Managing Large Financial Data with the SAS® System and the WEB: The Sequel

Paul J. Ratnaraj
The Wharton School of the University of Pennsylvania

Abstract

During the past decade, use of large and complex data sets has grown substantially. This usage explosion has compelled schools to experiment with better access and management methodologies for a diverse and dynamic user environment. The Web is poised to be the best medium for delivering data optimally across all platforms. However, the Web is still in its infancy both in terms of its use and the tools required to develop stable applications. The Wharton School was one of the pioneers in using SAS for information delivery in an academic environment. In 1993 we started using SAS to manage the delivery of our data. In late 1995 we started providing access to large financial data sets using SAS via the Web. This paper highlights our transition from using FORTRAN & SAS based delivery mechanisms to our current data delivery paradigm. Additionally, this paper discusses the Web and its components such as CGI scripts, HTML, and Java, and the importance of security. The paper concludes with the pros and cons of using SAS via the Web to deliver information.

Introduction

One of the most compelling visions of modern computing is the promise of easy access to vast data resources. Microsoft's CEO Bill Gates speaks of "information at your fingertips" and former Apple CEO John Scully envisioned a "knowledge navigator" automatically sifting through vast data repositories.

Such ideas, while deceptively easy to imagine, are often difficult to achieve. At the Wharton School of the University of Pennsylvania, a program to bring large financial data sets to faculty and students - Wharton Research & Data Services (WRDS) - has taken a significant step toward making easy access a reality.

The Wharton School, the premier business school in the nation, has over 3,000 undergraduate students, 1,500 graduate students and 200 standing faculty with varied computing needs. Like other academic institutions, use of large research data sets has grown substantially in the past decade. In addition, the data sets have grown significantly in size. What was once the domain of finance and accounting has now become the province of management, marketing, and other disciplines. The Wharton School data range from examining world-wide investment patterns to handicapping the box-office success of upcoming motion pictures. Increasingly, faculty at The Wharton School also use the data sets for instructional course-work assignments.

Managing these data sets requires extensive resources in terms of time, personnel and support at The Wharton School and in other schools nationwide. Increased usage has compelled schools to experiment with better access and management methodologies. Some have opted for database management products, while others have home-grown systems developed to meet their needs. In 1993, The Wharton School made the transition from using traditional languages like FORTRAN and C, and in-house developed systems, to using SAS to manage the delivery of the data. In late 1995, we started providing access to the data using SAS, Perl, Java and the Web.

Today other schools are using WRDS as an additional means of obtaining and analyzing data.

The Way Things Were

Data sets have been used at The Wharton School for many years. However, previous methods for delivering data were far from ideal. The financial data sets widely used at The Wharton School include market research data (such as CRSP, Fama and Market Indices), corporate data (such as Compustat), and banking and insurance data (such as BEST and FDIC). Prior to 1993 data sets were stored on large VMS/VAX systems, and users had to run FORTRAN programs to analyze or extract data. An increasing number of users preferred working with familiar desktop applications such as Systat or Excel. But working with the data using desktop tools required that the user be familiar with the formats of the data sets, FORTRAN programming, mainframe to PC file transfer techniques, the VMS operating system, and the data import format of desktop software. As Michael Phelan, Associate Professor of Statistics, at The Wharton School, pointed out, "These data access techniques were functional, but not for the timid."

This approach was not only cumbersome for faculty and students, but also difficult for The Wharton School's computing staff to support. To increase access speed we wrote in-house indexing programs. To help new users we provided interactive modules and help screens. Any changes in data format required updating everything written in-house and extensive in-house programming support.

For all the effort required, users, accustomed to point-and-click graphical interfaces on personal computers, were increasingly dissatisfied with this arcane, multiple-step procedure.

