Supercomputing at the Desktop: An Improved Interface Using Internet Facilities

Bernhard Bühlmann, Hanspeter Bieri

Institute of Computer Science and Applied Mathematics
University of Berne, Switzerland

Abstract. The traditional complex access procedure still prevents a lot of scientists to migrate their applications to massively parallel computers and to benefit from their computing power. We present a new approach for obtaining a simpler user interface using standard components of today's Internet technologies. By means of a parallel raytracing application, called DeepRay, as an example we show how to implement a simple yet powerful interface to supercomputers.

1 Introduction

Today's supercomputers still lack simple user interfaces and access procedures. Massively parallel computers offer a processing speed higher by an order of magnitude compared to common workstations, but the complex access methods prevent a lot of scientists to migrate their applications and to make use of this computing power. The investment for a researcher or scientist to use the supercomputer's facilities in the traditional way is notoriously big. In order to simplify the user interface of complex applications, several approaches have been made to adapt it to today's network environments [2] [7] [10].

We present a new approach towards a simpler user interface which is more general and uses standard components of today's Internet technologies. By means of a parallel raytracing application, called DeepRay, as an example we show how to implement a simple yet powerful interface to supercomputers. DeepRay should motivate users to apply supercomputers instead of their much slower workstations and personal computers. Using DeepRay, users connected to the Internet can easily render remote geometric scene definitions on a supercomputer. Our approach can be applied to a wide range of supercomputing applications.

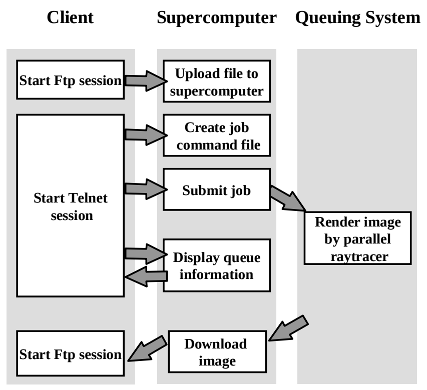
The present paper is essentially divided into four parts: In Section 2 the traditional access procedure for supercomputers is discussed. As an illustration, the execution of a parallel raytracing application on an IBM SP supercomputer is explained. In Section 3 the concept of DeepRay, a simple and easy to use method for working on supercomputers, is introduced. Section 4 explains how we implemented the DeepRay services using standard components of the Internet. Finally in Section 5 this access to supercomputing is discussed, and enhancements and possible future extensions are indicated.

2 The traditional access to supercomputers

In the early days of computing, simple terminals were connected to a central host computer. Users logged in from a terminal and worked on the host with their programs and data files on the host's disk. Later, when multiple computers could be connected by a network, it was possible for the users to interact with a remote host from their workstations. Since the advent of the Internet, remoteness has no longer been a problem. Today, computers are connected worldwide and can be accessed from any location by means of an Internet access point.

Working traditionally with supercomputers

A supercomputer can serve many kinds of researchers who wish to submit computation-intensive jobs to a number cruncher. A good survey of supercomputing is [12], which provides a lot of information about this growing market. Today, the scientist works at a personal workstation using a UNIX- or Windows-based operating system. A supercomputer is located at a computing center, and the users can access it by a network, for example by the IP Internet protocol. Each user has an account on the supercomputer with a personal password and a limited amount of disk space.



1. Prepare the input data
2. Upload the input data to the supercomputer
3. Login to the supercomputer
4. Start the job on the supercomputer
5. Wait for the results and check the progress of the computation
6. Download the results to the personal workstation
7. Analyze the results. If necessary, go back to step 1

Fig. 1. The main steps when accessing supercomputers in a traditional way

Figure 1 shows a typical traditional access procedure, where the user starts a Telnet session at his workstation. He transfers his files and data to the supercomputer using a file transfer protocol. For every access, he has to provide a login name and a password. In the corresponding working cycle, all steps must be done manually by the user. This time-consuming procedure has prevented a lot of researchers to use a supercomputer. The user prefers to remain in the well-known environment of his personal workstation and operating system, even if the processing power is lower by one or two orders of magnitude, depending on the scalability of the application. In the following, we illustrate this traditional approach by means of an example, where a geometric scene definition is rendered by a parallel raytracer.

