Managing Large Financial Data with the SAS® System and the WEB

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. Wharton 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 and the Web. This paper highlights our transition from using FORTRAN based delivery mechanisms to our current data delivery paradigm. Additionally, this paper discusses the Web and its components, like CGI scripts, HTML, and Java, and the importance of security. The paper concludes with the pros and cons of using SAS with 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 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 - The Wharton Research Data System (WRDS) - has taken a significant step toward making easy access a reality.

The Wharton School, the premier business school in the nation, has over 2,000 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. Wharton data sets range from examining world-wide investment patterns, to handicapping the box-office success of upcoming motion pictures. Increasingly, faculty at Wharton 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 Wharton 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 Wharton 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 and the Web.

The Way Things Were

Data sets have been used at Wharton for many years. However, previous methods for delivering data were far from ideal. The financial data sets widely used at Wharton 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 Wharton, points 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 Wharton's computing staff to support. To increase access speed, Wharton wrote in-house indexing programs. To help new users, Wharton provided interactive modules and help screens. 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. Developing an in-house system would be costly, time-consuming, and difficult to maintain as the technology changed. Commercial database management systems provided excellent data management capabilities and convenient access tools, but lacked strong analytical tools and were not suitable for time-series data. Commercial data access packages such as Fame, DART, and Intelligent Query offered good data manipulation tools, but also lacked sophisticated analytical abilities and required extensive programming to convert the wide selection of data sets used at the Wharton School (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 Wharton
- Providing X-Window access to UNIX systems from Wharton's labs and classroom teaching stations

SAS best met Wharton's objective of offering a single, unified tool for data management and analysis. SAS has long been popular for data analysis in the academic environment. The release of version 6.09 greatly simplified reading external data, as well as the aforementioned special data sets. 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 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, SAS/ASSIST' VT100 interface allowed students and faculty to access WRDS from home or a location away from Wharton.

In 1994, we converted the most widely used data sets into SAS/WINDOWS format. This conversion allowed students and faculty, who were more at ease with SAS for Windows, to access and analyze the same data. The ability to use SAS for Windows also helped instructional technology by allowing students to cut and then paste the results of the analysis generated in SAS into their reports created on a word processor.

WRDS offered a number of advantages for Wharton 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 throughout Wharton, including 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 Wharton 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. In May 1991, the www interface was made available on the CERN machines. The CERN team released information on the WWW protocols, capabilities and files to the Internet community in August 1991. On January 15, 1992, CERN made publicly available its www interface. Within the year there were well over 50 WEB servers. In January of 1993, the first graphical user interface, called *browser,* for the X Window System and Macintosh surfaced. In February, a X Window System browser known as Mosaic was released by National Center for Supercomputing Applications (NCSA). Mosaic was seen by many as the application that would promote the Internet, the way Lotus 123 helped the personal computer gain fame. Netscape further fueled interest in the Web and by mid 1995 there were well over 15,000 known public Web servers. In June 1996, Netcraft (*www.netcraft.co.uk*) reported that it had successfully contacted 252,000 Web servers. Today almost all major advertisements display the familiar *www.company.com* address and the term "dot" has become vernacular.

Internet, Intranets & Extranets

The **Internet** started as ARPANET (Advanced Research Projects Agency Network), a government funded network to enable remote educational and military research sites to exchange information. One of the main goals was to allow the addition and removal of new computers of many different types. In 1971, there were 23 hosts connected to ARPANET using NCP(Network Control Protocol). By the early 1980s, commercial, academic and ARPA research on networks led to the development of the TCP/IP (Transmission Control Protocol/Internet Protocol) network protocol, the language that computers connected to the network used to talk to one another [Hardy,1993]. During the early 1980s, a gradual conversion to the TCP/IP protocol was made by all the interconnected research computers of ARPANET. On January 1, 1983, TCI/IP was officially adopted as the standard for ARPANET. This also led to one of the first definitions of an "internet", as a connected set of networks, specifically those using TCP/IP, and of the "Internet" as TCP/IP connected internets. ARPANET became the backbone - the physical connection - of the new Internet. The Internet is also comprised of many services including the Web, email, Gopher and FTP. By 1984 there were over 1,000 Internet hosts. Netcraft estimated that there were over 9.5 million hosts connected to the Internet in June 1996.

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 software gave commercial organizations the impetus to join the Internet in hordes. The Internet thus became another communications medium for organizations to advertise, disseminate corporate information, and display products and services for sale.

Wharton installed its Web server in late 1993. The Wharton Web, *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. However, during implementation it was apparent that the Web could be used for more than just delivering static information. The technology, although not quite refined, was available to provide an interactive medium to deliver information.

