DOCUMENT RESUME

ED 392 337 HE 028 925

TITLE	Realizing the Potential of Information Resources: Information, Technology, and Services. Track 3: Serving Clients with Client/Server.
INSTITUTION	CAUSE, Boulder, Colo.
PUB DATE	96
NOTE	76p.; In: Realizing the Potential of Information Resources: Information, Technology, and Services. Proceedings of the CAUSE Annual Conference (New Orleans, Louisiana, November 28-December 3, 1995); see HE 028 922.
AVAILABLE FROM	CAUSE Exchange Library, 4840 Pearl East Circle, Suite 302E, Boulder, CO 80303 (individual papers available to CAUSE members at cost of reproduction).
PUB TYPE	Reports - Descriptive (141) -- Speeches/Conference Papers (150)
EDRS PRICE	MF01/PC04 Plus Postage.
DESCRIPTORS	Colleges; *Computer Networks; Computer Security; Computer System Design; Computer Uses in Education; Educational Change; *Higher Education; Information Management; Information Services; Information Systems; Information Technology; *Internet; *Program Implementation; Telecommunications; Universities
IDENTIFIERS	Arizona State University; Bradley University IL; Carnegie Mellon University PA; *CAUSE National Conference; *Client Server Computing Systems; Duke University NC; Ferris State University MI; Lehigh University PA; North Carolina State University; University of Delaware; University of Pennsylvania; World Wide Web

ABSTRACT

Eight papers are presented from the 1995 CAUSE conference track on client/server issues faced by managers of information technology at colleges and universities. The papers include: (1) "The Realities of Client/Server Development and Implementation" (Mary Ann Carr and Alan Hartwig), which examines Carnegie Mellon University's transition to client/server technology; (2) "Client/Server Architecture for Mainstream Administrative Systems--Bradley University's Experience" (Ellen Watson and Stephen Patrick); (3) "Internet Tools Access Administrative Data at the University of Delaware" (Carl Jacobson); (4) "A Successful Path to Client/Server Applications Through Rapid Application Development Methodology with a Self-Directed Work Team" (Sidney F. Holmes and Ellen L. Teague), which focuses on North Carolina State University; (5) "Data Warehousing Puts New Life into Legacy System" (Kathy L. Fisher), which discusses Ferris State University's data warehouse application; (6) "Assessing Risk: Developing a Client/Server Security Architecture" (Dave Millar and others), about the University of Pennsylvania's client/server network security efforts; (7) "Saving Enterprise Data from the Client/Server Grinch" (John D. Porter and others), on Arizona State University's efforts to keep control of its data sources; and (8) "Taking Advantage of WWW Technology" (Timothy Foley and others), on World Wide Web development at Lehigh and Duke Universities. Some papers contain references. (MDM)

TRACK 3
SERVING CLIENTS WITH CLIENT/SERVER

Coordinator: Susan J. Foster

Realizing the Potential of Information Resources:
Information, Technology, and Services

Proceedings of the 1995 CAUSE Annual Conference

The Realities of Client/Server Development and Implementation

Presented by

Mary Ann Carr
Director of Administrative Computing
Carnegie Mellon University
Pittsburgh, PA 15213
and
Alan Hartwig
Manager
Deloitte & Touche Consulting Group
Pittsburgh, PA 15222

for

The CAUSE 1995 National Conference
November 28 - December 1, 1995
New Orleans, LA

ABSTRACT

Carnegie Mellon University has made the initial transition to client/server technology. Most previous systems were developed using relational database, fourth generation language tools, C and Unix platforms with a character based user interface. The client/server model involves the use of the client desktop, a Netware server and a database server.

The paradigm shift to client/server should not be taken lightly. There are a number of issues that had to be addressed. These included which client/server tool(s) to use, which project should be the first client/server project, project planning and estimating, development standards and training for both development staff and end users. We also had to worry about end user desktop computing and network issues.

This paper will address these issues and reflect on the lessons learned throughout this process.

Introduction

Carnegie Mellon University is a private research university located in Pittsburgh, Pennsylvania. Carnegie Mellon enrolls approximately 7,000 students and has an annual budget of more than 300 million dollars.

The administrative computing environment at Carnegie Mellon is a mixed bag of hardware, operating systems and applications. These systems are supported by the Administrative Computing and Information Services(ACIS) department. ACIS consists of twenty-six professional and support staff. There are three primary hardware platforms supporting the administrative systems. These platforms are shown in the table below.

Hardware	Operating System	Primary Applications
Vax Cluster (6610,6430)	VAX VMS	Payroll, Alumni, General Ledger
Sequent S2000/490	Dynix PTX	Student Systems
Data General Aviion 9500	DG/UX	Financial Systems

The applications are a combination of commercial and Carnegie Mellon developed systems. The table below summarizes the key applications.

Application	Vendor/In-house
Student Systems	In-house
Financial Aid	In-house
Human Resources	In-house
Payroll	Cyborg Systems
Alumni/Development	Business Systems Resources
Financial Systems	In-house

The development tools used by Administrative Computing and Information Services have been primarily the Ingres fourth generation language (Application By Forms) and C with the Ingres relational database management system.

The primary desktop computer for users of administrative applications is the Apple Macintosh. The desktop environment is 80% Macintosh, 15% Windows PCs and 5% Unix workstations or terminals. The percentage of Windows machines has been growing in recent years.

The campus network is an ethernet backbone. The two primary network protocols being used are AppleTalk and Ethernet.

There are several Novell Netware servers being used for administrative applications. Most of these have been implemented in the last couple of years.

Client/Server Tool Choices

Administrative Computing and Information Services formed a team to evaluate alternative development environments. This team developed a set of criteria for a third party development tool. Several of these criteria were determined to be "drop dead" criteria. In other words, if a vendor did not support these criteria, their product would not be considered.

The drop dead criteria included supporting the Macintosh platform for both development and deployment, supporting the hardware and operating systems already in use, accessing a variety of relational databases, the ability to access more than one database simultaneously

and the ability to provide a character based interface without creating a separate set of code.

Indiana University had already gone through a very similar and very thorough evaluation of client/server development tools and shared the results of their work with Carnegie Mellon. This additional information provided some valuable insight into the strengths and weaknesses of a number of client/server development tools.

One of the primary goals in this process was to become independent of any particular database vendor. We wanted to be able to change database management systems without having to re-write applications. Another key goal was to be able to bring information stored in a variety of databases together on one screen. The tool selected needed to be able to access a variety of databases simultaneously. The final goal was to be able to offer a graphical user interface (GUI) front end.

Support for the Macintosh as a development and run-time platform became one of the most difficult criteria along with the ability to support a character based user interface with the same code. There were very few products available that met these criteria.

Uniface, now owned by Compuware, was identified by the team as the leading candidate for a client/server development tool. Uniface met these objectives by varying degrees of proficiency, but better than the competition overall. A Uniface evaluation (proof of concept) was performed by Administrative Computing and Information Services with support from Uniface. An evaluation plan was developed and Carnegie Mellon sent two developers to a week long Uniface training seminar.

The evaluation consisted of developing a simple Uniface application in the Carnegie Mellon University environment. Once the application was developed, it was ported to a variety of different platforms or databases to test these functions.

Uniface was chosen as the preferred client/server development tool upon the completion of the evaluation.

Project Choice

At the same time that Uniface was acquired, a relatively large module of the Human Resources Information System (HRIS) was under development. The majority of the detailed specifications had been completed by the project team. The project team was composed of key campus users, representatives from central administrative offices (payroll, benefits, student employment) and three people from Administrative Computing and Information Services. This project was replacing a paper based process for adding and updating employee and employment information. The project was very visible and had substantial end user involvement. There was concern about developing this important new application in a character based, database specific tool. A discussion was held with the project team regarding the pluses and minuses of switching to Uniface at this point in the project. The primary minus was that it would push back the schedule an estimated four to six months while Administrative Computing staff members became proficient with Uniface and built the infrastructure to support it. The primary plus was that the result would be a client/server application with a native graphical user interface that would be database independent. After weighing the pluses and minuses, the project team agreed to go ahead with the Uniface development. Thus, this application became the first significant Uniface (client/server) application developed at Carnegie Mellon University. This application is known as DRIVE (Distributed, Real-time, Interactive, Validated, Entry).