The BOOGA parallel raytracer

BOOGA[1] [1] is an object-oriented graphics framework developed at the University of Berne. The BOOGA parallel raytracer has been implemented using MPI, the message passing interface standard. To render an image using the parallel raytracer according to the steps shown in Figure 1, the user transfers the scene definition file (1) to the supercomputer (2) using a file transfer protocol or a directory service like NFS[2]. A Telnet session is started on the supercomputer (3) and the job for the calculation of the final image is submitted to the batch queuing system of the supercomputer (4). After job completion (5), the resulting image is transferred back to the user's workstation (6).

At the University of Berne, the BOOGA parallel raytracer is running on an IBM RS/6000 SP supercomputer. On the SP, LoadLeveler [5] is used as a batch queuing system for load balancing.

Weaknesses of the traditional approach

With such a classical user interface, the facilities of the BOOGA parallel raytracer have been accessible only by a small number of users who have an account on the supercomputer and enough skills to render an image using the LoadLeveler queuing system.

Accessing supercomputers in the way shown in Figure 1 has many disadvantages:

- Data must be kept on both workstation and supercomputer, which leads to synchronizing problems.
- The access using Telnet is line-oriented, i.e. lacking most advantages of a graphical user interface (GUI).
- The training of new users is very time-consuming because of the complex network environment.
- In most cases, a different operating system must be learned to utilize the supercomputer.
- A new account with a different login name and password must be memorized by the user.

Many experiences have shown that these disadvantages prevent a lot of potential users to benefit from the resources of supercomputers.

[1] **B**erne's **O**bject-**O**riented **G**raphics **A**rchitecture
[2] Network File System

3 An alternative approach using the Web

The Internet offers many new facilities allowing different platforms and operating systems to share the same data and to communicate using the standardized TCP/IP protocol. The Hypertext Markup Language HTML can be used to publish multimedia documents on a Web server which provides these documents for all users of the Internet. The HTML documents can be viewed by means of a Web browser. The Common Gateway Interface CGI [8] is a standard for interfacing an external application, such as a database server or an order-entry system, by means of a WWW-server. Java [3] is a platform-independent programming language with a strong network focus. Javascript [6] can add further multimedia elements and interaction to HTML documents. One of the key issues of the Internet is that data, the user interface, and the documentation can now be packed into one single document. All these new facilities encourage to think of a new generation of user interfaces.

Requirements on the design of user-friendly interfaces to supercomputers

How can the access to supercomputers be improved by means of Web facilities? In the following, we list a number of features leading to a more comfortable user interface to the facilities of a supercomputer than the traditional one.

1. **Intuitive use:** The user interface should be self-explaining. No efforts should be necessary for the user to learn the user interface. The user should be allowed to execute his application without realizing that he is accessing a supercomputer.
2. **Transfer of local data to the application:** Local data should be transferred from the user's workstation to the supercomputer without the necessity to provide a login name and a password.
3. **Platform independence:** The service should be offered by every hardware platform. With today's fast changing hardware and software standards, a platform-independent solution is required to guarantee future availability of the service.
4. **Location independence:** As it is possible to access network resources via the Internet from almost everywhere in the world, the service should be accessible from practically every location.
5. **Reporting of progress:** The user should have the opportunity to query the current status of his job in the processing queue.
6. **Notification after completion:** After job completion, the user should automatically get a notification message of the processed job by Email.
7. **Simple download of the results:** The results of the calculation should be downloaded without the necessity to specify a login name and a password. The results should be transferred to the user's workstation as a Web document.