Several alternatives were considered to provide easier access and improved data set management (Ratnaraj, 1994). In 1993, WRDS was implemented with the following components:

(Using SAS (and SAS/ASSIST) to extract and analyze data
(Managing data sets centrally while providing network access (through NFS mounting) to the complete series of data on UNIX systems throughout The Wharton School. In 1994 the data sets were also available in Windows format
(Providing X-Window access to UNIX systems from The Wharton School's labs and classroom teaching stations

SAS best met The Wharton School's objective of offering a single, unified tool for data management and analysis. Extracting data required only a few lines of code in SAS, versus several hundred lines of FORTRAN. Once extracted, a data set could immediately be used by a wide range of SAS' analytical procedures.

SAS/ASSIST offered a point-and-click graphical user interface, an online tutorial, and help screens. These helped users and reduced requirements for training and documentation. Although, not a great alternative, the SAS/ASSIST' VT100 interface allowed students and faculty to access WRDS from home or a location away from The Wharton School.

WRDS offered a number of advantages for faculty and students, and met the goals of universal availability, ease of use, and reduced maintenance and support.

The entire collection of data sets was available as a local resource, either by using SAS for Windows, or by using X-Windows on shared departmental systems and faculty desktop systems. Students could access these data sets with the same graphical environment by using X-Windows in the student labs. Users could now manage and analyze the data using a single tool. Because the same data tool was used for all data sets, users could easily analyze data across different data sets.

Professor Richard J. Herring, stated that "the beauty of the system is that wherever you go, whatever system you use at the School, the data is accessible and appears in the identical form." According to Dr. Herring, the key benefit this provides is that it "reduces the time researchers spend extracting data and allows them to concentrate on their analysis."

World Wide Web

While The Wharton School was taking a small step in delivering data to its faculty and students, Conseil Europeen pour la Recherche Nucleaire (CERN), European Laboratory for Particle Physics in Geneva, Switzerland was taking a giant leap in delivering information to the whole world. In March 1989, Tim Berners-Lee, a researcher at CERN, proposed a HyperText system to enable sharing information efficiently between its (CERN) members. The key components of the proposal were:

(An user interface that would look and feel the same across all platforms
(Enable the user to access different kinds of information (text files, graphics etc.) via the above mentioned interface
(A provision that would allow any user to access any information on the network

By late 1990 a line mode interface called www and an operating prototype of the World Wide Web (WWW), an Internet service that enabled interfaces to display styled text, graphics, and multimedia at the click of the mouse was completed. As of December 1997, the prototype has progressed to over 1.8 million web servers, the familiar www.company.com address and the term “dot” have become vernacular.

Internet, Intranets & Extranets

The Internet, comprising interconnected computers, started as ARPANET (Advanced Research Projects Agency Network), a government funded network to enable remote educational and military research sites to exchange information using the TCP/IP protocol. Until the arrival of the Web, the Internet remained a resource used primarily by academic and government institutions. The point-and-click transfer of information via the Web gave commercial organizations another medium to advertise, disseminate corporate information, and display products and services for sale.

The Wharton School installed its Web server in late 1995. The Wharton School’s Web, • HYPERLINK "http://www.wharton.upenn.edu/" ´•http://www.wharton.upenn.edu•, made available corporate and computing documentation. The information delivered via the Web server was static; you clicked on a link and were delivered the appropriate data.

One of the limitations of WRDS was the unappealing VT100 interface for those accessing the data from a remote location. The Web seemed perfect to enable users to select and choose the data using the Internet. However, since the data sets were proprietary we could not release their contents to the general public on the Internet. The Wharton School decided on adding another Web server (• HYPERLINK http://inside.wharton.upenn.edu ´•http://inside.wharton.upenn.edu•) that restricted the information to the Wharton School community. This topology referred to as an Intranet is often a subset of the Internet where access is restricted to a particular community. The information access is controlled by the server yet the graphics, easy-to-use quality of the Web and the topology of the Internet remains the same. The Intranet allowed The Wharton School to maintain a common interface and install security mechanisms while protecting proprietary information from unauthorized users.