Major Issues

There were a number of major issues that needed to be addressed once the decision had been made to switch gears and use Uniface as the development environment for the DRIVE project. These issues included project planning and estimating, developer and end user training and infrastructure issues such as desktop computers and network connectivity.

Project Planning and Estimating

Preparing accurate project estimates and keeping a project plan on schedule can be a difficult task when you are very familiar with the tools being used. It is a much more difficult task when working with brand new technology. Because we had no experience with Uniface, it was difficult to estimate how long it would take to develop the application or even small pieces of the application.

There were two major facets to the project planning. The first facet was the planning for the introduction of Uniface. This included training ACIS staff, development of standards and environment configuration. This was the first major application for ACIS using Netware servers as part of the application delivery. Standards for access and security for the Netware servers to both Windows PCs and Macintoshes had to be developed.

In general, we knew that the learning curve for Uniface was rather steep and that it would take some time for the developers to become proficient with the Uniface tools. In addition to formal vendor training, Carnegie Mellon co-organized a local Uniface users group to share in on-site training and provide another layer of support. We also had access to Uniface technical support and user bulletin boards for additional information. The Uniface pre-sales force was well aware of the learning curve and provided assistance in our start-up.

The second facet was the planning for DRIVE which was well underway at this point. The schedule needed to be adjusted for the time necessary to introduce Uniface, plus the time needed to determine what changes needed to be made to the existing design.

General screen designs had been done before the switch to Uniface was made. The graphical interface made these screen designs basically obsolete. The screens were re-designed to take advantage of the graphical interface capabilities.

Uniface provided the ability to do rapid prototyping. We planned to use this ability extensively in the re-design work, as well as to make up time on screens that had not been fully specified before the tool switch.

Training and Development Standards

Obviously, with a new tool and environment there were a variety of training issues. We needed to train developers in the Uniface tools, to train systems support staff in Uniface technical and communications issues and to provide training for the end users on both the application and the Uniface environment.

The first developers assigned to the project were sent to a couple of vendor training sessions. The second group of developers were trained by the first group of developers. Some amount of self-study and trial and error was necessary in making the paradigm shift from the Ingres fourth generation language toolset to the Uniface environment. The training model became "on the job training" for the most part.

One of the issues that confronted the project team was the creation of standards. Since this was a new tool, there were no screen design, coding, deployment or documentation standards in place. Several people worked on various pieces of the standards document. The standards were modified as needed as the project progressed. For example, the Main Menu screen design had to be changed at the last minute due to a technical problem which could not be resolved.

There was a great deal of discussion on how to do end user training. The hope was that the graphical user interface would be relatively intuitive and not require a great deal of training. An inquiry only version of the primary screen was developed and released to interested users. Formal training was not provided but a "quick reference" card and basic documentation were available. The project team hoped this would significantly reduce the training requirement at the time of the production release of the system.

Infrastructure

Infrastructure turned out to be one of the biggest issues of the project. There were a variety of key infrastructure issues including desktop computers, network connections, deployment strategies, security, database access and remote access.

As mentioned earlier, the predominance of Macintosh computers on the desktop was a key factor in the choice of a client/server tool. Many of the Macintosh computers communicated with the campus network using the AppleTalk protocol. AppleTalk is a relatively slow network connection and the client/server model moves a lot of data over the network. A concerted effort was made to encourage end users to convert their AppleTalk connections to Ethernet. Administrative Computing and Information Services worked closely with the Data Communications department to provide timely and low cost conversions. A large number of users did convert their connections, but many still use AppleTalk.

Many of the Macintoshes are older models and do not have adequate resources to run client/server applications. Many of these can have additional memory installed which would help in some cases. The advent of the DRIVE application at the same time as the release of a sophisticated Excel spreadsheet for use in sponsored research budget development brought the issue of desktop upgrades and replacements into the forefront.

Administrative Computing recommends that departments should budget to replace 25% of their desktop computers every year. This, at first, seemed excessive to many users. The current pace of change in desktop hardware and software is so rapid that a machine over four years old is of limited value and it may be difficult to continue to upgrade it at a reasonable cost. When the desktop computer was being used primarily to do terminal emulation this was much less of an issue.

Performance in a client/server environment is much more difficult to monitor and improve. The problem can be the client machine, the network connection to the application server, the connection to the database server, the database system or a combination of all of the above. It is also difficult to simulate a large system load during testing due to the number of variables, especially network traffic, in a complex network environment such as Carnegie Mellon University and even more so in the summer months when student network traffic is at its minimum.

The client/server model that was used involved client software running on the desktop, application software on a Novell Netware server and the relational database management system on a database server. Originally, the database server was on a VAX/VMS machine, but the database was moved to a Data General Unix machine which had a lighter load in order to improve performance.

A key deployment issue was what to put on the local desktop machine and what to put on the Netware server. Application files could be stored in either location. The advantages to storing the application files on the server included not having to worry about whether

the user had the most current version running on their machine. On the other side, there was a performance advantage to having the application files on the local machine rather than accessing them over the network. The initial deployment strategy was to put the application files on the server due to the frequency of fixes and enhancements which would be taking place in the early production release of the system. This would make version control easier. A future version of Uniface promises to provide version control when the files are stored on the desktop machine.

Security was another key concern. Security exists at the file server level, database server level, application level and database level. Security at the Novell Netware server level was a minimal issue. No data or source code was stored on this server. In order to reduce the systems management load a single account and password was created on the Netware server for all users of the application. The secure logon occurred at the database server level.

The database can only be accessed by users who are defined as having access to that database. If the user has access to the database, they are given permission to access all data and the application controls which data they have access to. This again was done to minimize the amount of system maintenance work. There are approximately 200 defined users for the application. The user must have an account on the system that is serving as the database server. The user logs into this account at the beginning of running the application. This account connects to a restricted shell that prevents interactive logins.

The application has a fairly sophisticated access and approval module. The access model is basically hierarchical. Once the top level, executive managers, were defined to the system they defined the users who would have access to their data. Each succeeding level in the hierarchy performed the same process. Thus, the application restricts the employees that any user has access to. The application requires a separate 'save' password before any transactions can be saved or passed on to the next level in the process. This prevents someone from finding an unattended computer running the application and using it to change data.

One of the biggest security concerns turned out to be network security. The client/server model sends a great deal of information over the campus network. That information is not encrypted. Carnegie Mellon continues to explore options for securing the network which will provide the desired level of protection at a reasonable cost.

Ingres is the underlying relational database management system that was used for the DRIVE project. Carnegie Mellon entered into a site license agreement with Ingres in 1986. The site license agreement has ended, but Carnegie Mellon retains a substantial number of Ingres licenses across a variety of hardware and operating system platforms. Thus, there was no additional cost involved for a relational database system for the DRIVE project.

Lessons Learned

There have been a great number of lessons learned throughout this process. Probably the least surprising, but still important is that it will always take longer and it may cost much more than you think.

Client/server Tool

In general, Uniface served and continues to serve us well. This conclusion was initially clouded by serious issues with a Macintosh product which was in its infancy. Uniface provides the graphical user interface, database independence and multiple platform support that were desired. This was born out by the relatively painless move of the database server

from a DEC VAX system running VMS to a Data General system running DG/UX two weeks before the release date.

The Macintosh version of Uniface caused several problems. At the time of our tool evaluation, the Macintosh version of Uniface was in beta release and while we saw the demo, we did not have hands on experience. By the time DRIVE was selected as the project, Macintosh support was available. What we found was that the Macintosh platform was simply not as robust as the Windows version. Few Uniface sites using Macintoshes meant that we invested hours in debugging and designing work arounds with very little supportive services to rely on. We have subsequently developed an excellent, post-sales relationship with Compuware/Uniface. (As a note, do not expect the same level of expertise and support post-sales and you will not be disappointed.)