It seems possible that by means of the Internet technologies, all these requirements can be met. Some steps in this direction have already been made by Krishnan et al. in the MMM project [7]. Rice et al. [10] proposed an approach for using the Web as a graphical user interface for applications not necessary aimed at supercomputers. Barbera et al. presented DRS [2], a distributed raytracing system applying TCP/IP sockets communication. The DRS system is based on a classical client-server model using the TCP/IP protocol. The user interface was implemented by means of the Open Interface Graphic library. DRS fulfills most of our requirements, but location and platform independence have not been achieved. In order to fulfill all requirements listed above, we propose a new concept for better user interfaces.

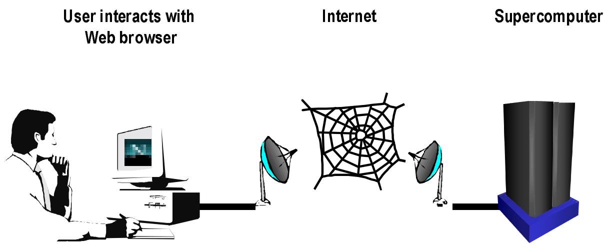


Fig. 2. The Web-based supercomputing working environment

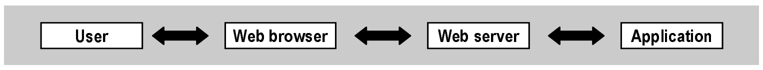


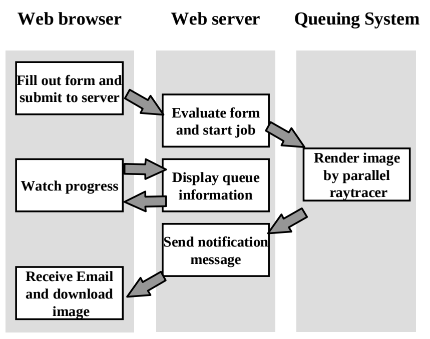
Fig. 3. The concept of DeepRay

The concept of DeepRay

Figure 2 shows a user interacting with a supercomputer and benefiting from the Internet as intermediate. More abstractly, Figure 3 shows the concept of DeepRay as an Internet service[3]. The user interacts with a HTML document which is displayed by a standard Web browser like Netscape Navigator or Microsoft Internet Explorer. The user's workstation, personal computer or handheld PC is connected to the Internet. On the supercomputer, a Web server acts as an interface between the Internet and the computing resources. The interface to our parallel raytracing application can be designed to consist of one single HTML page, where all the parameters of the application and the queuing system can

[3] The homepage of DeepRay is http://deep-ray.unibe.ch

be specified. In particular, the scene description file can be uploaded to DeepRay by means of a specification in the HTML form. This important possibility will be explained later in more detail.


Web browser
Web server
Queuing System
Fill out form and submit to server
Evaluate form and start job
Render image by parallel raytracer
Watch progress
Display queue information
Send notification message
Receive Email and download image

1. Prepare the input data
2. Fill out the form and submit it
3. Download the results to the personal workstation
4. Analyze the results. If necessary, go back to step 1