Today the growth of Intranets is much more rapid than the Internet. Zona Research Inc. (• HYPERLINK "http://www.zonaresearch.com/" ´•http://www.zonaresearch.com•) reports that predicted expenditures for Intranet products and services will exceed the market for Internet products and services by a ratio of 2 to 1 for the foreseeable future. Though the Intranet is growing faster than the Internet, there is significant concern that the Internet is slowing down due to increasing traffic and growth. Many large companies have opted for implementing private networks called Extranets that use the TCP/IP

protocol to communicate and share information company- wide, and with their close business partners that may be in remote locations. The Extranets have evolved from companies that have already had private networks in place generally using leased or direct telephone lines. The automobile industry has long allowed its parts suppliers to directly access many of the services to increase efficiencies in production. For example, just-in-time inventory lets suppliers view production quotas of a car manufacturer and ship the appropriate quantity necessary to meet the quota. Previously such solutions required multiple telephone connections. Today, you can point and click to view the data.

WRDS and the Web

The Wharton School had gained considerable experience with client-server technology in implementing WRDS in 1993. Thus it was easy to develop the necessary programs to create a Web interface to the data sets. In late 1995, WRDS was available on the Wharton School Intranet and access was granted to authenticated users via the Web.

Today WRDS has gone through multiple iterations and makeovers. We started with basic HTML and have now progressed to sophisticated dynamic images within frames and other means of displaying information. Most of the programming is done using CGI and Perl. We have also used Java to overcome the limitations of HTML.

WRDS is also available as a data service for other institutions. Currently, Stanford University, University of Southern California, Columbia University, & UC Berkeley are using WRDS as an additional means of obtaining data via the web.

Components of The Web

The Web technology is still in its infancy and thus constantly evolving. By the time you read this many of the components described below may have undergone major revisions but the fundamentals will remain pretty much the same for the foreseeable future.

Server

The Web uses a client-server architecture. The user-computer that requests the information is the client while the site-computer that delivers the information is the server. NCSA's HyperText Transfer Protocol (HTTP) server was the first popular server available free of cost. HTTP is often referred to as HTTPD because it is a daemon that runs in as a background process.

Servers are evaluated on the following factors: performance, ease of installation, manageability and security. In a Windows NT environment Microsoft's Internet Information Server is ideal for managing since it relies on NT's administration, authentication and management utilities. Netscape, on the other hand, is available for most platforms and uses its own database for authentication. Authentication is a lot more complex in the Unix environment if you do not use Netscape's authentication scheme. The Wharton School's computing structure is distributed, with many departments managing their own servers and computing resources. This means that users may have accounts on their respective machines as well as on the central machine. Since the Web servers are centrally managed, using Netscape's default authentication scheme will result in the users having another account and password to manage. Developing a unified authentication scheme that enabled Netscape's server to query the distributed computers for verification involved countless hours of programming Netscape's API.

Netscape's recent acquisition of Kiva Software Corporation (• HYPERLINK "http://www.kivasoft.com/" ´•http://www.kivasoft.com•) makes it a leading provider of enterprise Internet, Intranet & Extranet applications. However, the highly acclaimed Kiva Server is expensive. One should weigh the costs and benefits of deploying the Kiva Server within an organization.

WebCompare (• HYPERLINK "http://webcompare.iworld.com/" ´•http://webcompare.com•) is an excellent starting point for evaluating Web servers. SPEC's (• HYPERLINK "http://www.specbench.org/" ´•http://www.specbench.org•) SPECweb96 can be used to help determine which Web-server software performs best on a particular set of hardware systems and network connections.

Browser

To access the Web it is necessary to run a browser on your computer. The browser is an application that knows how to interpret and display documents accessed from the Web server. Microsoft and Netscape currently lead the market in browser products. The browser technology is constantly undergoing changes and thus many of the older browsers may not be able to use the features of the current ones.

HTML

As mentioned earlier, one of the provisions of CERN's HyperText proposal was the uniformity of display across all platforms. To obtain this uniformity HyperText Markup Language, or HTML, was established to define a structure and format of the document to be displayed on the Web. HTML is based on SGML (Standard Generalized Markup Language), an international standard for defining and using document formats using text-based markup. When you retrieve a document you generally find it is formatted and generally aesthetically pleasing. HTML commands, or tags, are inserted around blocks of text to describe what the text is. For example, you could mark some text as a heading, create paragraphs, format text to be displayed in italic form, etc. HTML also includes tags for including images within documents, forms that accept user input, and links to other documents or sites on the Web. Today there are many software applications that generate HTML code, including word processors like Microsoft Word. Although substantial improvements (currently version 4.0) have been made to HTML since its inception, HTML is a simple language suited for the display of small and simple documents. While easy to learn and implement, HTML does not provide the flexibility or customization needed for many complex documents and data.