We did not plan adequately for the amount of time that would be spent in working with the vendor and getting answers from technical support. We found that it is well worth the effort to identify all the support services you can before you begin work with any new tool in general. Carnegie Mellon made considerable use of user groups and electronic bulletin boards for acquiring information and finding solutions. More often than not, other users had the answer, and the right answer, before technical support even returned the call. Given all the unique combinations of hardware, network, database and application requirements possible now, a nation-wide search is more likely to provide the depth of experience you need rather than one organization, even if it is the vendor.

Another area that did not get complete attention was the quality of documentation and training. Understanding that the possible number and complexity of client/server configurations is huge, we nevertheless expected to find some documentation depth in areas that pertained to our installation. After substantial trial and error with incomplete and inaccurate documentation we discovered that the only way to get the information we needed was through technical support. In many cases the vendor had additional documentation on specific topics and configurations. We quickly learned to be more proactive about seeking this information before wading through the volumes of outdated manuals. We also quickly dismissed vendor training as a cost-effective way to deal with certain topics. Certain training curricula by nature are more broad than others, standards and guidelines for example. As such, these sessions were worthless to us. Pre-read of training materials and references would have made the difference between time well spent and wasted time. While documentation and training were not high on the priority list, more attention to these topics at the start may have uncovered more efficient methods for building product knowledge.

Another issue that we did not adequately plan for was the possible need for a 3GL routine to perform functions that Uniface could not perform effectively. This was made more difficult by the cross platform nature of the application.

Initial Project Choice

Conventional wisdom and advice from the vendor indicated that the first Uniface project should be a relatively small application with limited scope. The timing for the DRIVE project would not allow for the project to be put on hold for an extended period of time while other smaller Uniface applications were developed. There was an overwhelming desire to not release the DRIVE application using the old, character based technology.

Our project plan called for developing test application scenarios in order to test all of the elements of a client/server application. Developing throw-away applications or test application scenarios in order to test all of the elements was a luxury that we did not have in reality. We achieved the same results by staging the release of a more complex production application.

Carnegie Mellon issued an optional, inquiry-only, pre-release of the DRIVE system. DRIVE Inquiry enabled end-users to become accustomed to the look and feel of the new GUI environment while testing network connectivity, desktop performance, software distribution strategies and printing capabilities. We discovered that many users were connected via low performance networks or had substandard desktop machines. During this pilot period, we were able to concentrate on providing solutions to the configuration problems without having to deal with a live, transaction oriented system. We should have more strongly encouraged the users to work with this application. It was a very low risk application and additional usage would have helped us identify and resolve some of the problems earlier in the project.

Unfortunately, not everyone took advantage of the early release, leaving a huge number of end-users with serious problems when we did go live. Users who made use of the early release also tended to be more proactive about upgrading and developed a more positive attitude about the application and the new computing paradigm. For those who did not, trying to sort out the system configuration, database performance, and network traffic issues from appropriate use of a complicated update system was nearly unmanageable. The experience we had in troubleshooting during the inquiry phase paid off as many of solutions and work arounds were already documented and in production, but the ill-effects of unsatisfied end-users lingers.

Project Planning and Estimating

Project Planning is one thing, plan execution is another. With client/server the complexity increases dramatically and the control decreases sharply. All the rules change. It was difficult enough for us just to identify all of the variables, let alone estimate for them. In the end, there were numerous iterations of the project plan. As mentioned above, we reneged on our original plan to release the application in total and opted instead to stage the release.

Rapid prototyping proved to be both a blessing and a curse. Because it was so easy to change screens, at least on the surface, there were many more iterations of key screens than had been planned for. Scope creep became a difficult management challenge. It was also difficult for the documentation and training committees to work with a product that was constantly changing. Furthermore, we found the learning curve prohibitive for rapid application development in our timeframe. If developers are not fluent in the tool, you will not get rapid application development. If the learning curve is steep, development will not be rapid in the short term.

We found the need to enlarge the scope of the development team. Implementing a successful client/server project required the commitment and teamwork of networking, desktop mainframe systems, and software development, and of course, end-user groups. It was extremely important to verify that each of these groups understood their place in the critical path of a project as soon as possible. As an organization, we were creating a new breed of project team. We believe that the organization as a whole benefited from the development of cross-functional expertise and by the involvement of end-users in the transitioning to client/server.

Training and Development Standards

After years of developing Ingres 4GL applications, the infrastructure of our development environment was very rich: shared libraries, screen templates, all manner of standards, reusable modules, and a development methodology. Experienced developers devised and conducted training sessions defraying training expenses. Very little of this investment translated to the client/server scenario. Two months of product evaluation and testing produced a rough standards document and a few screen templates. Weeks of off-site training were necessary just to develop an elementary understanding of the development tool. The lack of infrastructure contributed to the additional costs and additional time needed to

complete the project. A lack of experience in optimizing for performance cost precious hours in development and troubleshooting.

End User Computing and Networking

Carnegie Mellon has an extremely heterogeneous computing environment. Outside of the student computing clusters, there is no centralized control or published standard for networking or desktop computing. Lack of standards created the potential for each and every client to have a completely unique configuration. When problems occurred, what was once a question or two in the host-based scenario and a simple scan of a central log file, became a litany of inquiries, involving database administrators, network gurus, desktop support personnel and system managers.

In our business division alone, 24 different models of Macintosh computers were in use and 6 different PC processors from a wide range of IBM compatible vendors. Printers run the gamut. Fifteen months after initial campus exposure, we are still working on local printing on a case by case basis.

Factoring in support for multiple user interfaces, Macintosh, Windows and character based, added degrees of difficulty not only for system maintenance and troubleshooting, but also for upfront application design. We found that we began excluding GUI objects from screen design in order to accommodate the limitations of the character interface and thus were beginning to chip away at end-user benefits. In the end, Carnegie Mellon choose to abandon the character mode interface platform because of design barriers, additional training burden, and poor user acceptance.

While client/server boasts the benefit of utilizing the 'power of the desktop' we found that in may cases 'power' was nowhere to be found. Since our existing host-based systems required very little from the connecting 'terminal', many of our end-users had minimal desktop configurations. There are hard and soft costs for the discovery, evaluation, and installation of machine upgrades and replacements when multiplied across hundreds of users. At Carnegie Mellon, the decentralized nature of hardware procurement makes it difficult to calculate the total cost.

Network connectivity must also be evaluated as early as possible. ACIS actually developed a program in concert with the Data Communications department to assist end-users in the conversion of their network connections from apple-talk to ethernet at a reduced rate. Budgetary constraints necessitates early communication to the end-user community about upgrade paths and options. Secondarily, it might be useful to establish a fund for grants to those users who cannot afford the necessary upgrades.

Even though we were proactive in discussing the possible need for desktop hardware and network connection upgrades, this cost still came as a surprise to many departments. The cost of the desktop computing upgrades depended on the equipment in use and the nature of the upgrade required. The costs for the network upgrades were relatively low (< $200), but still caused a great deal of concern among a number of customers. We also used this as a forum to discuss planning and budgeting for normal desktop computing upgrades.

On the plus side, network and desktop upgrades, while imperative for acceptable performance of our client/server application, also increased productivity and performance of the other desktop applications that our end-users were already using.

Other Lessons

We found that we needed to clarify our goals and to refer to them often to stave off both criticism and frustration. Our primary goals were platform independence and a GUI front-end. This was accomplished.

Our developers became very frustrated. Client Server and Event Driven Programming was a paradigm shift for our 3GL and 4GL developers. At the beginning, middle, and even the end of the learning curve, developers felt that they could have programmed faster in their favorite host based, character mode 4GL. In addition, client/server placed an additional burden to have or acquire expertise in other areas like server configuration, network administration, and cross-platform development. Cross-function training was a critical step down the client/server path as responsibilities and requirements changed. Constant reminders of all 'wins' was an important part of winning the development team over to the new paradigm.

It was necessary to teach end-users what client/server entails beyond the ability to use a mouse. Client/server requires certain capabilities of the client machine and the underlying network. Early education and communication was imperative for enabling our users to acquire the minimal and optimal desktop computing configurations.

Conclusion

Administrative Computing and Information Services' first campus-wide client/server application is in production use. Many lessons have been learned along the way. The strong role of the user community was absolutely essential to the success of this project. Our development team has created a strong foundation for future projects. The second release of DRIVE is in development and should be ready for production in January.