Fig. 4. The main steps when accessing supercomputers over the Web

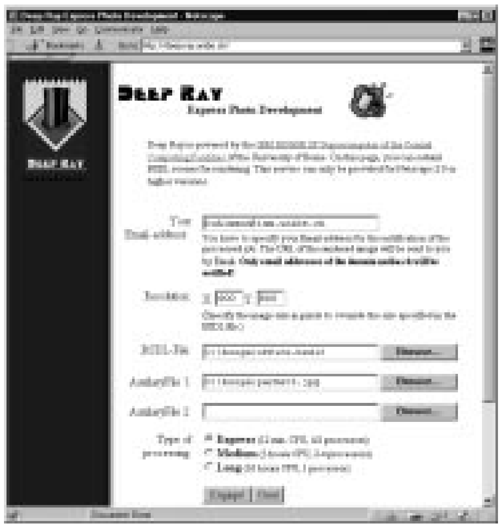
DeepRay's simple access procedure

In the following, we describe the data flow of the DeepRay parallel raytracing engine, as it is shown in Figure 4. The interface of the user to the DeepRay service consists of one single Web page (Figure 5) where all the parameters can be set. The user specifies his Email address for the notification of the job completion and for the delivery of the resulting image. The complete filename of the scene definition to be processed must be given. Optionally, the default values for the image resolution and the estimated computing time can be changed. Then the user submits the job to DeepRay. The scene description and the parameters are transferred to the supercomputer, where the job is automatically submitted to the queuing system. After job completion, a notification message containing the status of the job and a hyperlink to the calculated image is sent back to the user by Email. By activating the link, the user causes the browser to download the image, which is stored in the JPEG format, to display it. Then the user can save the image from the browser window to the local disk.

Comparing Figure 1 and Figure 4, we can see that the steps 2, 3, and 4 of the traditional access procedure have been replaced by the use of one HTML form. Step 6, i.e. the download of the result, can be reduced to one mouseclick, without having to provide a login name and a password.

Visual presentation of the user interface

DeepRay uses only a small number of Web pages, among which the homepage shown in Figure 5 is the most important one. Siegel [11] proposed some very important guidelines for modern and well designed websites, which have been


Deep Ray
Express Photo Development

[illegible]

Your Email-address [illegible]

Resolution X [illegible] Y [illegible]

BOOGA-File [illegible] Browse...

AuxiliaryFile 1 [illegible] Browse...

AuxiliaryFile 2 Browse...

Type of processing
- Express [illegible]
- Medium [illegible]
- Long [illegible]

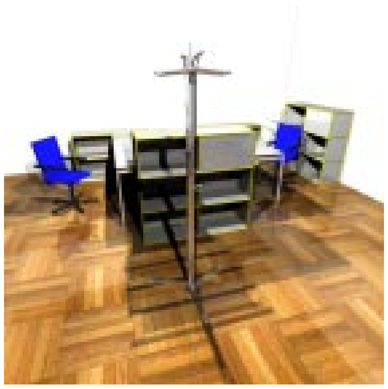


Fig. 5. The DeepRay homepage and a raytraced image of an office scene

adapted. The DeepRay Web pages have been designed to possess a consistent look and feel and easy navigation facilities.

A comparison of execution times

We compared the rendering of the scene in Figure 5 on an Intel Pentium 200 PRO personal computer and on an IBM SP with 8 nodes, i.e using DeepRay. The rendering on the PC in a DOS box by means of the BOOGA raytracer took 2 hours and 15 minutes. Processing the scene using the DeepRay service took 12 minutes, including the time for the delivery of the notification Email and the download of the image by a 33.6K modem. That is, DeepRay was faster by a factor of 11.25. Of course, DeepRay's execution time depends on the number of jobs pending in the LoadLeveler queue and on how fast the notification mail is sent to the mail client of the user. Other examples showed also that the rendering speed on the PC is slower by about a factor of 12. That is, rendering using DeepRay instead of a PC is as comfortable and much faster.

The proposed solution fulfills all the requirements on a modern, user-friendly interface. Using Web technologies can simplify the access to supercomputing resources and, therefore, enable more users to benefit from much computing power and high-performance applications.

4 A closer look at DeepRay

We implemented the concept of DeepRay using standard software components of today's Web technologies and the IBM SP supercomputer. A Web server is running on the SP where HTML documents define the complete GUI for DeepRay. Transparent to the user, LoadLeveler is used as queuing system and Perl [14] as programming language for implementing the control and interfacing of all components involved. In the following, we discuss the most important aspects of our implementation in more detail:

Validation of the Email address

Before submitting the job to the SP, the customer's Email address must be validated. So far, only users owning an Email address of the University of Berne have been allowed to use the DeepRay facilities. This restriction will soon be relaxed by means of a registration service. The Email address is validated by a Javascript [6] and must only be specified when visiting the DeepRay pages for the first time. With every subsequent visit, the Email address can be restored by means of a persistent cookie which is stored in the user's web browser.

