Active Netlib: An Active Mathematical Software Collection for Inquiry-based Computational Science and Engineering Education

Shirley Moore, A.J. Baker, Jack Dongarra, Christian Halloy
University of Tennessee

Chung Ng
Morehouse College

Email: active-netlib@cs.utk.edu

Abstract

The efficient application of scientific computing techniques requires specialized knowledge of numerical methods and their implementation in mathematical software libraries that many students, scientists, and engineers, working beyond the already strenuous demands of their particular field, must struggle to achieve. Active Netlib addresses this problem by creating an active collection of executable mathematical software deployed on computational servers and accessible over the network from familiar desktop client interfaces. The Netlib mathematical software collection is being extended in a number of ways to support this project. The NetSolve client-server system provides an active interface to the contents of Netlib by constructing network-accessible objects with executable content from the software packages in Netlib. The NetSolve adapative solver interface guides the user in selecting appropriate software, in setting parameters correctly, and in interpreting numerical results. In addition, Active Netlib provides mechanisms that enable resource users to become resource providers by dynamically uploading and deploying their own software applications which are reviewed before becoming part of the moderated publically available collection. It is hoped that Active Netlib will grow to be a world-wide collection of executable mathematical software, as well as scientific and engineering applications, that is both drawn upon and contributed to by researchers, educators, and students world-wide.

1 Introduction

A core subject in the undergraduate education of application scientists and engineers is the use of mathematical software to solve computational problems. To make effective use of mathematical software, application developers need a basic understanding of the underlying numerical methods and enough knowledge to be able to choose an appropriate solver, parameterize it correctly, and validate the computed results. Correct results are of course required, but good computational performance is desired as well.

Most application scientists have neither the time nor the interest to read the current literature in numerical analysis. They solve numerical problems by relying on the methods and programs they learned about in previous coursework. This tendency has the unfortunate consequence that new methods with improved functionality and/or efficiency may go unused by practicing engineers. Clearly there is a need for lifelong learning about numerical methods and their implementation in mathematical software libraries as well as a need for a mechanism to transfer ongoing research results from the numerical analysis research community to practicing engineers in a form that application scientists can understand. Newer algorithms and approaches such as those described in (Baumann and Oden 1999; Baumann and Oden 1999; Ainsworth and Oden 2000; Kolesnikov and Baker 2000) must be

disseminated from researchers to the broader user community.

Mathematical subroutine libraries can be highly complex. Few application scientists understand how they really work, and the usual practice is to treat the subroutines as "black boxes". A black box is a piece of software that can be used without knowledge of its inner working; the user supplies the input, and the output is more or less assumed to be correct. However, there are a number of pitfalls in numerical computation (e.g., roundoff error, ill-conditioning, non-convergence). Application engineers need enough understanding of the underlying numerical methods to be able to detect and diagnose problems that occur and to modify or customize the methods if necessary. This need is especially crucial in the use of iterative methods to solve large sparse linear systems, where the problem may need to be properly preconditioned in order for convergence to occur and where the appropriate method to employ depends on the nature of the problem being solved. A rich, inquiry-based learning environment can guide application scientists in gradually deepening their understanding of numerical methods to the point where they can confidently and correctly make use of these methods in their work.

A large amount of mathematical software is both commercially and freely available. However, not all the software that is available is of high quality. It can also be difficult to locate the appropriate software by using web search engines, since the descriptions available for searching may be lacking or may not match the vocabulary used by the searcher. A good solution to these problems is to have experts in the field of numerical analysis maintain a moderated collection of high quality software which is organized and catalogued with appropriate metadata to enable easy searching. The Netlib (http://www.netlib.org/) mathematical software repository is such a collection which has been contributed to and managed by the numerical analysis community for the past fifteen years (Browne, Dongarra et al. 1995). Netlib is continually evolving as new state-of-the-art software continues to be contributed by leading researchers in the field of numerical analysis. The mathematical software portion of Netlib is classified using the Guide to Mathematical Software (GAMS) (http://gams.nist.gov/) system, a tree-structured taxonomy of mathematical and statistical problems (Boisvert, Howe et al. 1985). A user may download a copy of the entire GAMS hierarchy and select classifications for searching or may use a Web browser to traverse the hierarchy as a decision tree.