Client/Server Architecture for Mainstream Administrative Systems - Bradley University's Experience

Ellen Watson, Associate Provost for Information Resources and Technology
Stephen Patrick, Director of Computing Services
Bradley University, Peoria, Illinois

Abstract:

Bradley University is approaching the end of our conversion of all mainframe administrative software to a client/server environment. These mission-critical systems support very large databases and provide multiple users and hardware platforms with a complex array of data entry, verification and reporting options. The following major systems have been/are being implemented in a client/server architecture:

- Advancement -- Summer 1993
- Finance -- Summer 1994
- Admissions -- Winter 1995-1996
- Student Records -- Spring 1996

The presentation will review our approach to client/server administrative applications and focus on our methods of working with system owners to design the client/server applications and to solve the problems and difficulties encountered. Particular attention will be devoted to the following issues:

- Involvement of system owners/users in client/server design and implementation
- System performance in a client/server environment
- Printing and reporting
- Integration with auxiliary applications -- DARS, SPEEDE, telephone registration
- Cross platform integration -- Mac, PC
- Remote access

The University Context

Bradley University is an independent, privately endowed, coeducational institution located in Peoria, Illinois. It was founded in 1897 as Bradley Polytechnic Institute by Lydia Moss Bradley as a memorial to her children and husband, Tobias. Bradley became a four-year college in 1920, and in 1946 became a university and began offering graduate programs. It is fully accredited.

Mainframe Administrative Systems at Bradley

In 1990, Bradley University's administrative systems were centralized on a mainframe computer. These systems were either locally developed, or were purchased software heavily modified by Bradley and no longer supported by the vendor. There were significant problems with these systems: The systems were not implemented in a modern data base management system; the systems were very inflexible; design decisions made in the 1970s were causing numerous operational problems. In addition, the mainframe computer was a proprietary design manufactured by Control Data Corporation. There was a concern that we would face a difficult problem if Control Data did not succeed in the mainframe market.

Scope of Bradley's Administrative Systems:

Advancement	
Alumni, Friends and Corporations	70,000
Size of Centennial Campaign	$100,000,000
Number of users	44
Remote staff	3
Finance	
Number of University Accounts (Account+Category)	60,000
Number of primary users	40
Admissions	
Student Search Prospects	200,000
Active Prospects	65,000
Applicants	3,500
Freshman target	1,050
Number of primary users	36
Number of secondary users	13
Remote staff	8
Student Records	
Number of enrolled students	6,000
Student Records on-line	36,000
Number of primary users	12
Concurrent secondary users (potentially all faculty)	30

Preparing for the Transition to a Client/Server Environment

After reviewing the existing environment and projected needs, Bradley University's Information Resources and Technology unit developed a "Computing Initiative" to move the campus from a mainframe environment to a distributed environment. Initially, we did not target client/server as the architecture of choice.

We recommended a phased implementation. The highest priority was the Advancement system. Bradley was to begin a $100 million campaign in 1993. The Advancement staff considered the then-existing system inadequate to support the needs of the campaign. Finance and accounting were given the second priority in part because we felt that the conversion of our system to a "standard" accounting system should be relatively straightforward. Student Records and Admissions were considered the most difficult applications to convert and they were put at the end.

We initially decided to look at purchased systems operating in a mini-computer environment. Technical staff and the major administrative computer users examined a variety of options. None of the systems available at the time provided all the services needed and the prices were too high for the perceived value of vendor-based systems.

A computer industry executive (and Bradley alumnus) recommended that we do a pilot test of the Advancement system in a client/server environment using Rapid Application Development methodology. Because this was uncharted territory for Bradley's administrative development staff, we employed a consultant to serve as the project leader for the effort.

June, 1992, the consultant and one Bradley programmer/analyst began a two month project to prototype an Advancement system based on the needs of our Advancement staff. At the completion of the pilot project, the overwhelming consensus was to complete the project and implement the system. We were able to demonstrate that performance of the system in a client/server environment was better than on the mainframe. Based upon the success of the pilot project, the implementation timetable for the Computing Initiative became:

1. Develop the Advancement system internally. Implement the system in June 1993.
2. Purchase a client/server-based Accounting System. Implement that system in June 1994.
3. Develop internally Student Records and Admissions systems. Implement in 1995.

The Computing Initiative was discussed with senior University Administrators throughout its development. The final budget agreed upon in 1993 was:

Advancement system cost	$45,000
Finance system cost	$95,000
Student Records system cost	$105,000
Admissions system cost	$48,000

Design Issues

We recognized from the beginning that "client" involvement -- the participation of the owners and users of the system -- would be essential to the success of system development efforts. The Rapid Application Development (RAD) methodology was integral to the success of our efforts. In prior development efforts, the time lag between system specification, coding and delivering a demonstrable p oduct was a hindrance to our understanding the needs of the client and delivering the product. Our RAD technique used weekly meetings with our clients to involve them in the development effort. Senior administrators, office managers and front line staff were all involved in the RAD implementation. At each meeting we demonstrated the week's progress, discussing issues and setting milestones for the next meeting. Clients saw the actual functioning of the system as we developed it. We were able to make major and minor adjustments to the system early and continuously in the development cycle.

A further complication was that none of us (including the client) had a clear understanding of what was data needed to support a $100 million campaign. We needed to insure that the system was flexible enough to allow for changing and emerging needs. By demonstrating the ease of program and data base modification, we all felt that we could adapt the system to meet whatever needs developed.

Experience with the Uncertainties of the Client/Server Environment

We knew that a client/server architecture would be fundamentally different from a mainframe or a mini-computer environment. When we began our project, we were not -- and could not be -- aware of the extent of the differences. Furthermore, we could not find successful implementations of client/server "bread and butter" administrative systems either to be purchased or to serve as models.

As we developed this new environment, we encountered a number of technical challenges that are worth discussion.

1. System performance in a client/server environment
2. Printing and reporting
3. Integration with auxiliary applications
4. Workstation configuration and cross platform integration
5. Remote access

System Performance

Client/Server architectures are much more complex than host-based systems. There are many factors that impact performance (generally negatively) including the server, the network and the client computer. Also, monitoring and control of these factors is much more difficult in a client/server environment. Several application development products provide acceptable performance with small data bases, or small groups of users, but cannot provide the performance needed for a modern, large administrative system with a large user base. Our Advancement database consisted of 70,000 individuals and institutions. Normalizing the data resulted in many data tables and millions of records. This environment makes the efficiency of

the application development tool and the data base a critical factor. For the Advancement system, we chose a DOS (non-Windows) environment using Clarion as the development tool. At the time, we did not find any Windows based environments that provided adequate performance. The Student Records and Admissions systems are developed in a Windows environment (with some reporting tasks in DOS) again using Clarion.

Printing and Reporting

Printing and reporting offer special challenges and opportunities in a client/server environment. Clients typically have a variety of printers that are incompatible with each other. Also, there exist needs for volume printing in either the user office or the computer center. At Bradley, different offices solved the problem in different manners. The Advancement unit elected to have Computing Services print most reports. The Controller's Office prints to a high volume laser printer in the office. We found that reporting using Windows standards adds significantly to the complexity of the programming task -- dealing with different sized fonts, proportional fonts, page layout vs. line layout. We chose to avoid this on the initial implementation by sticking with a DOS reporting strategy.

Executing report programs is another challenge. Most central host computers are multi-user/multi-tasking systems making background processing of reports a routine task. Desktop computing especially in a DOS environment are single tasking systems, and most users do not want to "give up" the use of their systems to run a report. Also, some reports need to be scheduled for later processing. On a positive note, integration with Word-processing and other office automation software is much easier for users to understand and use than in a centralized environment.

Because we developed the Advancement System internally, we were able to solve this problem to some extent. We have a pair of computers in Computing Services dedicated to running *ad hoc* reports. These computers constantly process reports placed in a report queue. Output from the report servers can be printed on any network printer or saved on a network disk for later processing. During busy times, users have the option of running these reports on their office computers. With the purchased accounting system, we did not have the ability to implement this system. Rather, we installed "extra" high performance computers in offices; clients can use these workstations to run reports without losing productivity. The Student Records and Admissions systems are Windows-based systems; we can multi-process reports and other tasks, though performance suffers.

