Software Distribution Using Xnetlib

JACK DONGARRA
University of Tennessee and Oak Ridge National Laboratory

TOM ROWAN
Oak Ridge National Laboratory
and
REED WADE
University of Tennessee

Xnetlib is a new tool for software distribution. Whereas its predecessor netlib uses email as the user interface to its large collection of public-domain mathematical software, xnetlib uses an X Window interface and socket-based communication. Xnetlib makes it easy to search through a large distributed collection of software and to retrieve requested software in seconds.

Categories and Subject Descriptors: C.2.3 [**Computer-Communication Networks**]: Network Operations—*public networks*; D.2.2 [**Software Engineering**]: Tools and Techniques—*software libraries*; *user interfaces*; D.2.7 [**Software Engineering**]: Distribution and Maintenance—*documentation*; *portability*; G.1.0 [**Numerical Analysis**]: General—*numerical algorithms*; G.4 [**Mathematics of Computing**]: Mathematical Software—*portability*; H.3.0 [**Information Systems Applications**]: Communications Applications; H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*search process*; *selection process*; H.3.5 [**Information Storage and Retrieval**]: Online Information Services—*databank sharing*; H.5.2 [**Information Interfaces and Presentation**]: User Interfaces—*windowing systems*; K.6.3 [**Management of Computing and Information Systems**]: Software Management—*software development*; *software maintenance*; *software selection*

General Terms: Algorithms, Documentation, Management

Additional Key Words and Phrases: Netlib, software repositories

1. INTRODUCTION

Xnetlib is a new software distribution tool recently developed at the University of Tennessee and Oak Ridge National Laboratory. The goal in developing

This work was supported in part by ARPA and ARO under contract DAA 03-91-C-0047, and in part by the Applied Mathematical Sciences subprogram of the Office of Energy Research, U.S. Department of Energy, under contract DE-AC05-84OR21400.

Authors' addresses: J. Dongarra and R. Wade, Department of Computer Science, University of Tennessee, Knoxville, TN 37996-1301; T. Rowan, Oak Ridge National Laboratory, Oak Ridge, TN 37831-6367.

ACM Transactions on Mathematical Software, Vol. 21, No 1, March 1995, Pages 79–88.

xnetlib was to provide Internet users faster and easier access to netlib's large collection of software, data, and documents. Unlike netlib, which uses email to process requests for software, xnetlib uses an X Window interface and socket-based communication between the user's machine and the xnetlib server[1] machine to process software requests. This enables users to search through a large distributed collection of software easily and to retrieve requested software in seconds.

2. BACKGROUND

Xnetlib's predecessor, netlib, grew from a need to have a quick and easy method for distributing small pieces of mathematical software. Netlib services began in 1985 at two sites, Argonne National Laboratory and AT & T Bell Laboratories, and distributed software from about 30 libraries. For additional information about netlib's operation and use, see the introductory paper by Dongarra and Grosse [1987].

One of the changes since netlib's introduction has been the transfer of netlib services from Argonne National Laboratory to Oak Ridge National Laboratory. Also, the availability on netlib of the netlib program itself has enabled many other sites to set up their own software repositories.[2] The netlib software collection has now grown to over 150 libraries. The number of software requests sent to netlib also has grown dramatically. The most heavily used netlib server, at Oak Ridge National Laboratory, processed over 147,000 requests in 1993.

Xnetlib use has grown even more rapidly. In 1993, its third year of operation, xnetlib processed over 233,000 requests.

3. OVERVIEW

We based xnetlib's design on the following requirements:

Speed. Retrieving software should take seconds, not minutes as typically required by email.

Usability. The user interface should make searching through a large collection of software and documents easy.

Organization. The system's repository should be a moderated collection, with up-to-date indexes and a database organized to facilitate searching and ease of retrieval. The repository may be distributed over several sites.

Record Keeping. The system should have the capability of logging requests so updates and corrections can be reported to users.

Security. The system should be secure from accidental or intentional misuse.

[1] Throughout this paper *server* refers to the process-handling software requests and not to the X display server.

[2] Send the message **send sites from netlib** to netlib@ornl.gov to receive a list of **netlib** sites.