The Active Netlib project (http://icl.cs.utk.edu/active-netlib/) is adding interactivity to Netlib to create an inquiry-based learning environment for computational science and engineering education. The NetSolve (http://www.cs.utk.edu/netsolve/) client-server system for accessing hardware and software resources over a network (Casanova and Dongarra 1997) is being used to provide an active interface to the contents of Netlib. NetSolve essentially constructs network-accessible objects with executable content from the software packages in Netlib. By making the subroutines housed in Netlib available over the network on computational servers, NetSolve enables access to up-to-date mathematical software from a variety of client interfaces running on users' workstations, without requiring the users to download and install the software themselves. Use of NetSolve seamlessly maintains the currency and usability of the content as the underlying hardware, operating systems, and software evolve. Furthermore, the NetSolve Adaptive Solver Interface guides the user in selecting appropriate software, in setting parameters correctly, and in interpreting numerical results (Arnold, Blackford et al. 2000).

The Netlib collection is being further extended through use of the Repository in a Box (RIB) (http://www.nhse.org/RIB/) toolkit, which enables an individual or organization to set up and maintain a repository that interacts with other RIB repositories (Browne, McMahan et al. 1999). RIB allows the Netlib collection to be selectively mirrored and contributed to by all project participants. RIB is based on an IEEE standard data model for software cataloging on the Internet called the Basic Interoperability Data Model (BIDM) (Browne and Moore 1997). The BIDM can be extended to allow cataloging of additional information about software and related resources, such as teaching modules and evaluations. Thus, RIB provides an easy-to-use interface for resource users to become resource providers.

The remainder of this paper is organized as follows:

- Section 2 describes how an active mathematical software collection is being used as a teaching tool.
- Section 3 describes inquiry-based computational science and engineering education.
- Section 4 describes how the Repository in a Box toolkit is being used to enable users to search, interact with, and contribute to the Active Netlib collection.
- Section 5 gives conclusions.

2 An Active Mathematical Software Collection as a Teaching Tool

Netlib has long been the repository of choice for storing mathematical software used for teaching numerical methods courses. For example, the software that accompanies a number of excellent textbooks such as (Forsythe, Malcolm et al. 1977) is stored on Netlib. However, this software has until now been used with a traditional teaching approach that has the following limitations:

- The software is usually downloaded and installed for local use, and learning is limited to those particular subroutines.
- Students use the subroutines as black boxes and get little or no help from the software on making appropriate choices, nor do they get much feedback from the software on whether they are making proper use of it.

Use of effective information technology to deliver active mathematical software, however, can engage students with the material and allow inquiry to be non-linear and guided by the interests and experience of the user, as recommended in (NSF 1998). We are using the NetSolve system to achieve these goals, as described below.

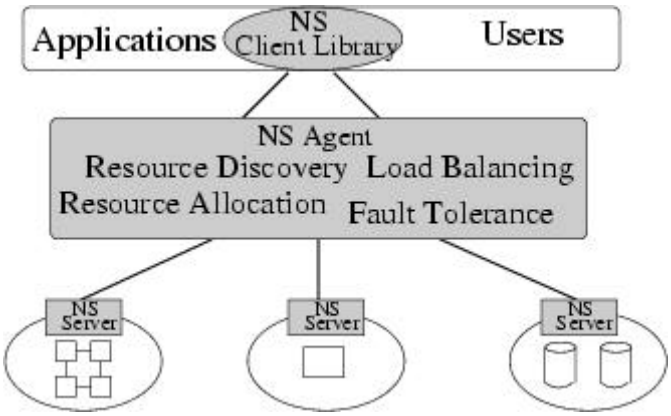

Figure 1. The NetSolve system

The original motivation of the NetSolve project was to alleviate the difficulties that application scientists and engineers encounter when trying to locate, install, and use numerical software, especially on multiple platforms. NetSolve provides remote access to computational resources, both hardware and software. NetSolve is built upon standard Internet protocols and is available for all popular variants of the UNIX operating system, and parts of the system are available for the Microsoft Windows NT/2000 platforms. Figure 1 shows the architecture of the NetSolve system and its relation to the applications that use it. The shaded parts of the figure represent the NetSolve system. At the top tier, the NetSolve client library is linked in with the user's application. The application then makes calls to NetSolve's application programming interface (API) for specific services. The

client can be a programming language, such as Fortran or C, or it can be an interactive interface such as Matlab or Mathematica, both of which are widely used by application engineers in specifying and solving engineering problems. Many engineering applications require the solution of linear systems which may be dense or sparse. NetSolve provides an easy interface to a variety of linear systems solvers available from Netlib. The solvers available via Netlib include those from LAPACK (Anderson, Bai et al. 1999), ARPACK (Lehoucq, Sorensen et al. 1998), PETSc (Balay, Gropp et al. 1997), Aztec (Hutchinson, Shadid et al. 1995), SuperLU (Li 1996), and MA28 (Duff, Erisman et al. 1986). With NetSolve, the user can seamlessly access all these solvers in a uniform way. Example 1 invokes PETSc from the NetSolve Matlab client interface. PETSc (http://www-fp.mcs.anl.gov/petsc/) is a suite of data structures and routines for the scalable parallel solution of scientific applications modeled by partial differential equations (PDEs). Discretization of PDEs results in large sparse linear systems which then need to be solved.

