. The latter is more important, since while a constant unit of computing power continues to plummet in price, large storage systems remain relatively expensive, and thus less likely to be part of one’s local computing environment. A reliable, high-throughput wide-area network (WAN) is thus critical to Sequoia 2000.

Bigfoot Architecture and Implementation

In response to the concerns outlined above, the Sequoia 2000 Project has implemented the Bigfoot computing environment. This section describes the current implementation of Bigfoot. Figure 1 gives the overall system architecture.

Figure 1 goes here

The storage layer

Current computing environments all have at least a two-level storage hierarchy, with primary memory (RAM) at the top and magnetic disk underneath. Bigfoot adds a third layer (tertiary memory) to the bottom of this hierarchy, comprising various persistent storage media (tapes, optical disks, etc.) configured as multi-volume ‘jukeboxes’ with robotic media manipulation. These robotic jukeboxes are called ‘near-line’ storage (Katz, 1992), since any volume may be brought online subject only to the latency of the robotics (e.g. the time for a robot arm to fetch and load a tape), as opposed to the much greater and more variable latency of a human exchanging tapes in a tape drive.

Bigfoot currently supports about 40 gigabytes of magnetic disk storage (the number fluctuates as workstations are added to or removed from the network) and 10 terabytes of tertiary storage, on the following devices:

Table 3 goes here

The Footprint layer

The Sony and Exabyte jukeboxes are managed by the Project-developed ‘Footprint’ software. Footprint is a generic program interface for robotic storage devices. As such, Footprint provides a uniform interface to devices with often wildly varying physical characteristics (robot commands, tape sizes, etc.). These device-specific details are normally concealed by Footprint but may be accessed if needed (for example, some devices provide a local ‘duplicate media’ command which is useful for regenerating worn tapes.)

The file system layer

The Project supports four independent filesystem interfaces. The HP and Metrum jukeboxes are managed by the commercial packages EpochServ and UniTree, respectively. Both of these products are ‘hierarchical storage managers,’ which means that they invisibly migrate files between the jukeboxes and magnetic disk caches; in effect, they make each jukebox look like a single huge disk drive.

The Exabyte jukebox is managed by the locally-developed ‘Jaquith’ software package which makes the jukebox look like a single huge tape drive. The Sony jukebox is managed by the ‘Inversion’ file system, described in the next section.

The database management system (DBMS) layer

Any data stored on Bigfoot may, at the owner’s discretion, be managed by the POSTGRES database management system, an extended relational DBMS developed at UC Berkeley (Stonebraker, 1991). At a minimum, dataset owners associate metadata with their datasets. POSTGRES can then use these metadata to do attribute-based retrieval of portions of huge datasets. A POSTGRES ‘retrieve’ operation may reference multiple datasets. If portions of more than one dataset are returned, a new data structure (‘class’) is automatically created to hold them.

In addition to the usual relational DBMS capabilities, POSTGRES provides some major additional functionality of particular importance to the management of Earth science data:

- *large objects*: POSTGRES supports binary large objects of unlimited size, and the ‘Inversion’ file system (Olson, 1993) to access their contents. The name Inversion is derived from the fact that this is an ‘upside-down’ file system − the file system is implemented on top of the database, rather than vice-versa. Inversion offers the standard UNIX file system access functions (read, write, lseek, etc.), and supports both File Transfer Protocol (FTP) and Network File System (NFS) servers. Inversion combines the familiar, efficient file-level access model with all the benefits of a DBMS: transaction protection, time travel (recovery of old versions), audit trails, plus access to the file and its contents by POSTGRES queries and functions.

- *Footprint*: POSTGRES uses Footprint to directly access tertiary storage, bypassing any file systems. The POSTGRES query optimizer can then take into account the peculiar latency and throughput characteristics of tertiary storage devices.
- *User-specified types*: In addition to the usual scalar types supported by any DBMS (characters, integers, text strings, floating-point numbers, etc.), POSTGRES allows users to define their own types and associated functions. These types may be implemented as large objects, if necessary. Multidimensional arrays, polygons, and physical quantities such as temperature are some of the types added to POSTGRES for Sequoia 2000. These types may also have multiple representations. For example, the type ‘temperature’ might be defined with a ‘units’ component that indicates the units of measure (Fahrenheit, Celsius, etc.). Functions defined on temperature may then consult the units to determine whether a conversion is necessary for the current operation, and if so, do the conversion transparently.
- *User-specified functions*: Users may define functions that operate on either built-in or user-specified types. These functions may be written in the POSTQUEL query language, in which case they are stored in POSTGRES, or they may be written in C and compiled, in which case they are dynamically loaded when they are referenced. POSTGRES functions have access to the internals of large objects, thereby enabling data-intensive operations to be done entirely within the DBMS. The return value of a function (built-in or user-defined) may be used as a selection criterion in a query; this is how a large object may be queried by content as well as by metadata. Indices may also be built on function values to dramatically accelerate such queries.