Early benchmarks showed that the processing time needed for a client performing a massive batch run (for example, grade processing or census reporting) compared favorably to the same runs in the mainframe computer. Attention to the indexes and keys in the data-base has made selection and reporting significantly faster. We provide user-controlled parameters to specify the amount of resources to allocate to foreground processes. While this is just the opposite of what we would like, that's a Windows limitation that we must accommodate.

Auxiliary Applications

The Student Records system has a number of auxiliary subsystems that must be supported in a client/server environment.

- Bradley University has been using voice response registration for a number of years and this feature must be supported in the new system. The voice response system communicates to the mainframe through a single serial line. While we could write a PC program to simultaneously handle multiple conversations, we decided to do this with one UNIX system written in C++ communicating with the application server. This implies that the database management system must be able to support both personal computers and UNIX systems.

- The mainframe student records system audits degree requirements. Reprogramming a degree audit system was such a complex task that we felt it was more than the internal staff was prepared to accomplish in the planning time frame. We purchased the DARS system from Miami University to replace our degree audit system. DARS is an IBM mainframe, CICS-based, system available in COBOL or PL1. We initially thought we would build our own front end to DARS, then just execute the COBOL code to actually produce an audit. We found this another difficult challenge. We maintain degree requirements using a CICS emulator. We devoted our resources to the actual execution of an audit in a user friendly manner rather than devote resource to maintenance of the audit rules database.

- Bradley University was an early adopter of SPEEDE for exchange of electronic transcripts. We use the Supply Tech PC based package. Because we are eliminating a step in the process -- translation between the mainframe and the PC -- the resulting system is simpler than the original.

Configuration and Cross Platform Integration

Bradley University's computing environment is a diverse mixture of IBM-compatible desktop computers, Macintosh computers and UNIX workstations supported by a variety of servers and networks. Our goal is to provide access to our systems from a variety of platforms. We faced two major challenges: the configuration of individual computers is critical to our success; and we wanted to offer the systems on a variety of hardware platforms.

To minimize our development effort we purchased as much software as possible. Application software vendors (particularly in a DOS environment) have a tendency to demand a certain computer configuration. This does not present a problem for individuals who use only a single program, but presents a significant problem to those using a variety of software products. We solved this problem by making extensive use of boot menus.

Windows and other software vendors have a tendency to change the computer's configuration on installation of their software. Often this reduces available base memory to a level that will not support our applications.

Users occasionally change their computer's configuration without contacting us. We have had the entire Advancement system freeze-up because a user configured it

to run from a DOS Window -- and allocated no time to the process while minimized.

Many Windows-based applications are memory hogs, reducing available resources below an acceptable level.

We chose not to pursue getting our applications working on Windows95 until after we complete the initial implementation. Our applications do not run under Windows95.

Programmers need to monitor the impact of maintenance on the memory requirements of the destination computers, making testing and debugging a greater challenge.

We have found that DOS emulators will run our administrative software, but users pay a severe performance penalty. Running on Power Macintosh computers with an Intel hardware co-processor provides acceptable performance for a large cost premium. We now require either a 386 or better Intel processor to run our administrative applications.

Remote Access

Dialing in to campus using a modem was supported on one of our first mainframe computers. Running client/server applications remotely is not as easy a prospect. We have two types of remote users. One wishes to call in and access the University's administrative database. We installed several "remote control" computers that are on the Bradley network. Clients call in and take over these computers that are on our local network. This process is conceptually the same to the user as a terminal is to the mainframe.

Admissions Field Representatives run the same program in their remote office (typically on a portable computer) that the home office staff uses. We have a batch program to upload changes to the main database and download new prospects and changed information from the master database to the admissions representative. This has several advantages over a separate and special program for the Field Representatives, including easier training, consistent support, and flexibility so that our staff can move between the field and the home office easily. The software's security features controls the ability of staff to change only authorized data. Security is consistent in the field and at home.

For both types of users, we are experimenting with bridging/routing Ethernet over ISDN telephone lines. We have a project underway to determine if we can provide true client/server applications to our remote clients. For the Admissions field representatives, we are experimenting with a commercial network. We found it difficult to support remote users' modem and communications problems from Peoria.

Benefits

There are two main categories of benefits we have realized from the new systems. The most significant set of benefits result from a good design, using relation database modeling techniques to build systems that are flexible and provide clients with control of their system. While the same system (with a different look and feel) could have been built in a mini-mainframe environment, the benefits associated with the client/server architecture also substantial. These benefits are primarily financial savings and ease of use.

Financial Benefits

The full cost of a minicomputer vs. a client/server system will be about the same IF the minicomputer system consists of a minicomputer and low cost terminals on desktops. If we consider workstations on desktops, there is a clear cost advantage with client/server architectures because the cost of a server is significantly less than that of a mini-computer that would provide equivalent performance.

Even assuming an equivalent cost structure, there are financial benefits with client/server architectures. At Bradley there has been a several year lag between the time the mainframe begins to provide unacceptable performance and the time we upgrade the mainframe, due to the high cost of upgrading the mainframe. Because in a client/server environment the bulk of the computing takes place on the desktop, the incremental cost of improving performance for a particular user is far lower than in a central host-based system. With smaller incremental cost/performance amounts, we can choose a strategy of continuous upgrading rather than waiting for performance to become unacceptable before upgrading the host computer.

We found with our mainframe computer that most computer users were migrating to personal computers to emulate mainframe terminals. Clients preferred to use personal productivity tools on desktop computers rather than on the mainframe. Also, we were installing local area networks to provide shared resources, other network applications and access to the Internet. In this case, we found a significant cost advantage by trading a single very expensive mainframe computer for several low-cost servers. We can purchase high-end servers with all the bells and whistles (RAID, UPS, and built in tape drives) for the mainframe hardware maintenance money and have change left.

Ease of Use

We can provide far more services in a PC environment than in a block mode mainframe terminal. In our DOS applications, we edit and correct each field as it is entered. Pop-up screens with list and scroll boxes are very easy to develop and provide significant help to our clients. We use normal Windows conventions for our Windows products. Once an individual is familiar with Windows applications, the training needed to run our applications is minimal. This is particularly important for casual users such as faculty and advisers.

Frequent users of the systems (particularly data entry clerks) can be slowed down by a poorly written "standard Windows" interface. All of our systems have been written to not require mouse input for data entry screens. On the other hand, the

mouse is extremely useful for the point-and-click users (management) of the system.

Because the applications follow Windows rules, we can open multiple administrative applications and multiple instances of the same applications. When a client is in the middle of one task and needs to respond to an inquiry, that individual can open other window, handle the work, then return to the previous task.

Conclusion

The Computing Initiative which began in 1992 has completely transformed the environment for administrative computing at the University. Here is a summary of where we are now on each of those components of the Computing Initiative:

- Advancement -- Implemented on-time -- Summer 1993
- Finance -- Implemented on-time -- Summer 1994
- Admissions -- Implemented December 1995
- Student Records -- Planned phased implementation beginning April 1996

If we were to undertake the project at this point in time, we would do some things the same way, and certainly make some changes, as well. The decision to move to a distributed environment was definitely correct, as was the decision to implement a client/server architecture. Based on our experience, we strongly recommend that client/server applications be given careful consideration at any institution considering advancing from a mainframe environment. We have experienced genuine improvements in both cost effectiveness and user satisfaction through implementing the client/server applications. The Rapid Application Development methodology was also highly successful, although as we gained more experience we made some modifications in the original methodology to take advantage of our increased expertise and the differing desires of clients to be more or less deeply involved in the development process.

We have learned -- through sometimes painful experience -- that it is worth the extra money to buy tested, network-certified servers, with maximum capacity and appropriate, fully-compatible components and peripherals. Our experience with systems assembled from a variety of manufacturers has been that it has taken a great deal of time and expertise to assemble, tune and later enhance the systems, causing downtime and problems for users.