Plugin & Dynamic HTML

A plugin is a program written in C or C++ that extend a browsers capabilities. They are provided separately by vendors. Although plugins are easy to install and use, not all browsers support them. Moreover, a user will have to install the plugin before it can be used.

Dynamic HTML, implemented by both Microsoft and Netscape, enables a web page to interact with the user on the fly. In the past, once HTML was loaded it could not be changed without refreshing or loading another page. Dynamic HTML enables developers to put programming logic into the page. A user can thus change the font or display attributes on the fly without the need for a plug-in or a request to the server. However, there are differences between Microsoft's implementation of Dynamic HTML and that of Netscape's.

XML

SGML is very comprehensive, complex and capable of a multitude of tasks including handling large and complex documents, and managing large data repositories. Most Web users will not need all the capabilities found in SGML. It would also be very expensive to implement SGML in its entirety. The Extensible Markup Language (XML) is a simplified subset of SGML. It has the key benefits of SGML and is presented in a manner that is easy to understand and implement.

XML is ideal for Web applications that:

(Require the Web client to mediate between multiple databases. For example, you download the stock prices from WRDS and would like to analyze it using your own application. In HTML you would have to either input the data by hand into your application or cut & paste. Using XML the language would know the data structures of both the databases and you could just drag the data into the application directly on the Web.

(Processing load on the client. Some tasks, like booking an airline flight, are compute-intensive and would strain the server. XML enables you to design applications that use the computational cycles of the client.

(Different views of the data depending upon the user. For example, a user may see the installation instructions in the language selected.

(Automatically deliver customized information to the user based on what an intelligent application thinks would be of interest to the user. For example, movies with selected leading actors, or news on a local sports team.

Dan Connolly, the architecture domain lead for the W3C (• HYPERLINK "http://www.w3c.org/" ´•http://www.w3c.org•), cautions that key pieces of the XML architecture is still in flux.

CGI

CGI, or Common Gateway Interface, is a method for the Web server to accommodate programs that may use external applications like SAS or databases like Oracle. The “Common” stands for platform independence, while “Gateway” describes the relationship between the server and the external application. “Interface” is the mechanism controllable by any suitable language. CGI is to the Web what batch files are to DOS and shell scripts are to UNIX. CGI is the most common technique for creating dynamic Web pages by linking servers to external applications. All financial data delivered by WRDS via the Web makes use of CGI. Most of the popular programming languages like C, C++, and Visual Basic can be used for CGI programming. Additionally, Unix shell scripting languages like Bourne, C, and Korn shells can be used for CGI programming. Perl (Practical extraction report language) is probably the most popular CGI programming language since its platform independent structure makes it easy to develop an application quickly. Perl combines many of the best features found in C, sed, awk and sh and presents them in a syntax similar to C. Perl has excellent string manipulation capabilities and succinct ways of solving many programming problems. Perl is an interpreted language and is therefore slower than C or C++. C, on other hand, requires programming effort to manipulate strings, which is a common task in Web applications.

Choosing a programming language depends upon the familiarity with the language as well as the support that is available for the language both internally and externally. At The Wharton School we use Perl, C, and Visual Basic depending upon the needs of the application and the operating system it runs on.

Alternatives to CGI

CGI is essentially a batch oriented process. Every time a user clicks on a link the CGI program starts all the necessary programs and shuts them down when it is complete. As traffic increases each URL

request starts a new process thus loading the machine running the Web server. Additionally, CGI is considered stateless because it does not save any information on the state of the transaction it has launched. Some of the financial data sets are over four gigabytes in size. We had to develop code that restricted users from generating queries that either downloaded all the data or went into time consuming searches that could not be aborted by the user.