By using POSTGRES for data access, Bigfoot users are freed from having to remember arbitrary file and volume names and how they map into (portions of) datasets. Collectively, the POSTGRES extensions support a new model of scientific computing in which the DBMS is treated less as a *data* server, and more as a *procedure* server; i.e. in addition to asking the DBMS for data, the user may also ask it for results.

Example: image data retrieval: Here is a simple example that illustrates some of the capabilities of POSTGRES on Bigfoot. Assume a query of the form:

> I'm studying seasonal vegetation patterns in southern California. I want AVHRR NDVI data for a certain date, within a geographic rectangle corresponding to my study area.

The following POSTQUEL query would extract the AVHRR sub-images from the relevant images, and leave the new sub-images in Inversion files. The return value of the query is a list of the new Inversion file names.

```
retrieve(
        clip_avhrr(
                AVHRR_table, output_format,
                northwest corner, southeast corner
        )
) where AVHRR_table.date = ...
```

The implementation of the query is straightforward. The AVHRR images themselves are stored in POSTGRES large objects as Inversion files. The metadata (in this simple case, the data format, date of acquisition, and geographic bounding box) are stored in the class 'AVHRR_table' which also contains a reference to the appropriate Inversion file. The function 'clip_avhrr' is user-defined function that returns instances of class AVHRR_table.

POSTGRES and analysis: Data analysis and visualization on Bigfoot exploits several features of POSTGRES. Many of the functions normally invoked as tools are built into POSTGRES, so much of a typical analysis is done inside the DBMS. To do functions that aren't built-in, POSTGRES can invoke an external tool. In such cases, POSTGRES is told what data formats the tool reads and writes, and then does the necessary data conversions invisibly. All function invocations, internal and external, are tracked using standard DBMS logging mechanisms, which permit data recovery at any stage of the processing.

POSTGRES facilitates visualization by assigning a ‘renderable form’ to each data type that is likely to be visualized. The renderable form is a specific kind of metadata which an external rendering tool can use to determine how the data ought to be displayed. The ‘Tioga’ effort, described in the next section, is building a renderer that incorporates feedback to POSTGRES, allowing manipulators (keyboard, mouse, etc.) in the renderer to provide input to functions executing in POSTGRES.

POSTGRES communicates with existing tools in two ways. The first, already described, involves POSTGRES invoking the tool with data formats that the tool expects. The second involves modifying the tool to talk to POSTGRES, typically by issuing queries to obtain data instead of opening files. Major existing tools which have already been modified in this fashion include the commercial visualization packages AVS and IDL, the ‘S-plus’ commercial statistics package, and the public-domain GRASS geographic information system (GIS) (Westervelt, 1988).

Data analysis on Bigfoot is still complicated, but POSTGRES does many of the more tedious tasks, freeing the scientist to concentrate on science instead of data reformatting. POSTGRES internal functions minimize data transfer outside the DBMS. The use of renderable forms as data attributes allows almost any data in the DBMS to be visualized, at any point during an analysis sequence.

The network layer

All Bigfoot components are networked: the DBMS, the file systems and tertiary storage devices, and the display systems. Standard protocols like NFS and the X window system are used where applicable. The Bigfoot networking strategy is to make all services available to all users. The network should not be the transfer bottleneck; remote users should receive the same level of service as local users.

The Sequoia 2000 Project maintains a private WAN, comprising T1 (1.54 Mbit / sec) connections (donated by MCI) linking the San Diego, Los Angeles, Santa Barbara, Berkeley, and Davis campuses, and the California Department of Water Resources. The WAN will be upgraded to T3 (45 Mbit / sec) service during November 1993. Each site has a gateway between the WAN and a local FDDI (100 Mbit / sec) fiber-optic ring, to which the investigators’ workstations are connected.