Example 1. [x, its] = netsolve('petsc', A, b, 1.e-6, 500);

Here **A** is the sparse coefficient matrix, **b** is the right hand side, **1.e-6** is the error tolerance, and **500** is the maximum number of iterations. In the output set, **x** is the solution vector, and **its** is the number of iterations that were executed. By simply changing the string **petsc** to one of the other available options, such as **aztec**, the user can change which library will be invoked to solve the problem. Thus, the user can experiment with various libraries without investing the time to learn the data structures and specifics of a particular library.

The NetSolve Adaptive Solver Interface makes the decision of which library and routine to use to solve a given problem and is provided for the user who does not know which library is most appropriate for his application (Arnold, Blackford et al. 2000). From Matlab, the function call in Example 2 will use heuristics discussed below to determine which package to use.

Example 2. [x] = netsolve('LinearSolve', A, b);

When a user requests a linear system to be solved using this black box interface, a number of decisions must be made based on the structure of the matrix **A** to determine which solver to use. Characteristics of **A** to be considered include matrix shape (i.e., whether the matrix is square, rectangular, or triangular) and matrix element density (i.e., how sparse the matrix is, where a sparse matrix is one with many elements equal to zero). For sparse matrices, the main question is whether to use a direct or an iterative solver. Other questions are how to estimate the amount of fill-in (i.e., the number of zero elements that become non-zero during factorization) and how to gauge numerical properties. For a black box iterative solver, NetSolve can automatically supply parameters such as the maximum number of iterations (i.e., how many solution steps will be taken to try to achieve convergence to a solution). If the method reaches the indicated maximum, NetSolve can choose to restart the method or to switch to a direct solver.

NetSolve currently gives only limited feedback concerning the heuristics used and the decisions made. For Active Netlib, the NetSolve Adaptive Solver Interface is being expanded to provide more detailed feedback, thus developing the system into a true teaching tool. As students become more expert, they will wish to invest the time to make the appropriate decisions themselves so that they have more control over the solution process and can achieve better performance.

NetSolve can also serve as a teaching tool for high performance computing. The NetSolve server can run on single workstations, clusters of workstations, symmetric multi-processors, or machines with massively parallel processors. The NetSolve agent (shown in Figure 1) maintains a database of NetSolve servers along with their capabilities (hardware performance and installed software) and dynamic usage statistics. The agent attempts to

find the server that will service the request the quickest. Thus, NetSolve makes the power of supercomputers accessible from low-end machines such as desktop machines and notebook computers. Making cycles on supercomputers and workstation clusters available will enable students to learn about the benefits and tradeoffs in using different high performance computing systems without needing to know anything about computer networking or distributed computing. Later on in their careers when they have their own accounts on such machines, they will be better prepared to make decisions about what resources to use in solving their applications.

Although the main focus of Active Netlib is undergraduate engineering education, the NetSolve system will also provide high school teachers and students access to client interfaces to high performance computing problems in a number of areas. These areas include neurobiology simulations, environmental quality modeling, and image processing. The client interfaces will allow students to experiment with various input sets and observe the results of the mathematical models, as described in (Arnold, Lee et al. 2000). This virtual laboratory will enable high school students to learn about real-world applications of high performance computing research. In addition, graduate students and even researchers will enjoy the versatility and capabilities of the tools offered by Active Netlib.

The project also presents a solution to the problem of how to achieve the benefits of incorporating information technology into education without incurring the accompanying costs. The costs described in (Tissue 1997) include the initial cost of hardware and software and the continual cost of upgrades, maintenance, and technical support. The initial and maintenance costs for the active repository approach are largely concentrated in a network of remote computational servers, with only a small cost to users for desktop machines and software that can run Web browsers and NetSolve clients. The technical work of upgrading and maintaining the software on NetSolve servers is carried out by a small number of trained system administrators, rather than by each individual user. Other costs listed in (Tissue 1997) include increasing percentages of instructor and student time spent learning and remaining adept in the use of information technology. With the NetSolve approach, however, the details of networking and distributed computing are hidden from the user, and users need only become adept at using familiar client interfaces, such as Web browsers and Matlab, that are information technology skills they need anyway to be competitive in the job market in their fields.