Multiple alternatives to CGI are currently available. However, the alternatives are not a panacea for CGI's limitations. FastCGI can keep track of state information and also run the CGI program as a process external to the Web server. However, FastCGI has not been embraced by any of the major Web server developers including Microsoft and Netscape. Instead, Microsoft and Netscape are encouraging the use of Application Programming Interfaces (API's) to directly access server executables. Although the API's are openly published they are proprietary. Moreover, Microsoft and Netscape have not agreed on a common API.

Many companies, including SAS and Borland, have developed tools that mask CGI's limitations by providing their own database access middle-ware.

SAS/IntrNet

The SAS/IntrNet software acts as a gateway between the browser and SAS. The gateway, using CGI protocol enables you to access SAS data sets and SAS applications. SAS/IntrNet is ideal for those who do not have programming skills in CGI or a programming language. With basic skills in HTML you can easily create a page that accesses and analyzes data from SAS data sets. Although SAS is continually developing utilities which makes it easy to access and display the results on the Web, SAS/IntrNet is limited in creating custom or complex applications. At Wharton we decided to continue with Perl & Java because we needed to pre-process the form data for errors before submitting it to the SAS engine. It is also easier to pre-process at the CGI level rather than execute a SAS program to pre-process. We also observed that SAS/IntrNet executes slightly faster than our implementation using Perl.

Java

Java was introduced in 1995 by Sun Microsystems. Today, the promise of its potential has reached stratospheric levels. Java has become so popular that over a million developers are using it or have plans to use it. Although, many consider Java to be still in its infancy, it has not stopped organizations from deploying Java in production environments.

Java is an object oriented language that was developed originally to enable small devices to communicate. The developers quickly found that the language could be used very easily and effectively on the Internet as TCP/IP was integrated in the language. Today Java programs have the capability to do almost anything that can be done in other languages especially C++. The major difference between Java and other languages is that Java is designed to be platform independent. That is, you can take programs written and compiled in Java using an UNIX platform, and run the executables on a different platform such as Windows 95. The portability is obtained by Java's Virtual Machine (JVM) or Java Sandbox - an environment on top of the operating system. Java abstracts the details of the operating system and the hardware from the programmer thus making all the platforms appear to be same.

The portability advantage is offset by slower speed since Java is essentially running in a second computing environment. On the other hand C or C++ will enjoy speed benefits because they can be compiled to native code and make use of all the resources without intermediary processing. The sandbox design was developed purely to protect against malicious applets or applications that deliberately corrupt systems. However, Java is not completely secure and you are urged to research the precautions needed for your application. The Computer Science department of Princeton University (• HYPERLINK

http://www.cs.princeton.edu/sip ´•http://www.cs.princeton.edu/sip•) offers an excellent guide to Java's security.

Developments are underway to increase execution speed. Microsoft's implementation of JVM is currently the speed champion. Symantec Corporation's Visual Café for Java is offering the capability of producing a Windows executable from Java code. Similarly, other vendors like IBM, SunSoft, and Microsoft are developing native compilers for Java.

Speed is not the only hurdle that Java faces in the years to come. Java is facing the development problems of most languages. Bugs are frequently reported and generally resolved over time. We looked to Java to develop a simple search feature – a user could type the first few characters and automatically be shown all the variables in the data set having the typed characters – in the web page. During testing it was apparent that there was a bug that slowed down the whole applet when more than 500 elements were listed in a drop box. The bug has since been fixed but the version is not currently supported by either Netscape or Microsoft.

The built in security features of Java limit the number of things you can do in an application. In the earlier version of Java you could not access local files. Products like digital signatures are being developed to overcome this hurdle. However, the enhancements may pose new security problems. Other hurdles include support for development tools, libraries, and API's which is currently limited and a far higher learning curve than HTML which could lead to delays in implementation.