The Sequoia 2000 network is being used as a testbed for a new suite of protocols called RTIP (Ferrari, 1990), developed at UC Berkeley. The RTIP protocols enable ‘guaranteed delivery,’ whereby a client program may reserve a fixed portion of network bandwidth. This is important for continuous-media applications (sound, video) where the delay between successive transfers is critical to the application’s performance. Sequoia 2000 researchers are using RTIP, together with prototype video compression hardware, to develop a desktop teleconferencing system that can be installed in any Sequoia 2000 investigator’s workstation.

The application layer

This section discusses some of the more ambitious applications being developed on Bigfoot.

Tioga: ‘Tioga’ (Stonebraker, 1992) is an attempt to build an integrated programming and visualization system within and on top of POSTGRES. Tioga has three major components:

- *recipes*: Tioga allows users to build programs or ‘recipes’ out of POSTGRES functions and to store those recipes in POSTGRES.
- *graphical programming and query language*: Tioga includes a graphical programming environment analogous to AVS, in which functions are depicted as boxes and data flows by directed lines. Programs built graphically are stored as recipes. A skeletal program, with some or all boxes replaced by regular expressions, may be used to query POSTGRES and retrieve any recipes whose structure matches the skeleton.
- *smart renderer*: Tioga will send renderable forms of recipe outputs over a network connection to a ‘smart renderer’ being built at the San Diego Supercomputer Center. In addition to the usual mechanisms for visualizing geometry and images, the smart renderer will incorporate a knowledge base allowing it to suggest appropriate representations for specific types of objects (e.g. use of a thermometer as an icon for temperature).

The Big Lift: The ‘Big Lift’ is a project to connect the UCLA GCM directly to POSTGRES; i.e. the output from the GCM will be deposited directly into the DBMS, without being saved in intermediate files. This involves modifying both the GCM (replacing its output routines) and POSTGRES (enabling it to accept data at the GCM’s output rate). The name ‘Big Lift’ is borrowed from the large pumping plant in the California state water system that pumps water from Northern California over the Tehachapi Mountains to Southern California.

There are two expected benefits from the Big Lift. The first and most immediate is the ability to browse the intermediate output of the GCM while it is running. This will allow the user to detect errors and adjust parameters or restart the model, using POSTGRES to extract subsets from the GCM’s voluminous output. The second benefit, the ability to visually control the GCM’s execution, is a longer-term goal that depends on the completion of Tioga. If successful, this should radically change the way GCMs are used.

Post-GRASS: With the cooperation of the U.S. Army’s Construction Engineering Research Laboratory (the originators of the GRASS GIS), the Sequoia 2000 Project is working on subsuming the functionality of GRASS into POSTGRES. The first phase of this effort, already complete, has replaced the input/output subsystem of GRASS with calls to POSTGRES. The relevant GRASS external data structures have been replaced by a custom POSTGRES schema. This ‘GRASS-on-top-of-POSTGRES’ has been dubbed ‘Post-GRASS.’

The second phase of Post-GRASS, currently in progress, will migrate specific GRASS commands into POSTGRES as internal functions. The GRASS user interface will then be modified to issue POSTQUEL queries instead of UNIX commands. In the third phase, Tioga will replace the GRASS user interface, and POSTGRES will be able to provide equivalent functionality to GRASS, but with access to all the data in Bigfoot.

Lassen: ‘Lassen’ is a full-text document retrieval system built into and on top of POSTGRES. Text pages are stored as large objects, with weighted keyword indices (Larson, 1991) built automatically by POSTGRES as the text is entered. Lassen includes a separate natural-language query tool that interfaces to

POSTGRES, allowing text retrieval in a manner familiar to users of automated library catalogs.

Lassen will be used to access a growing online collection of text on Bigfoot. The Computer Science Division at UC Berkeley is currently scanning all their technical reports into Bigfoot, saving the text as images and using optical character recognition (OCR) to extract keywords. Eventually all printed material associated with Sequoia 2000 will be accessible via Lassen.

Conclusions

Bigfoot addresses several pressing needs in Earth science data management, and does so in novel ways. Robotic tertiary storage, POSTGRES, and a fast Project-wide network combine to both speed and simplify access to previously unmanageable quantities of data. POSTGRES manages both metadata *and* data, thus freeing users from worrying about low-level file management, and allowing them to instead concentrate on the data's information content.