3 Inquiry-based Computational Science and Engineering Education

In the past several years, high-performance computing has become an increasingly important tool in many academic disciplines. Computational science is now considered by many to be a third approach to scientific investigation, equal in importance to theory and experimentation. Particularly in the sciences and engineering, where modeling and simulation of physical processes have become routine, high performance computers are essential tools. It is important, therefore, that research scientists, in disciplines in which computation plays a role, be able to use high-performance computers effectively, just as they routinely use the tools of theory and experimentation.

Computer-based simulation is crucial to the design of today's engineered systems. The need for fast transition from concept to product, coupled with the high cost of prototyping, drives this approach. Powerful computational simulation software packages are available for use in select industry segments. Misuse by engineers who do not possess sufficient knowledge to use this software correctly occurs too often, i.e., they lack literacy, which identifies the educational need and opportunity. Advances in engineering education often occur as new needs coincide with newly available technical developments. The need for new educational programs to support engineering instruction in applied computational simulation and the emergence of the Internet-based education environment are the confluence making this innovation opportunity timely.

Our approach to inquiry-based education of computational scientists and engineers initially focuses on the Matlab interface to the NetSolve system. As demonstrations of the efficacy of the approach, we are developing teaching modules in the general area of iterative solution of sparse linear systems as well as in the specific engineering areas of finite element analysis and computational fluid dynamics.

The field of iterative methods for solving systems of linear equations is in constant flux, with new methods and approaches continually being created, modified, tuned, and some eventually discarded. Researchers at the University of Tennessee, together with colleagues, have developed an on-line book that contains templates for this topic, where a template is a description of a general algorithm rather than the executable object code or source code commonly found in a conventional software library (Barrett, Berry et al. 1994). In contrast to a black box piece of software, templates can be adapted for a specific application and/or computing environment. This approach is especially relevant to the use of iterative methods for solving large sparse systems of linear equations, where the trick is to find the most effective method for the problem at hand. Unfortunately, a method that works well for one problem type may not work as well for another or may not work at all.

Templates exploit the expertise of two distinct groups. The expert numerical analyst creates a template reflecting in-depth knowledge of a specific numerical technique. The computational scientist then provides “value-added” capability to the general template description, customizing it for specific contexts or application needs. Over a dozen iterative methods that either illustrate the historical development of iterative methods or represent the current state-of-the-art for solving large sparse linear systems are covered in (Barrett, Berry et al. 1994). Each template provides a general description, including a discussion of the history of the method and numerous references to the literature, as well as the mathematical conditions for selecting a given method and a discussion of convergence and stopping criteria. Each template also includes an example Matlab implementation. For the Active Netlib project, we are integrating the template approach with the NetSolve Adaptive Solver Interface to allow users to explore and experiment with the various templates using NetSolve.

The CFD Lab at the University of Tennessee has developed what is proving to be a highly effective Internet format for integrating all academic components of finite element analysis (FEA) coursework at the Master of Science level. Now in its third year, this course’s academic content now exceeds that which could be included in the classical lecture room blackboard venue. The enabling development was construction of a fully functional website to support the Internet format, which has now been extended to the graduate course in computational fluid dynamics/numerical heat transfer (CFD/NHT). The website (http://cfdlab.engr.utk.edu/) responds to the intent, spirit, context and requirements of a genuine, collaborative education environment on the Internet via:

- specifically composed and written graduate-level academic courseware (electronic form) replacing the "classic" hardbound text;
- collaborative computational laboratory experiences with electronic reporting online, admitting real-time review, correction and grading;
- a primer specifically written to provide just-in-time directions on use of Matlab for designing/executing computational experiments;
- archived lectures available for random access review at any time via video streaming (no transmission of large files);
- complete hardware/communication directions, live-feed reception spanning 28.8K modem to T1 connections, identification of minimal (cheap!) and optimal PC equipment for the receiving end;
- totally interconnected (hot-worded) website for class schedule, courseware, problem assignments, lab projects, grades, questions; and
- continuously available two-way chat-room communication during lecture, 24 hour open group email

enabling posting/answering questions by any participant as well as graduate teaching assistants and faculty.

3.1 Finite Element Analysis (FEA) for Undergraduate Engineers

The key word above is Matlab, which is a highly effective matrix manipulation system resident within NetSolve. In development of the FEA course computational experiment capability, a comprehensive Matlab "toolbox" has been developed. Via elementary appearing templates created in a word processing environment, the toolbox enables communication of the FEA theory to computational syntax without getting into the computer science details. This construction is ideal in meeting the undergraduate education requirement, as coursework focus is on engineering and design, not applied computer science. Integrating this toolbox with NetSolve means that an academic searching for the capability need only address an inquiry to Active Netlib. The opportunity for collaboration on development of the academic course content is enabled by the Internet venue for the toolbox, as well as existence of the developed website as a template for development of the undergraduate course in FEA.