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. Wharton decided upon using two Web servers; the Internet to serve the worldwide community, and the **Intranet** (*inside.wharton.upenn.edu*) to serve the Wharton community. The Intranet can be likened to Local Area Networking using the TCP/IP protocol, 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 Wharton to maintain a common interface and install security mechanisms. Wharton also saw an increase in transmission speed since the information did not have to go on the crowded Internet. Wharton 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 early 1994, WRDS was available on the Intranet and access was granted to authenticated users.

Today the growth of the Intranet is much more rapid than the Internet. Zona Research Inc. (*www.zonaresearch.com*) reported that predicted expenditures for the 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 may slow 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. Most of the Extranets that parallel the Internet are currently being deployed by large Internet service providers.

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 will 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 in that 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. Today there are countless web servers. Netscape and Microsoft currently lead the market in server sales.

Servers should be evaluated upon 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 Windows NT's administration, authentication and management utilities. Netscape, on the other hand has general purpose servers as well as servers tailored to a specific function. *Enterprise* is the general purpose server while *Live Payment* is specific for credit card transactions and other payment services. Authentication is a lot more complex in the Unix environment if you do not use Netscape's authentication scheme. Netscape by default uses its database for authentication. Wharton's computing structure is very distributed with many departments managing their own servers and computing resources. This means that users have accounts on their respective machines. 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.

WebCompare (*webcompare.iworld.com*) is an excellent starting point for evaluating Web servers. SPEC's (*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. Netscape offers different browsers with different levels of features. 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. 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 or Uniform Resource Locators (URL) to other documents or sites on the Web. Today there are many software applications that generate HTML code including word processors like Microsoft Word. However, the HTML authoring tools do not provide the flexibility needed for some complex web sites and you may have to resort to hand-coding some of the elements.

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 make 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 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 slower than C or C++ which are compiled languages. 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 Wharton we use Perl, C, and Visual Basic depending upon the needs of the application.

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 provides *htmSQL*, a gateway between the browser and SAS data sets.

Java

CGI and HTML are server-side tools, in that they run on the server. Similarly, on the client side we have languages and scripts like Java, JavaScript, and VBScript. 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 200,000 developers are using it or have plans to use it. More than 60 vendors, including Microsoft, Apple, and Netscape, have licensed Java from Javasoft (*www.javasoft.com*).

Java is an object oriented language that was written solely to take advantage of the Internet. Java programs have the capability to do almost anything that can be done in other languages especially C or C++. The major difference between Java and other languages is that Java is platform neutral. 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. By default, Java abstracts the details of the operating system and the hardware from the programmer thus making all the platforms appear to be the same. However, if the need arises, one can develop programs that use system resources for a particular platform. Java is also structured with integrated TCP/IP access, and security features.

Java nevertheless has some significant hurdles to overcome. Java is an interpreted language thus raising the issue of execution speed compared to C and C++. Developments are underway to increase execution speed but as yet nothing is on the horizon. The built in security features limit the number of things you can do. Support for Java in development tools, libraries, and API's is very limited. Compared to HTML, Java has a far higher learning curve and could lead to delays in implementation.

JavaScript and VBScript

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.

JavaScript may also be run on the server itself. Server-side programming is easier than creating a CGI application. Netscape's Enterprise server comes with a JavaScript compiler that maintains states across multiple requests. This enables faster access to databases. To enhance speed, the JavaScript is compiled and stored on the server.

Security

In recent months 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, *www.trouble.org*, that he used SATAN to surreptitiously probe over 1700 web sites for security flaws. Over 65% 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(*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 (CERT) 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*, a*pplication gateways* and s*tate 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 (*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 becoming the standard for moving 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 (*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. Wharton 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.

We found UNIX to be the best platform for developing CGI scripts. Windows does not provide an I/O redirection utility, available in UNIX, that allows you to get and put data respectively. This limitation in Windows creates individual processes for each execution of the program that requires a ‘read’ or ‘write’. As such multiple requests would result in stressing the computer’s resources. Microsoft recommends using its server product along with Dynamic Link Libraries to overcome the CGI limitation. Currently, Netscape does not have an alternate solution to the Windows limitation.

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 (*www.sas.com*) has made considerable efforts in delivering various tools that deliver data via the Web. Most of the tools are basically browse or query utilities. SAS plug-ins are also available for the Windows environment. SAS has written a Java Database Connectivity (JDBC) driver to enable an Java applet to run on the client and browse through a data set that is on the server. The speed problems of Java are quite evident in SAS' demonstration of Java. 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

Hardy, Henry. "The History of the Net." Master's thesis, School of Communications, Grand Valley State University, Allendale, Michigan, 1993. ftp://umcc.umich.edu/pub/users/seraphim/doc/nethist94.html

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 19104-6301
Tel: (215) 898-7602
Fax: (215) 898-1395
email: ratnaraj@wharton.upenn.edu