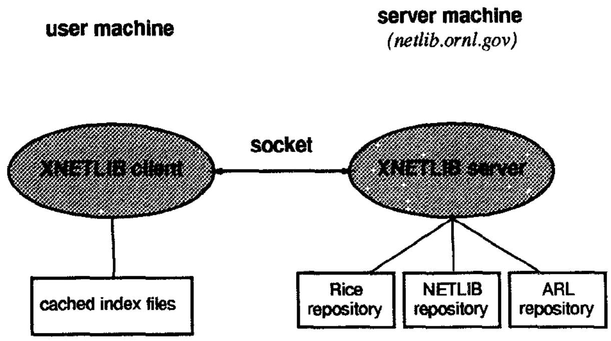


Fig. 1. **Xnetlib** configuration.

Portability. The system's implementation should be as portable as possible.

Accessibility. The system should be accessible to a large number of users.

There already exists a variety of tools for Internet browsing and for software and document retrieval [Krol 1992; Studt 1993]. Archie, ftp, gopher, mosaic, WAIS, and WorldWideWeb are among the most widely used.

In contrast to these other tools, xnetlib operates on a limited, moderated collection that contains the most up-to-date version of software packages. This approach of retaining control over the size and contents of xnetlib's repositories simplifies greatly any organization, record keeping, and security. Xnetlib uses socket-based communication to provide the desired speed and a client design based on X Windows for portability and usability. The accessibility requirement is satisfied by making the client software freely available over the Internet.

Figure 1 shows the basic configuration of the xnetlib system. The system consists of an xnetlib server process running on a machine at a repository site, an xnetlib X client process running on a user's local machine, and TCP/IP socket-based communication links between the client and server.

Xnetlib's server process runs continually at an xnetlib repository site, listening for incoming requests from xnetlib client processes. Typically, the xnetlib server runs on the same machine and accesses the same software repository as the netlib server. Upon receiving a request, the xnetlib server determines the nature of the request and responds by transferring the appropriate file from the repository to the xnetlib client process.

The xnetlib client running on the user's local machine provides an X Window interface to the xnetlib repository. It is programmed in C, using the Athena widget libraries. This interface makes searching through the software and document collection easy. For example, an xnetlib user can view the contents of any library simply by clicking a button. Other commands, such as key word searching or requesting software, also require just a few button clicks. To avoid unnecessary communication between the client and server, requested indexes are cached locally. Frequently requested information can therefore be quickly retrieved from local cache instead of repeatedly retrieved from the remote server. Other requests are passed to the server via sockets. Section 4 describes the use of the interface in more detail.

4. FEATURES

Xnetlib features and capabilities include

—access to a distributed repository,
—searching by a software libraries list,
—searching by software classifications,
—searching by key word,
—software and document retrieval,
—access to a computer performance database,
—access to a conferences database,
—access to the NA-NET White-pages, and
—online help.

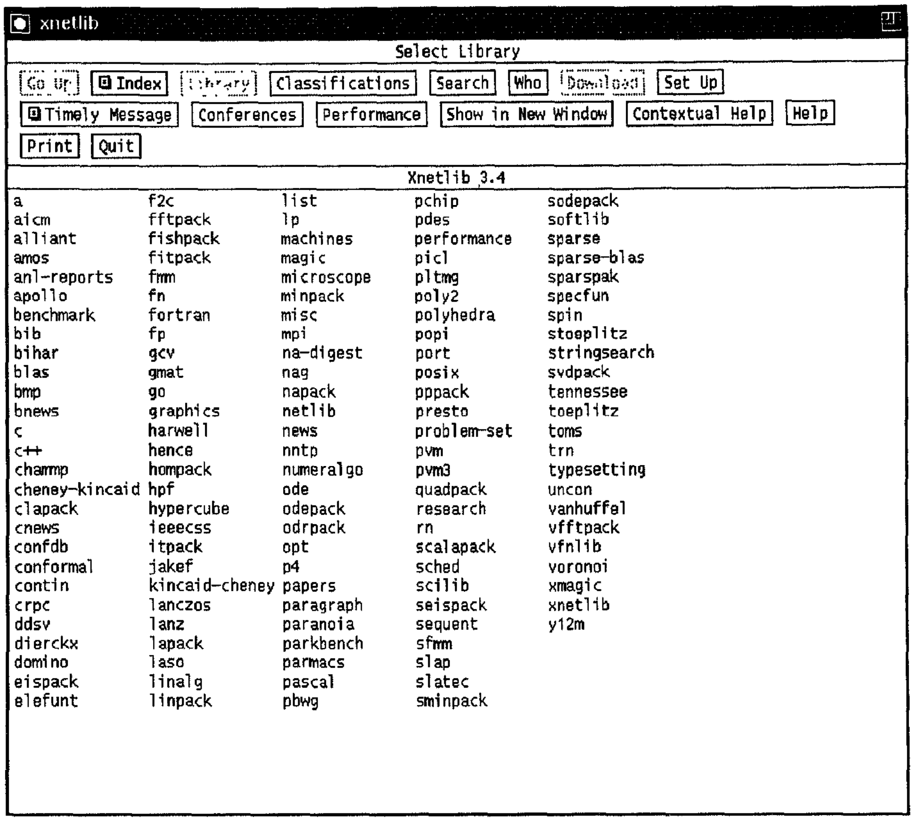
Many Internet sites have sizable collections of documents or software. It is both unnecessary and undesirable to require that these collections reside at a single site. Xnetlib gives users access to a distributed repository of software and documents by establishing socket-based links with the xnetlib repository sites. Users have access to any or all of these repositories through a single interface.