Our original intent to purchase developed software from reputable vendors would still be our first choice now -- and there are now a number of choices available that are worth consideration, which was not the case when we begin implementation. However, we have found that we can provide far better service to our users if we have access to source code, so that we can appropriately debug, problem-solve, and modify applications to meet the specific needs of our institution. We will certainly be considering vendor-developed and -supported software as we add applications to our client/server environment.

Now that the Computing Initiative is nearing completion, it would be nice to think that we could sit back and rest on our laurels for a few months -- or at least polish up the final modifications, finish debugging, etc. In the real world, however, that is not an option. As soon as we have programmers free, we need to begin the cycle of development again, porting the Advancement System to a Windows environment, upgrading servers and networking for the whole architecture, resolving some nagging problems, and adding "essential" enhancements to meet user needs. In addition, we need to fully integrate all of the applications, develop an executive information system, develop a data warehouse structure to support institutional research ... well, you get the picture.

It has been very important that we have had -- and continue to receive -- the support of senior administration and our users, as we push to provide the best possible systems and service to meet the University's needs. Those are the essential realities that underlie the decisions that we have made. Based on our experience, client/server architecture can be an important element in reaching those goals.

Internet Tools Access Administrative Data at the University of Delaware

by Carl Jacobson

The introduction of NCSA's Mosaic browser ignited a fire of interest that is changing the face of the Internet and the way we deal with networked information. The scramble for commercial success on the Internet has brought many technology vendors into the Web trade, resulting in the development of new tools and methods. As these advances define the role of commerce on the Internet, they will also change the way we conduct routine business on our networked campuses.

Internet Tools Access Administrative Data at the University of Delaware

by Carl Jacobson

Director of Management Information Services
University of Delaware
carl.jacobson@mvs.udel.edu

The World Wide Web offers a new model for application development in colleges and universities. At the University of Delaware, for example, Web tools are effectively being employed to produce multi-platform administrative applications. Web applications are quickly and easily crafted to interact with administrative databases, providing powerful, new functionality. Web applications cross most client platforms and can be simultaneously GUI- and character-based, reaching users of both old and new desktop hardware. Web tools are particularly suited to customer outreach efforts, delivering direct service to students, faculty, and staff. The capabilities of the Web's HyperText Markup Language (HTML) facilitate new classes of applications, including hyper-reporting, mixed media, electronic forms, and kiosk services.

Administrative systems and customer service

The University of Delaware provides widespread access to its administrative systems, delivering improved customer service to students, faculty, and staff. The Internet's free, public, outreach tools (World Wide Web, Gopher, and e-mail) have been merged with the institution's closed, proprietary administrative systems (student records, human resources, and financial management).

Private, personal information, including student and employee records, is integrated with the public, general information of the campuswide information system. Freely distributed clients for DOS, Windows, Mac, UNIX, and timeshare users allow access to official, production data on both MVS and UNIX platforms. The methods employed to achieve this success are simple, inexpensive, and easily adapted.

While the administrative systems of the University can be characterized as closed, proprietary, controlled, and secure, the student view of computing is open, pedestrian, public, and wide-reaching. In keeping pace with trends toward a more student-centered campus, Delaware's administrative systems have been reworked to place an emphasis on self-service. Self-service technologies have been applied to deliver timely information directly to the customer. These technologies empower the customer and provide cost-effective, automated services that know no geographic boundaries. Self-service technologies include interactive voice response dialogs, kiosk systems, debit-card transactions, and World Wide Web applications.

Technologies merge

With a healthy portfolio of mainframe-based administrative systems, Delaware chose to adapt existing information resources to open, network technologies, in order to meet the goals of improved customer service.

It is impossible to grant the large, expanding customer base direct access to mainframe-based information systems. Faculty and research users of "academic" machines have little desire to log on to "administrative" machines and navigate through unfamiliar territory in search of needed information. Nor is it feasible to allow 22,000 students to log on to the administrative mainframe to review grades on the day they are posted. These closed, proprietary systems must be opened to allow such "pedestrian" use. Administrative

information services must be adapted to behave more along the lines of a publicly available campuswide information system (CWIS).

To meet these goals, Delaware chose to leverage existing resources by merging
- *the established, closed, proprietary mainframe-based administrative systems with*
- *the emerging, open, public, client/server- based campuswide information systems*

in order to
- *deliver customer service in the environment of the customer,*
- *do "administrative things" in "the student way,"*
- *allow the free, public access tools of the Internet to be used to do official university business.*

The key to successfully merging these technologies is compromise. It is necessary to bring the security of the administrative environment to Internet tools, while opening the administrative systems to Internet protocols.

As Delaware first turned to the Web for administrative support, official institutional data were maintained using Software AG's ADABAS database system and processed by programs written in COBOL and NATURAL. At the same time, CWIS information was collected, maintained, and delivered on the World Wide Web. The use of Web browsers was widespread among campus customers, while existing Natural/ADABAS systems were robust and useful. These disparate resources were combined in a unique but simple way to deliver improved information service to students, staff, and faculty.

This combination requires the transformation of the "host" of a host-terminal system into the "server" of a client/server system. The host and its associated applications become part of a client/server network enabling outreach and supporting diverse data types.

Opening closed systems

The opening of such closed systems focuses on the need for secure servers to translate Internet protocols into the languages of the administrative systems. Web HTTP (HyperText Transport Protocol) servers meet this need, functioning as effective gateways between the Web browsers and administrative programs and databases.

Such Web gateway servers may be built or bought. Several HTTP servers are available commercially at surprisingly low cost. Apple's Internet Server and Netscape's Commerce Server are examples of general-purpose HTTP servers that provide packaged sets of tools needed to develop Web applications. They are popular, inexpensive, vendor supported, and utilize economical hardware.

While commercial gateway servers provide the convenience of packaged toolsets, they may require additional hardware, new communications protocols, and unfamiliar programming languages. As an alternative, special-purpose HTTP servers can be developed in-house to perform these translations directly on existing hosts. Interpretive servers may be written on any networked platform, using any language supporting Internet communications interfaces.

This approach would, for example, allow COBOL programmers to open legacy systems to the Web using the tools, techniques, and training of the legacy environment. While Web browsers expect information to be packaged using HTTP, they are not concerned with how that packaging is performed.

Whether built or bought, gateway servers use standard HTTP to communicate with Web browsers on the user side. On the application side, these servers employ common gateway interfaces (CGIs) to communicate with external programs and databases. CGIs are programs or scripts, and may be written in many languages, including C, Perl, and AppleScript. CGIs allow Web servers to communicate with other servers, DBMSs, external programs, screen scrapers, and a variety of network program interfaces.

CGIs may be used in conjunction with DBMSs and programming languages to build complete, new administrative applications, or CGIs may play the role of transforming closed, proprietary administrative systems into compelling Web applications.

With many Delaware administrative systems residing on an MVS mainframe, interpretive servers were developed to run in this environment, accept Internet packets, recognize Web HTTP protocol, and call administrative application programs based on the content of these packets.

With interpretive servers speaking to administrative programs, existing tasks such as transcript production can be reused rather than re-developed. Upon request from a student client, the server simply invokes the existing COBOL transcript program. However, instead of printing or displaying the results, they are packaged in a Web packet and sent out onto the network.

Authentication, authorization, and encryption

In order to provide the levels of security needed in conducting personal business, authentication, authorization, and encryption routines must be employed.

With an overall design goal of "using existing resources whenever possible," security schemes used for touch-tone registration were enlisted at Delaware to provide similar protection to the Internet clients. Student-ID and PIN (Personal Identification Number) authentication was already known and in use by students and staff. PIN-based authorization tables were already in place in existing administrative systems. (Exhibit I illustrates the Web SID/PIN authentication and access to records.)