Bigfoot is also well-positioned to address some of the major future challenges in Earth science data management. Bigfoot's seamless integration of storage, database, and networks is a prototype of the transparent, distributed environments that will be commonplace in the next century. The tight coupling of data and functions provided by POSTGRES enables new modes of analysis that the Project believes will become increasingly important.

Sources of Further Information

Further information about Bigfoot and the Sequoia 2000 Project may be obtained via anonymous FTP to the Internet host ‘toe.cs.berkeley.edu’. A Sequoia 2000 software distribution is currently planned for late 1993 which will include POSTGRES, Inversion, Footprint, RTIP, and any other Project-developed software. Email correspondence regarding the Project may be addressed to ‘frew@crseo.ucsb.edu’.

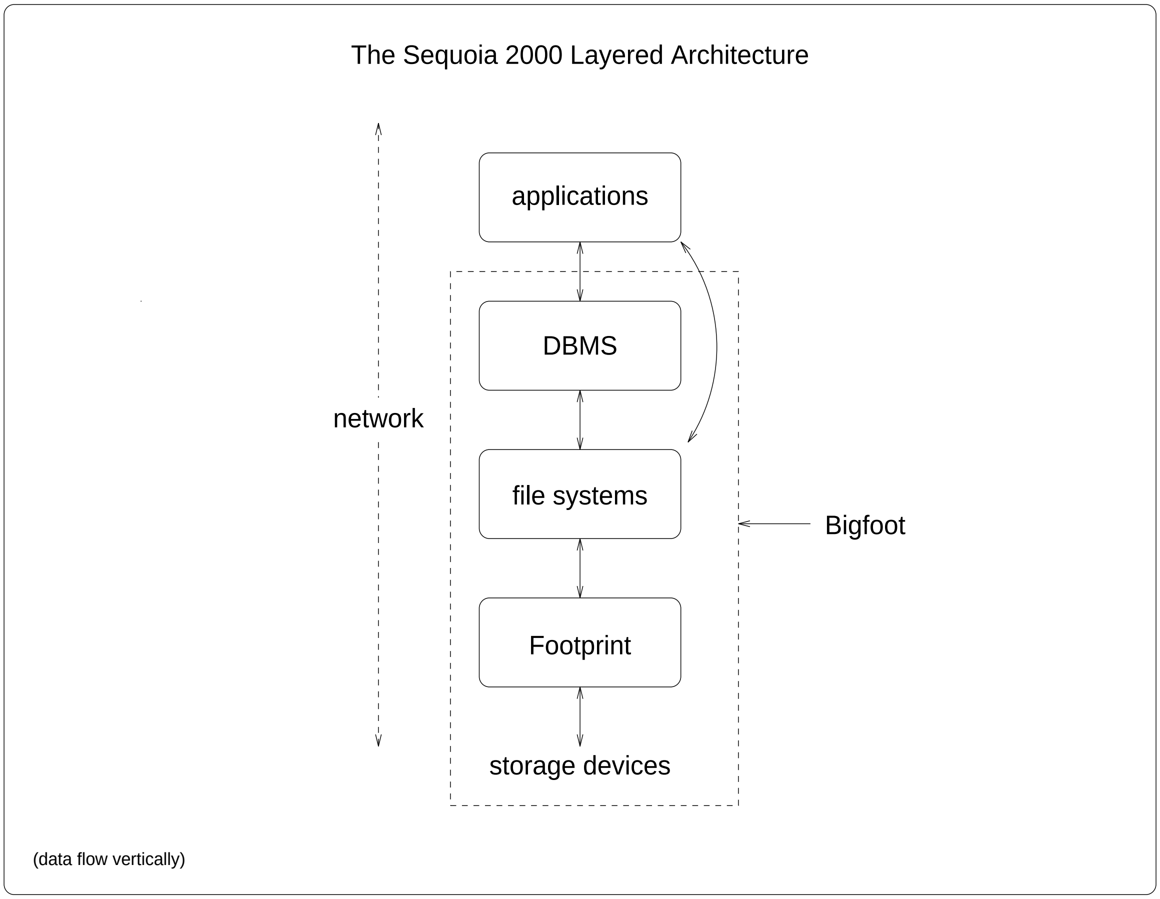
The Sequoia 2000 Layered Architecture
network
applications
DBMS
file systems
Footprint
storage devices
Bigfoot
(data flow vertically)