Upon insertion, the continuation task element is to disseminate knowledge of the toolbox existence, such that academics teaching FEA at the undergraduate level may assimilate and enhance its capability, hence applicability across all engineering disciplines. As an extension, Matlab currently possesses the GMRES sparse solver with appropriate preconditioner. NetSolve provides access to recent innovations in sparse solvers, hence one could improve on the inherent performance capitalizing on the vision of integration of research and education.

3.2 Computational Fluid Dynamics (CFD)/Numerical Heat Transfer (NHT) for Undergraduate Engineers

The current Master of Science level Internet course uses a CFD Lab-generated "problem solving environment (PSE)" to support computer experiments on the associated very non-linear Navier-Stokes (NS) Partial Differential Equations (PDE) system. The goal is to extend the current Matlab toolbox capability to address a modest first level of the NS conservation law system. The scope of this activity will be expanded to the more comprehensive development aimed at support of the desired undergraduate engineering academic course in CFD. At present, in such a course, topical coverage is limited to potential flow and boundary layer analysis, both of which constitute substantial simplifications of the parent NS system.

The resultant toolbox, when placed under NetSolve, will substantially enhance the scope of the Matlab environment in support of undergraduate academic content opportunity in applied computational simulation. Here again, as above, the venue will exist for collaborative enhancement of the generated capability, as engineering academics become aware of the potential for such a system on Internet.

4 Searching, Interacting with, and Contributing to the Collection

Together with the Repository in Box toolkit discussed below, NetSolve provides a mechanism that resource users can use to become resource providers. More specifically, a key component of the NetSolve server is a source code generator that parses a NetSolve problem description file (PDF). A PDF contains information that allows the NetSolve system to create new modules and incorporate new functionalities. A Java GUI is provided that aids in the creation of PDFs. Thus, users can contribute their own subroutines and applications to the NetSolve system. For example, an engineer may find it necessary to modify a subroutine or use a variation of a method to solve a related problem and may wish to make his work available to others. Or an application scientist may compose an application from existing modules, supplemented by his own code, and wish to make the final application available from NetSolve via a Web or other client interface so that interested teachers and students can learn about his application.

4.1 Repository in a Box Toolkit

The Repository in a Box (RIB) toolkit was designed from the ground up with the goals of interoperability and extensibility (Browne, McMahan et al. 1999). RIB provides easy-to-use mechanisms for creating and maintaining software catalog records, exchanging these records with other repositories, and automatically generating a browseable and searchable user interface to the software repository. In addition, the Repository in a Box Application Programmer Interface (RIBAPI) provides a backend interface for interoperation with other web-based digital library and grid computing toolkits such as Globus (Foster and Kesselman 1997; Dongarra, Millar et al. 2000). RIB has been used by the National High Performance Software Exchange (NHSE) project to provide a uniform interface to a distributed set of discipline-oriented, high performance computing (HPC) software repositories (Browne, Dongarra et al. 1995; Browne, Dongarra et al. 1998). The NHSE repositories include not only mathematical software and parallel programming tools, but also a number of autonomously operated repositories in areas such as earth and space sciences, computational chemistry, and signal and image processing. RIB interoperation capabilities not only allow these repositories to interoperate with each other, but also allow construction of a top-level NHSE virtual repository that indexes all the available HPC software.

Software cataloging in RIB is based on the Basic Interoperability Data Model (BIDM). The BIDM, an IEEE standard (1420.1-1995) for software reuse libraries, specifies a minimal set of metadata that a reuse library should provide in order to interoperate with other reuse libraries (IEEE 1995). The BIDM is expressed in terms of an extensible entity-relationship data model that defines classes for assets (the reusable entities), the individual elements making up assets (i.e., files), libraries that provide assets, and organizations that develop and manage libraries and assets. A pictorial view of the BIDM is shown in Figure 2. Although the Basic Interoperability Data Model has greatly enhanced the ability of reuse libraries to interoperate, it is desirable to be able to extend the basic model to cover specific areas more thoroughly or to meet the needs of specialized libraries. Two areas for which IEEE standard extensions have been developed are asset evaluation and certification and intellectual property rights. The Asset Certification Framework (1420.1a-1996) defines a standard for the consistent structure, labeling, and description of evaluation and certification policies and results. The Intellectual Property Rights Framework (1420.1b-1999) provides a consistent framework for labeling and describing intellectual property rights and other legal restrictions on software assets. Netlib and NHSE developers have not only participated in the software reuse community effort to define the above standards, but have also developed their own extensions to the BIDM in such areas as software deployment and performance. The software deployment extension adds classes to the BIDM for describing machines and software installations on those machines. RIB uses the software deployment metadata to generate a matrix that displays for the user what software is available on what machines and provides pointers to usage information.