ActiveX

Microsoft introduced Object Linking and Embedding (OLE) in 1989 designed solely for allowing Windows applications to communicate with each other. Pasting an Excel spreadsheet into a Microsoft Word document involves OLE. ActiveX is basically OLE with features added to handle the Internet technology. There is a huge repository of ActiveX controls available to ease the burden on Web programming. Controls are objects that have pre-defined properties which one can manipulate and thus create applications quickly. For example, if you wanted to draw text boxes you could buy a TextBox Control from a commercial vendor and just drag the box to fit your needs. The conventional method – writing from scratch - would be to specify exactly the dimensions and the various properties using the syntax of the language. Java programmers can use ActiveX controls directly from Java programs. Although it seems that ActiveX is competing with Java the reality is that ActiveX has a few advantages over Java applets:

(Fast execution speeds
(Read and Write access to the local files
(Easily interface with other languages and tools like Visual Basic, C++, Shockwave, and Adobe Photoshop.

The biggest drawback about ActiveX is that its security model relies on the user's judgement to accept or deny ActiveX applications. Although, ActiveX's digital signature is similar to Java's implementation of signed applets one clear distinction is that ActiveX requires either full trust or no trust at all. In Java you can select your browser to partially trust the applet by denying it certain capabilities.

Active Server Pages, JavaScript and VBScript

Active Server Pages (ASP) developed by Microsoft is a server-side application environment that can combine HTML pages, scripts, and other components to create powerful applications. ASP provides the functionality of CGI with better performance. With ASP you can easily connect to many of the databases using ODBC.

Other than the similarity in the name, JavaScript has very little to do with Java. JavaScript from Netscape and Visual Basic Scripting Edition (VBScript) from Microsoft allow the creation of client-side execution. VBScript is based on the Visual Basic language and is currently limited to the Windows platform. The scripting code is embedded in the HTML page. This means that the code is downloaded to the client every time the page is accessed. Updating the client is easily done by changing the files on the Web server. Both scripting languages allow standard programming logic such as loops, conditional statements, and math operations. Additionally, they can call on outside resources. For example, VBScript can execute both ActiveX objects or Java applets that are stored on the client. JavaScript can execute Java applets on the client machine. Scripting languages are ideal for simple applications, like data entry screens. The screen could have the necessary input checks on the client side rather than sending back the information to the server, thus minimizing network traffic.

With the emergence of XML the usage of Script languages is expected to decline in the near future.

Security

In the past year the vulnerability of Web servers has been dramatically exposed by the defacing of Web pages of the U.S. Airforce, the U.S. Department of Justice, and the Central Intelligence Agency. Dan Farmer, co-developer of the controversial Security Analysis Tool for Auditing Networks (SATAN) reports on his web site, • HYPERLINK http://www.trouble.org ´•http://www.trouble.org•, that he used SATAN to surreptitiously probe over 1700 web sites for security flaws. Over 60% of the sites had some flaw or another. Farmer added that there were few sites that detected his unauthorized probe.

Security is probably the biggest challenge for organizations planning to use the Web. The UNIX system was designed to be open and thus has many holes in its security system. As mentioned earlier most Web servers use UNIX and therefore share this vulnerability. This is not to say that other systems are less vulnerable. Information Week (• HYPERLINK http://www.informationweek.com ´•http://www.informationweek.com•) discovered and published a hole in Windows NT's security. Common vulnerabilities in UNIX that most hackers are aware of are: finger, NFS, DNS, uucp, sendmail, and telnet. Agencies like the Computer Emergency Response Team (• HYPERLINK http://www.cert.org ´•http://www.cert.org•) notify the Internet community via newsgroups and email about known intrusions or vulnerabilities and solutions if available. Additionally, vendors post similar notifications and solutions.

Commercial vendors provide software solutions called firewalls to prevent unauthorized access and tampering of the network and other computing services. Firewalls, like most other software, come in different flavors and use different techniques to provide security. Common among them are packet filtering, application gateways and state inspection.