Xnetlib users control which sites are linked into the distributed repository using the **set up** button. Current repository sites include netlib.ornl.gov, spark.brl.mil, and softlib.rice.edu. The **timely message** button displays news about individual repository sites, and the **index** button displays their general indexes.

The **library** button shows the libraries that are available through xnetlib. The collection includes major packages such as LAPACK, LINPACK, EISPACK, algorithms published in the *ACM Transactions on Mathematical Software*, and a large variety of less well-established packages. Figure 2 shows a unified list of software and documents available at the repositories of Oak Ridge National Laboratory, Rice University, and the U.S. Army Research Laboratory.

Clicking on a library name lists the contents of that library. For example, clicking on **lapack** displays a partial listing of LAPACK's contents (Figure 3). The complete contents list of LAPACK is too large to fit in the window, but the provided scrollbar allows the user to scroll through the rest of the list. Clicking on the box adjacent to a routine name selects that routine for future downloading. In Figure 3 the user has selected **sgetrf** and **sgetrs** from the LAPACK library.

The **classifications** feature allows a user to narrow a search. The classification of the xnetlib software libraries is based on the highest level of the GAMS [Boisvert et al. 1985] classification system, augmented to include classifications other than mathematical software. Selecting **linear algebra**



```
a              f2c            list       pchip        sodepack
aicm           fftpack        lp         pdes         softlib
alliant        fishpack       machines   performance  sparse
amos           fitpack        magic      picl         sparse-blas
anl-reports    fmm            microscope pltmg        sparspak
apollo         fn             minpack    poly2        specfun
benchmark      fortran        misc       polyhedra    spin
bib            fp             mpi        popi         stoeplitz
bihar          gcv            na-digest  port         stringsearch
blas           gmat           nag        posix        svdpack
bmp            go             napack     pppack       tennessee
bnews          graphics       netlib     presto       toeplitz
c              harwell        news       problem-set  toms
c++            hence          nntp       pvm          trn
chammp         hompack        numeralgo  pvm3         typesetting
cheney-kincaid hpf            ode        quadpack     uncon
clapack        hypercube      odepack    research     vanhuffel
cnews          ieeecss        odrpack    rn           vfftpack
confdb         itpack         opt        scalapack    vfnlib
conformal      jakef          p4         sched        voronoi
contin         kincaid-cheney papers     scilib       xmagic
crpc           lanczos        paragraph  seispack     xnetlib
ddsv           lanz           paranoia   sequent      y12m
dierckx        lapack         parkbench  sfmm
domino         laso           parmacs    slap
eispack        linalg         pascal     slatec
elefunt        linpack        pbwg       sminpack
```

Fig. 2. Library menu.

causes the names of the libraries with linear algebra software to be displayed (Figure 4).

A user may also wish to **search** by key word instead of viewing the contents of a particular library. In this mode, index files containing lists of files and their descriptions are searched by a key word string the user provides. The key word search can be a search on the intersection or union of the words in a search string, a literal search for an exact string (with or without case sensitivity), or a fuzzy search based on the latent semantic indexing technique [Deerwester et al. 1990]. Latent semantic indexing uses statistical analysis to find useful matches that may not be uncovered by other types of searches. In contrast, the fuzzy-search capability in WAIS is based on a heuristic rather than on a statistical approach.

Clicking on the **download** button causes xnetlib to display a list of selected software and documents. The **download path** button allows a user to change the directory to which files will be downloaded. The **dependency checking** button is a modal switch. If **dependency checking** is off, xnetlib will send only the selected routines. If it is on, xnetlib will send the selected routines

DOCKET ALARM

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS

Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS

Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS

Sync your system to PACER to automate legal marketing.

WHAT WILL YOU BUILD? | **sales@docketalarm.com** | **1-866-77-FASTCASE**

fastcase®
Smarter legal research.