Figure 1

gigabytes	dataset
4	normalized-difference vegetation index (NDVI) from Advanced Very-High Resolution Radiometer (AVHRR) (Eidenshink, 1992) (one year of biweekly composite images; contiguous US)
23	UCLA hybrid coupled general circulation model output (Neelin, 1990) (one simulated year; global)
675	all Coastal Zone Color Scanner raw data ever collected (Feldman, 1989)
3000	AVHRR level 1B ‘Pathfinder’ dataset (Wiscombe, 1992) (daily 1981-present; global)

Table 1

type	examples
raster	satellite image; digital elevation grid
vector	drainage basin boundary; river network
point	weather station data; river discharge
text	algorithm description; instrument manual

Table 2

device	capacity (gigabytes)
HP magneto-optical disk jukebox	100
Sony write-once optical disk (WORM) jukebox	360
Exabyte 8mm tape jukebox	580
Metrum VHS tape jukebox	9000

Table 3

References

Champine, 1991.

G.A. Champine, *MIT Project Athena : A Model for Distributed Campus Computing,* Digital Press, Bedford, MA (1991).

Davis, 1992.

F. Davis, D. Schimel, M. Friedl, J. Michaelsen, T. Kittel, R. Dubayah, and J. Dozier, “Covariance of biophysical data with digital topographic and land-use maps over the FIFE site,” *Journal of Geophysical Research,* 97, D17, pp. 19,009-19,021 (1992).

Eidenshink, 1992.

J. Eidenshink, “The 1990 conterminous U.S. AVHRR data set,” *Photogrammetric Engineering and Remote Sensing,* 58, 6, pp. 809-813 (June 1992).

Feldman, 1989.

G. Feldman, “Ocean color: availability of the global data set,” *Eos,* 70, 23, pp. 634-640 (06 June 1989).

Ferrari, 1990.

D. Ferrari, “Client requirements for real-time communication services,” *IEEE Communications Magazine,* 28, 11 (1990).

Katz, 1992.

R. Katz, “High-performance network and channel based storage,” *Proceedings of the IEEE,* 80, 8, pp. 1238-1261 (August 1992).

Larson, 1991.

R. Larson, “Classification, clustering, probabilistic information retrieval and the Online Catalog,” *Library Quarterly,* 61, 2, pp. 133-173 (April 1991).

National Research Council Committee on Global Change, 1990.

National Research Council Committee on Global Change, *Research Strategies for the U.S. Global*

Change Research Program, National Academy Press, Washington, DC (1990).

Neelin, 1990.

J.D. Neelin, “A Hybrid Coupled General Circulation Model for El-Nino Studies.,” *Journal of the Atmospheric Sciences*, 47, 5, pp. 674-693 (March 1990).

Olson, 1993.

M. Olson, “The design and implementation of the Inversion File System” in *USENIX Association Winter 1993 Conference Proceedings,* San Diego (January 1993).

Pasquale, 1991.

J. Pasquale, G. Polyzos, K. Fall, and V. Kompella, “Internet throughput and delay measurements between Sequoia 2000 sites,” Sequoia 2000 Technical Report 91/7, University of California, Berkeley (Dec 1991).

Stonebraker, 1992.

M. Stonebraker, J. Chen, N. Nathan, and C. Paxson, “Tioga: providing data management support for scientific visualization applications,” Sequoia 2000 Technical Report 92/20, University of California, Berkeley (Dec 1992).

Stonebraker, 1993.

M. Stonebraker, J. Frew, and J. Dozier, “The Sequoia 2000 Architecture and Implementation Plan,” Sequoia 2000 Technical Report 93/23, University of California, Berkeley (May 1993).

Stonebraker, 1991.

M. Stonebraker and G. Kemnitz, “The POSTGRES next-generation database management system,” *Communications of the ACM,* 34, 10, pp. 78-92 (October 1991).

Westervelt, 1988.

J. Westervelt, “An Introduction to GRASS,” program documentation, U.S. Army Construction Engineering Research Laboratory (August 1988).

Wiscombe, 1992.

W. Wiscombe, *(personal communication)*, NASA Goddard Space Flight Center, Greenbelt, MD (Dec 1992).