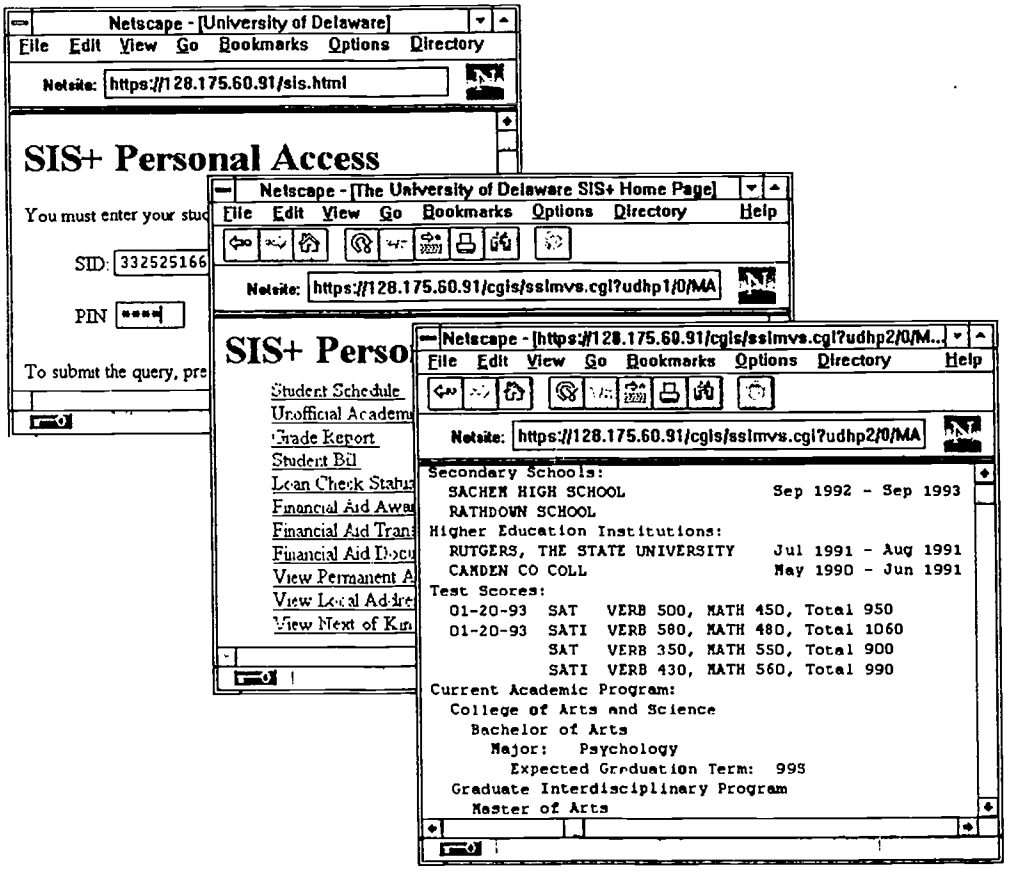


Exhibit I: *Using Secure Web Browsers, students first provide SID and PIN numbers to gain access to a menu of services. A simple point-and-click on "Unofficial Academic Transcript" produces, in real time, a complete student transcript. All data displayed on these screens is protected when crossing the network using public key encription.*

In order to protect the authentication information as well as the private records of students, faculty, and staff, Netscape's Secure Socket Layer (SSL) encryption protocol was adopted. This protocol was selected because of the popularity and success of the Netscape's Web browser and because its socket-level encryption is ideal for supporting the re-use of existing authentication and authorization schemes.

SSL uses encryption to enhance user privacy by providing a communications channel that is secure against eavesdropping. When an SSL-aware browser connects to an SSL-secured server, all information passing between browser and server is fully encrypted. This secure data circuit allows existing authentication and authorization information to be safely exchanged on the network.

SSL is not the only security alternative available to those wanting to do business on the Web. Secure HTTP, Digest Access Authentication, Shen, and DCE-Web security are several examples of current Web security efforts.

Stateless client/server relationships

A significant advantage to adopting the Web-server model to provide student services lies in the "statelessness" of these servers. The transactions may be viewed as "stateless" in that a server has no lasting connection with each requesting client. The server "comes alive" upon receiving a request message, interprets and fulfills the request by passing a message back across the network, and returns to a "wait state" until the next user request comes along.

Since students do not log on to the administrative system, there is no data communications overhead. A single task monitors an Internet port and responds to customer requests. This "stateless" client/server relationship allows many customers to effectively use administrative resources without becoming members of that environment.

Without the overhead of CICS or TSO sessions, a mainframe server performs its simple tasks with little impact on the overall system. Response is immediate, even for longer packages such as student transcripts. In addition, due to the nature of the Web itself, the response time expectations of Web users are lower than those of interactive, transaction-based systems, so that if a delay is encountered, it is unremarkable.

Such interpretive servers have the advantage of accessing production data directly. They need not rely on data extracts, but instead return timely and accurate information from official production records. As students perform touch-tone drop-add, they can immediately confirm schedule changes. As students pay bills, they can quickly print summaries of charges and payments. With many business transactions reaching databases in real time, it has become necessary to report these changes in real time. "Just-in-time" production of course schedules and transcripts calls for this level of timeliness. The stateless Web server allows this to be accomplished easily and inexpensively.

At Delaware, servers have been deployed to run on MVS, UNIX, and MacOS platforms to allow information to be gleaned from various databases across campus and to take advantage of the relative merits of each operating system.

Training and support

With Web browsers already in the hands of students, faculty, and staff, the issues of training, support, and software distribution are minimized. Student grades and transcripts may be accessed in a manner familiar to all existing Web users, allowing students to use these tools to conduct institutional business (see Exhibit I), as well as to explore academic frontiers.

Client-side development costs are usually a large portion of a client/server budget. However, Web applications differ from the popular client/server model in that all Web development effort is on the server side. Since Web client tools are free and widespread, client-side costs have been kept to a minimum.

Server-side development may be as simple as re-routing the formatted-text output of a COBOL report program to a routine to place the output in an HTTP packet. In many cases, there is no need to add HTML codes to a formatted text document and no need for application programmers to learn the details of HTML.

However, HTML syntax is easy to learn and enables application developers to transform simple, pre-formatted text reports into powerful hypertext documents supporting multimedia and user input.

Software distribution

One strength of the client/server model of computing is the increased functionality provided at the desktop. Not only can Internet browsers access grades and course schedules, but they can also retrieve and display images, sounds, and even brief video clips. Any "digital object" of reasonable size can be delivered to any client workstation. This includes the delivery of client software itself.

In keeping with the goal of "self-service," Delaware's Internet client software is stored on a Web server and made available to anyone in the campus community across the network. A simple point-and-click causes the newest version of a program to be loaded across the network to the user's hard drive.

For Web applications themselves, the bulk of processing code remains on the "server side," and version-control is centralized. HTTP mark-ups are, in effect, software code that is delivered and interpreted in real time, ensuring that the most recent code changes are invoked by every user.

The Web's hypertext capabilities provide for easy access to associated documentation for all network-delivered software.

Classes of application

The powerful capabilities of the Web enable the rapid develop of new classes of administrative applications. While formatted text reports such as course schedules and transcripts can easily be delivered to Web browsers, the hyper-linking and multimedia features of the Web offer exciting new potential. The Web's hypermedia model expands the potential of administrative computing.

Hyper-reporting

An HTML document may be linked to any other document on the Web, creating a powerful hypertext application that may be used to produce hyper-reports. Hyper-reporting can be used to link existing summary reports and detail screens to produce effective executive information systems. Institutional executives may receive regularly generated summary reports with built-in "drill-down" capability, with links to official, detailed, production data from administrative databases.

Mixed media

Web hyperlinking also supports diverse data types, such as photographic or document images. Student demographic data may be gleaned from a legacy student information system, while student photographs are retrieved from a UNIX-based image server. Both could be merged seamlessly by the desktop Web browser.

Electronic forms

Web browsers support fill-in-the-blank forms with ease-of-use features such as scroll boxes and radio buttons. Paper forms used for routine campus business may be effectively replaced by electronic documents, available to users on all platforms and routed and processed on the campus network. (See Exhibit II, an illustration of the University's forms home page.)