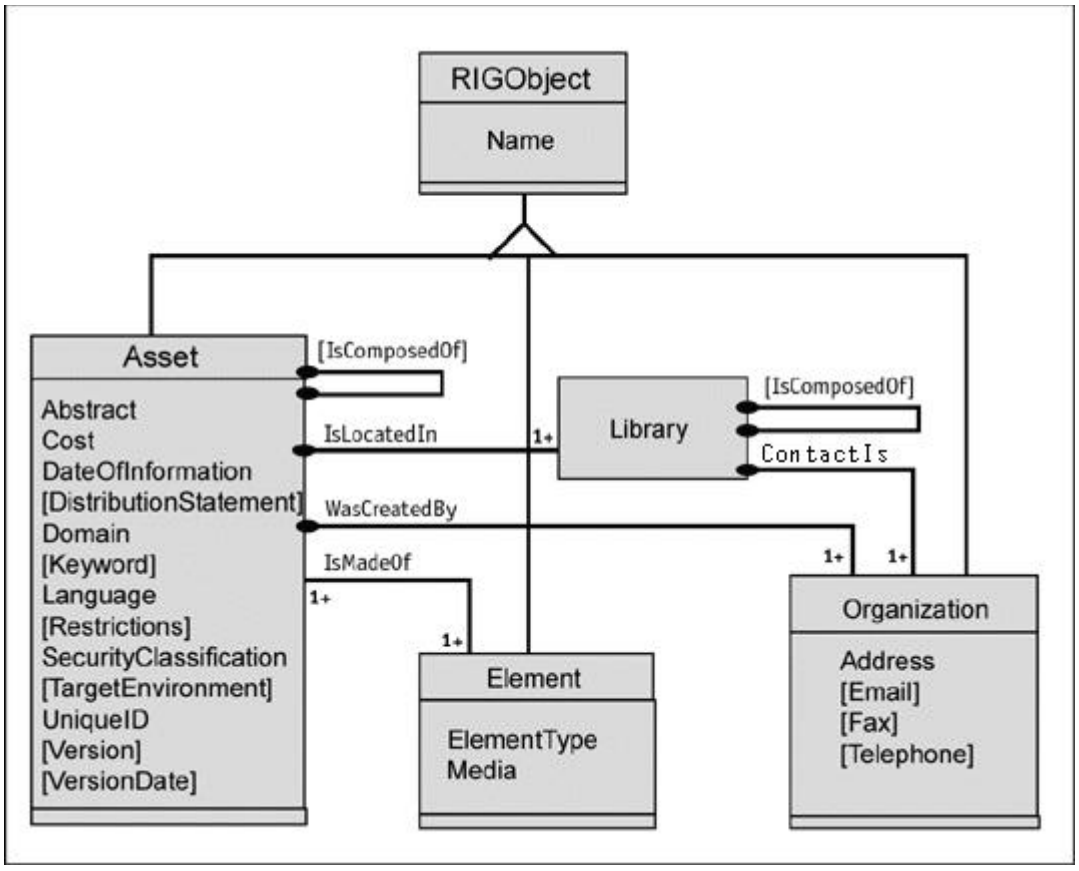

Figure 2. Graphical view of the Basic Interoperability Data Model

RIB is being extended in a number of ways to support the Active Netlib project. The software deployment extension to the BIDM is being used to catalog the executable mathematical and application software content of the collection – i.e., the installation of that software on NetSolve servers – as well as the active interfaces available for invoking the software. RIB is also being used to catalog accompanying instructional and other related materials. A user can search the repository, for example by traversing the GAMS hierarchy to the level that his or her expertise allows. Upon locating software (or a class of software) relevant to solving his or her problem, the user would see a list of available interfaces and be able to click on a particular interface to invoke it. For example, clicking on the Matlab interface for "Solution of Linear Systems" would start up Matlab on the user's machine if it wasn't already running (assuming Matlab was installed on that machine), download and install the NetSolve Matlab client if necessary, and guide the user through invoking the NetSolve LinearSolve interface from Matlab. Clicking on a link to a Web interface would take the user to a web page from which he could enter or upload input data, send the data to a NetSolve server that would carry out the request, and view the results of the computation. An example of such an interface is the web-enabled interface to IPARS (Arnold, Lee et al. 2000). IPARS (Integrated Parallel Accurate Reservoir Simulator), developed at the Center for Subsurface Modeling at the University of Texas Institute for Computational and Applied Mathematics (TICAM), is a framework for developing parallel models of subsurface flow and transport through porous media. It currently can simulate single phase (water only), two phase (water and oil) or three phase (water, oil and gas) flow through a multi-block 3D porous medium. IPARS can be applied to model water table decline due to overproduction near urban areas, or to model enhanced oil and gas recovery in industrial applications. NetSolve provides a gateway to the IPARS system without downloading and installing IPARS code. With just access to a web browser, the user can enter input parameters and submit a request for execution of the IPARS simulator to a NetSolve system. The output images are then brought back and displayed via the web browser.