Specification of the parameters

Besides the scene description, other parameters can be specified on the submitting Web page. On the DeepRay page, the user can specify the estimated execution time for his job. He can also specify the resulting image's resolution with the aid of the GUI elements of the FORM tag provided by HTML.

Upload of the data to the Web server

In the World Wide Web most of the network traffic flows from the Web server to the browser. It is not very common in the Internet to upload files to a server. There exist security restrictions which are very severe in case of accessing private data on the user's machine. The HTTP protocol provides two mechanisms for user feedback:

1. A hyperlink can be selected which requests a new document from the Internet.
2. A HTML form, where a more detailed user interaction can be defined using selections, radio buttons and menus.

Java and JavaScript allow even more user interaction, however such an upload of a file to a server violates fundamental security restrictions. Nebel and Masinter [9] proposed RFC 1867 which defines a file upload protocol standard for HTML forms. So far, Netscape Navigator 2 and Microsoft Internet Explorer 3.02, or higher versions, include RFC 1867 and support file upload facilities.

The example in Figure 6 shows the HTML definition and the resulting form for uploading files to a Web server. On browsers which do not support file upload, the browse button will not appear. When the user clicks the browse button, a file requester appears which can look in different ways depending on the platform on which the browser is running. The user can then select the file which he wishes to

```
<HTML><BODY>
<FORM ENCTYPE="multipart/form-data"
  ACTION="upload.pl">
File to process:
<INPUT NAME="userfile" TYPE="file">
<INPUT TYPE="submit" VALUE="Send file">
</FORM>
</BODY></HTML>
```

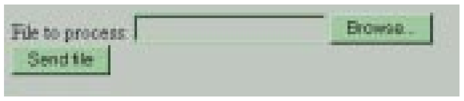


Fig. 6. HTML definiton of an upload form, and the form's representation in the browser

```
#!/usr/local/bin/perl5
require 'CGI_LIB.pl';

open(TMP, "> userfile.txt");
binmode(TMP);
print TMP $CGI{'userfile'}->{'Contents'};
close(TMP);
```

Fig. 7. Perl script for storing a file on the server

upload. After closing the file request dialog, the complete local path is displayed in the input text box.

Up to three files can be uploaded at once to DeepRay. Geometric scene definitions can consist of multiple files which contain the geometric descriptions, bitmap textures or object libraries. Figure 7 shows an excerpt of the Perl script which receives the file from the user's workstation and stores it on the local disk of the Web server. The Perl library CGI-LIB [13] is used to parse the input stream. A more detailed description of CGI programming can be found in [4]. The number of files which can be uploaded to a Web server by a single transaction is not limited.

Submitting the job on the SP

The job to be submitted to the LoadLeveler queuing system is split into three steps. In the first step, the image is rendered by the BOOGA parallel raytracer. In the second step, the resulting image is converted to the JPEG file format and copied to the image library of DeepRay. A notification message is composed and sent to the customer. The last job step removes the input files and cleans up all temporary data.

Notification of the user

After job completion, the user will be notified by an Email message. This message consists of the reporting and the error messages of the parallel raytracer, and of a hyperlink to the resulting image. With a single mouseclick, the user can download the image to his personal workstation without having to provide a login name and a password. After the notification of the user, the input files on the supercomputer are deleted to save space. All the generated images are kept on a filesystem of the supercomputer. Since the JPEG image-compression algorithm is very efficient, the average file sizes of the resulting images are very small. If the filesystem fills up, the oldest images can be deleted to create more space.