Packet filtering is generally done at the router and is very fast but provides low security. In packet filtering the router or the application examines one packet at a time. This makes it cumbersome to apply rules, and to think of every possible rule to provide access to authorized users. To prohibit telnet traffic, the router could be programmed to deny any packet that has port 23 in the destination header. However, this same logic poses a problem when securing FTP. FTP uses two connections; the first is made to port 21 and when the request is made to move data, it initiates another session to choose a random port between 1,024 and 65,535, and tells the server to use this port as the destination to send data. Since it is a random number, a typical router cannot tell which number it will use. Programming the router to allow destination ports of 1,024 through 65,535 would expose these ports to potential hackers. There are a few vendors like Checkpoint Software Technologies (• HYPERLINK http://www.checkpoint.com ´•http://www.checkpoint.com•) that track the state of the FTP session and grant a temporary access to the port.

Application gateways reside between the server and the client. Application gateways examine all layers, including their content before passing the data. The server only accepts requests from the gateway and therefore can be shielded from the outside world. Proxy agents are created for each protocol such as Telnet, FTP, and HTTP. Creating agents enables you to customize the filtering for a particular service. Application gateways tend to be slow because they have to check everything. In addition, upgrades to the protocols require upgrades to the proxy agents.

To overcome the limitations of the above methods, hybrid products using state inspection methodology are available. The software maintains states of previous communications or applications and enables the packets to be denied or allowed access based on the state information. This is very secure but complex to implement. It must be supplemented with additional authentication methods on the client machine.

The Secure Socket Layer (SSL) protocol is the standard for moving confidential information between a client and a specific Web server. The protocol ensures that your information is transmitted privately and reaches the destination without alteration. Both Microsoft and Netscape support SSL. You have to obtain a digital certificate from a certified authority such as Verisign (• HYPERLINK http://www.verisign.com ´•http://www.verisign.com•) and then configure the server to use the SSL protocol.

Securing your network is not very difficult, but it does require somebody who is knowledgeable and dedicated to the task. However, one has to recognize that it is almost impossible to close all holes and still make it easy for authorized users to obtain information efficiently.

Conclusion

The Web seems to be the best medium for delivering data to the masses. Its platform neutrality reduces the maintenance of systems across multiple platforms. Having a uniform user interface also reduces support and maintenance. Navigating the Web is just a point-and-click away.

It is imperative that considerable thought be given before one implements a Web site. Some of the key strategies for a successful Web implementation include:

(Ensuring that the site is scalable. The Wharton School started with delivering static information, progressed to delivering interactive information and now is delivering real-time information. In addition there are multiple divisions from Admissions to Zoology which are setting up their own Web pages that require additional computing resources.
(Making the interface intuitive to enable users to quickly and easily obtain information.
(Making sure that you are consistently delivering current and valuable content. Graphics are great to garner attention but it is the content that will bring them back.

Constant monitoring for integrity and system up-time, rapid development, and high access speed should become common tasks for an organization to succeed in the Internet world. Adequate personnel and budgetary considerations must be dedicated for the above tasks.

The Web is still a few years from the ease of WRDS wherein data access and analysis is done locally in the framework of SAS/ASSIST. Analyzing on the Web, similar to the SAS framework wherein data management and analysis is integrated, is still very much in its infancy. We have successfully built adequate queries and can throw in a few statistics for the user, but clearly feel it would be a monumental task to provide the capabilities of a full-blown user-empowered interactive statistical analysis. SAS (• HYPERLINK http://www.sas.com ´•http://www.sas.com•) has made considerable efforts in delivering various tools that deliver data via the Web. Unless SAS delivers analytical capabilities the Web will be used primarily to obtain data and not information. Until Java matures one cannot expect it to be used for manipulating large data sets.

The immaturity of the Web should not deter anyone from devoting adequate resources for developing applications. The Web clearly has the potential to make data delivery far easier and more elegant than ever before.

References

Ratnaraj, Paul and Carol Katzman. “Managing Large Financial Data with Ease: CRSP, COMPUSTAT, Etc., with SAS” Proceedings of the Nineteenth Annual SAS User’s Group International Conference. 19 1994: 659:666

Author

Paul J. Ratnaraj
Director - Core Systems, Data Services & Operations
Wharton Computing & Information Technology
The Wharton School
3620 Locust Walk
Philadelphia, PA 19104-6302
Tel: (215) 898-7602
Fax: (215) 573-6073
email: ratnaraj@wharton.upenn.edu