4.2 Resource Users Becoming Resource Providers

RIB and NetSolve are being used to allow resource users to become resource providers, while safeguarding the high quality of the collection. Although the Netlib mathematical software repository has contributors from all over the world, it is a moderated collection controlled by an editorial board, and all software is reviewed by an area editor who may request changes (e.g., better documentation, conformance to software standards) before admitting the software to the collection. RIB allows objects to be designated as either approved or unapproved, and a RIB repository can be configured so that unapproved objects do not appear in the catalog that is displayed to users. A web form allows users to contribute software, interfaces, and related materials to the collection, but the contributed resources are reviewed and approved before appearing in the catalog. Reviewers of software and related material are given a special password that allows them to try out contributed resources for the purpose of review. The NetSolve project is developing a dynamic loading capability that will allow software catalogued in RIB to be automatically downloaded and installed on NetSolve servers. The dynamic loading capability is being designed with appropriate security precautions in mind. Thus, users who contribute software routines or applications will be able to enter metadata about the software into RIB, upload the software files, and then have the software automatically deployed on a NetSolve server and the deployment information automatically entered into RIB.

We expect that many contributions from users will take existing software and provide interfaces to that software such as web forms or Matlab scripts, or provide post-processing and visualization capabilities. Users will be able to use RIB to catalog such resources and to specify relationships between their contributions and existing resources so that other users will be able to navigate among them. For example, a user who traverses the GAMS hierarchy to the level of “Solution of Linear Systems” might find not only the NetSolve “Linear Solve” interface, but also links to educational materials developed using that interface, or perhaps to Matlab specifications of application problems solved using that interface. Some Netlib software is also maintained on SourceForge (http://sourceforge.net/), and Active Netlib complements SourceForge by providing the additional capability of executable content on computational servers.

The interoperation capability of RIB allows a distributed set of repositories to be autonomously operated by all participants in the project, as well as by other individuals and organizations who may wish to participate. The NetSolve software is also freely available, and assistance is provided to interested parties who wish to set up their own NetSolve servers. Such servers will be incorporated into the NetSolve distributed system, with adequate review of the services provided to ensure high quality. In particular, RIB and NetSolve installations for Active Netlib are hosted at the University of Tennessee and at Morehouse College. The Morehouse installation supports computational science and engineering education at Morehouse, Clark Atlanta University, Spelman College, and Morris Brown College. The sites making up the distributed Netlib software repository all mirror each others’ contents, with a specific site being the master site for a particular subcollection of software. Similarly, Morehouse mirrors selected contents from Netlib on their RIB server and install these contents on their NetSolve server. Project personnel at Morehouse will be responsible for managing contributions to their collection, and these contributions can then be mirrored by the University of Tennessee server. RIB interoperation also allows participants in the project access to a large body of software available from other RIB servers across the world. For example, the HPC software repository at the National Center for Supercomputer Applications contains metadata about HPC software available in a number of application areas, including chemical engineering, computational fluid dynamics, and astronomy and cosmology.

5 Conclusions

The Active Netlib project is intended to advance knowledge and understanding of state-of-the-art numerical methods and high performance computing across scientific application areas and engineering fields. The Active Netlib project explores new concepts of executable content and dynamic creation of new content. The activities will advance discovery and understanding of new numerical algorithms and new uses of high performance computing by engaging students, educators, and scientists and engineers at all levels in on-line investigations. The infrastructure for research and education in the area of computational science and engineering will be enhanced by a growing network of software repositories and computational services. Researchers, educators, and students will all be able to both learn from and contribute to the growing collection of executable mathematical software content as well as applications of and interfaces to that software.

References

IEEE(1995). IEEE Standard for Information Technology 1420.1 - Software Reuse - Data Model for Reuse Library Interoperability: Basic Interoperability Data Model (BIDM), IEEE.

NSF (1998). Information Technology: Its Impact on Undergraduate Education in Science, Mathematics, Engineering, and Technology, National Science Foundation.

Ainsworth, M. and T. J. Oden (2000). A Posteriori Error Analysis in Finite Element Analysis. New York, John Wiley and Sons.