Using all these standard components, a comfortable solution for easy access of a supercomputer can be provided. Perl is the glue between the components and provides a good runtime performance, although it is an interpreted language.

5 Final remarks

Generalizing the DeepRay concept

The DeepRay concept allows to be adapted to most of today's supercomputing applications, where the input data can be provided by one or more files and the output of the calculations can be visualized and processed at the user's workstation. Especially, applications with many users are candidates for such a Web interface.

Extensions

Because of the use of standard components and an open architecture, the DeepRay services can be expanded in many ways:

- **Other raytracing formats:** The DeepRay concept could be extended to support other raytracers, like POVray and Rayshade, or other BOOGA components, like an off-screen OpenGL renderer.
- **Upload of compressed files:** Geometric scene definitions and file sizes may be too big to be uploaded to DeepRay in an affordable time. File compression could cut down transfer times significantly.
- **Drag-and-drop support:** A good extension of the upload facility would be a drag-and-drop support, where the user can drop the icon of the file he wants to upload over the text input field of the HTML form. Adding drag-and-drop support would be a substantial enhancement of the file requester dialog.
- **Adding more computing power:** Due to the scalability of the IBM SP supercomputer, a performance increase could be achieved by adding more nodes, resulting in shorter rendering times when using the BOOGA parallel raytracer.
- **Clustering of supercomputers:** One can think of connecting several supercomputers to one virtual supercomputer over the Internet using a load balanced queuing system. It would also be convenient if the application itself were responsible for localizing the appropriate computing resources for CPU-intensive calculations.

Conclusions

By a combination of some of the Internet's standard software components we implemented a platform- and location-independent user interface to a supercomputing application. This concept can be adapted to all applications of supercomputers which take one or more input files as parameters and generate a resulting file which can be processed at the user's workstation. The proposed user interface makes high performance applications available to many kinds of users and can significantly reduce education costs.

References

1. Amman S., Streit C., Bieri H.: BOOGA - A Component-Oriented Framework for Computer Graphics. Proceedings of GraphiCon' 97 Moscow, 193-200 (1997)
2. Barbera H., Skaramenta A.: DRS - Distributed Raytracing System. International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA'95) (1995)
3. Flanagan D.: Java in a Nutshell. O'Reilly International Thomson (1996)
4. Gundavaram S.: CGI programming on the World Wide Web. O'Reilly International Thomson (1996)
5. IBM Publication SC23-3989-00: Using and Administering IBM Loadleveler. (1997)
6. JavaScript World. http://www.jsworld.com/ (1997)
7. Krishnan et al.: MMM: A Web-Based System for Sharing Statistical Computing Modules. IEEE Internet Computing 59-68 (1997)
8. McCool R.: The Common Gateway Interface. http://hoohoo.ncsa.uiuc.edu/cgi/overview.html (1994)
9. Network Working Group: Form-based File Upload in HTML. http://sunsite.auc.dk/RFC/rfc/rfc1867.html (1995)
10. Rice J., Farquhar A., Piernot P., Gruber T.: Using the Web Instead of a Window System. CHI '96 (1996)
11. Siegel D.: Creating Killer Web Sites. Hayden Books (1996)
12. SPCS Report No. 25. Massively Parallel Processing: Computing for the 90s. The Superperformance Computing Service, Palo Alto Management Group (1994)
13. The Comprehensive Perl Archive Network (CPAN). ftp://sunsite.cnlab-switch.ch/mirror/CPAN/
14. Wall L. et al.: Programming Perl, Second Edition. O'Reilly & Associates, Inc. (1996)

Address: Dipl. Inf. Bernhard Bühlmann, Prof. Dr. Hanspeter Bieri
Institute of Computer Science and Applied Mathematics
Neubrückstrasse 10
CH-3012 Berne, Switzerland
Phone: ++41 (0)31 631 86 81 Fax: ++41 (0)31 631 39 65
Email: {buhlmann, bieri}@iam.unibe.ch