Anderson, E., Z. Bai, et al. (1999). LAPACK User's Guide. Philadelphia, PA, Society for Industrial and Applied Mathematics.

Arnold, D. C., D. Bachmann, et al. (2000). Request Sequencing: Optimizing Communication for the Grid. Euro-Par 2000 -- Parallel Processing, Springer-Verlag.

Arnold, D. C., S. Blackford, et al. (2000). Seamless Access to Adaptive Solver Algorithms. 16th IMACS World Conference on Scientific Computation, Applied Mathematics and Simulation, Laussanne, Switzerland.

Arnold, D. C. and J. J. Dongarra (2000). The NetSolve Environment: Progressing Towards the Seamless Grid. 2000 International Conference on Parallel Processing (ICPP-2000), Toronto, Canada.

Arnold, D. C., W. Lee, et al. (2000). Providing Infrastructure and Interface to High-Performance Applications in a Distributed Setting. High Performance Computing Conference.

Balay, S., W. Gropp, et al. (1997). Modern Software Tools in Scientific Computing, Birkhauser Press.

Barrett, R., M. Berry, et al. (1994). Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods.

Baumann, C. and T. J. Oden (1999). “A Discontinous hp Finite Element Method for the Solution of the Euler and Navier-Stokes Equations.” International Journal for Numerical Methods in Fluids 31: 71-95.

Baumann, C. and T. J. Oden (1999). “A Discontinuous hp Finite Element Method for Convection-Diffusion Problems.” Computer Methods in Applied Mechanics and Engineering 175(Special issue on Spectral Element and HP Methods in CFD): 311-341.

Boisvert, R., S. E. Howe, et al. (1985). "GAMS: A Framework for the Management of Scientific Software." ACM Transactions on Mathematical Software 11(4): 313-355.

Browne, S., J. J. Dongarra, et al. (1995). "The National HPCC Software Exchange." IEEE Computational Science and Engineering 2(2): 62-69.

Browne, S., J. J. Dongarra, et al. (1995). "The Netlib Mathematical Software Repository." D-Lib Magazine 1(9).

Browne, S., J. J. Dongarra, et al. (1998). "The National HPCC Software Exchange (NHSE)." D-Lib Magazine 4(5).

Browne, S., P. McMahan, et al. (1999). The Repository in a Box Toolkit for Software and Resource Sharing, University of Tennessee Computer Science Department.

Browne, S. and J. W. Moore (1997). Reuse Library Interoperability and the World Wide Web. ACM SIGSOFT Symposium on Software Reusability, Boston, MA, ACM Press.

Casanova, H. and J. J. Dongarra (1997). "NetSolve: A Network Server for Solving Computational Science Problems." International Journal of High Performance Computing Applications 11(3): 212-223.

Dongarra, J. J. and E. Grosse (1987). "Distribution of Mathematical Software via Electronic Mail." Communications of the ACM 30(5): 403-407.

Dongarra, J. J., J. Millar, et al. (2000). RIBAPI -- Repository in a Box Application Programmer's Interface, University of Tennessee Computer Science Department.

Duff, I. S., A. M. Erisman, et al. (1986). Direct Methods for Sparse Matrices. Oxford, Clarendon Press.

Forsythe, G. E., M. A. Malcolm, et al. (1977). Computer Methods for Mathematical Computations, Prentice-Hall.

Foster, I. and C. Kesselman (1997). "Globus: A metacomputing infrastructure toolkit." International Journal of Supercomputer Applications 11(2).

Hutchinson, S. A., J. N. Shadid, et al. (1995). Aztec User's Guide: Version 1.1, Sandia National Laboratories.

Kolesnikov, A. and A. J. Baker (2000). "Efficient Implementation of High Order Methods for the Advection-Diffusion Equation." Computer Methods in Applied Mechanics and Engineering 189: 701-722.

Lehoucq, R., D. Sorensen, et al. (1998). ARPACK Users' Guide. Philadelphia, PA, Society for Industrial and Applied Mathematics.

Li, X. (1996). Sparse Gaussian Elimination on High Performance Computers. Computer Science, University of California at Berkeley.

Plank, J. S., H. Casanova, et al. (1999). "Deploying fault tolerance and task migration with NetSolve." Future

Generation Computer Systems 15(5-6): 745-755.

Tissue, B. M. (1997). The Costs of Incorporating Information Technology in Education. Summer On-Line Conference on Chemical Education.

Zia, L. L. (2001). "Growing a National Learning Environments and Resources Network for Science, Mathematics, Engineering, and Technology Education." D-Lib Magazine 7(3).