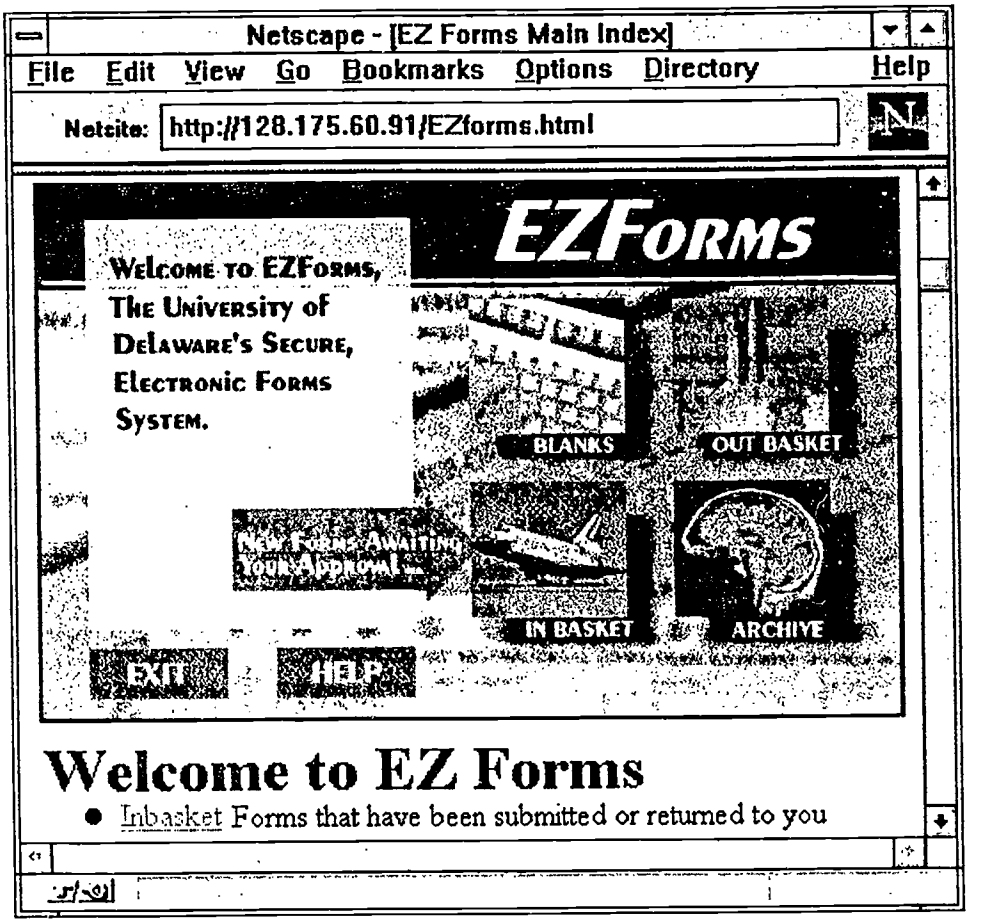


Exhibit II: *Groupware applications requiring collaboration, routing, and approval are ideal candidates for the Web.*

Touch-screen, multimedia kiosks

PODIUM,[1] a multimedia authoring tool developed at the University of Delaware, has been made "Internet aware," allowing it to "speak" Gopher and Web protocols. This tool, originally designed as a classroom technology, is now used by several institutions to develop compelling multimedia, touch-screen kiosks—merging image, sound, and video with administrative information. PODIUM is an early example of an emerging class of tool, facilitating the construction of special-purpose browsers for custom Web applications.

Evolving Web capabilities

One of the primary strengths of the Web is the ability to deal with diverse data types—the ability to support multimedia objects. Complex data objects may be sent across the network and "unwrapped" and "displayed" at desktop browsers. In the future, these objects will become even more complex. For example, an electronic form and its associated processing rules might be delivered directly to, and processed locally on, the client workstation.

Vendor efforts, such as Sun Microsystems' Hot Java, demonstrate the ability to deliver secure program code as an integral part of a Web transaction. This capability will redefine distributed computing, allowing host servers to deliver machine-independent code to desktop clients for just-in-time processing.

Conclusion

Rapid advances in the development of tools for the Internet will impact the processes of teaching, learning, and research at our institutions. Many of these same advances will contribute to the way we conduct business and affect daily campus life for students, employees, and visitors.

The World Wide Web is emerging as a new model for administrative service on our campuses. With the application of emerging tools and technologies, existing resources can be re-used effectively to return immediate benefits for small investments. Each early adopter of these technologies will gain valuable experience and insight into the issues of delivering networked services and will establish a foundation for controlled growth and change.

Development Checklist

As the Web capabilities listed below demonstrate, Web development offers many advantages over traditional application development methods.

✔ *Multi-platform*
Web clients exist for DOS, Windows, Mac, UNIX, and other popular operating systems.

✔ *Low cost*
Commercial Web browsers are available to educational institutions at no cost.

✔ *GUI*
Web applications may be simultaneously GUI- and character-based, delivering functionality to users of older desktop hardware.

✔ *Mixed media*
Web protocols support images, sounds, and video clips as well as text, allowing character-based administrative data to be merged with these rich data types.

✔ *Common user interface*
Although Web browsers run on disparate platforms, a certain look and feel is maintained across platforms, providing an easy-to-support common user interface.

✔ *Software distribution*
Web browsers themselves may be easily and inexpensively distributed across the network, using the Web itself.

✔ *Self-documenting*
Hypertext capabilities allow application help and tutorial routines to become an integral part of any Web application.

✔ *Distributed servers*
Web browsers merge information from several servers onto a single screen, without specific user knowledge of these servers.

✔ *Network security*
Socket-level encryption provides a secure network communications channel that can be employed to protect any existing or emerging campus authentication scheme in addition to all user data.

✔ *Local processing*
As Web browsers employ "helper applications" to display and process information, Web applications can therefore make use of local processes, such as spreadsheet or word processing programs.

Additional Resources

Visit Delaware's administrative Web site (http://www.mis.udel.edu/admin.html) for live demonstrations of secure business transactions. A Web version of this paper with hyperlinks to demonstrations and other relevant resources can be found at the URL http://cause-www.colorado.edu/cause-effect/cem95/cem9533.html

"The Web: A New Model for Application Development," a CAUSE95 pre-conference seminar to be held November 28 in New Orleans, will provide an opportunity to explore, in depth, the use of the Web for application development. Seminar leader Carl Jacobson will target a general audience, addressing technical issues of interest to programmers and DBAs in the non-technical language of managers and directors. For more information about CAUSE95 activities, visit the CAUSE Web server (http://cause-www.colorado.edu/) or Gopher server (gopher://cause-gopher.colorado.edu/) or call 303-939-0315.

FOOTNOTE:

[1] PODIUM is an object-oriented multimedia application generator developed by Professor Fred Hofstetter. For more information, see http://www.udel.edu/lynam/fth/podium.html.

A Successful Path to Client/Server Applications Through Rapid Application Development Methodology With a Self-Directed Work Team

Presented at CAUSE'95
November 28 - December 1, 1995

Sidney F. Holmes
Applications Development Manager
sid_holmes@ncsu.edu

Ellen L. Teague
Applications Development Project Supervisor
ellen_teague@ncsu.edu

Administrative Computing Services
North Carolina State University
Box 7209
Raleigh, NC 27695
Phone: (919) 515-2794
Fax: (919) 515-1958

Abstract

In providing optimal service to customers, Information Technology departments are faced with many challenges. One of the biggest is making the transition to a distributed applications environment with minimum impact on current structure and plans. With support from key areas, North Carolina State University has taken an innovative approach to moving into the Client/Server arena using Rapid Application Development methodology and a dedicated work team. We were able to build and maintain momentum installing Client/Server applications, while continuing to maintain legacy systems and respond to existing customer requests. This paper will demonstrate a proven, successful path to the understanding and implementation of developing technology.

Introduction

One of the greatest challenges facing IT professionals today is how to implement Client/Server applications and maintain momentum in the Client/Server arena without negatively impacting the application development staff's ability to stay on top of an overwhelming backlog of work. Failure to take advantage of new technology is an unacceptable alternative.

With the support of the University Administration and a few key customers, Administrative Computing Services at North Carolina State University was able to take an innovative approach to solving the problem. By creating a new application development team, operating as a self-directed work team, and employing major components of Rapid Application Development (RAD) methodology, we have quickly and successfully moved into the Client/Server arena. We also have been able to maintain momentum in this environment and to explore and implement new technology as it becomes available.

Our work team completed one cycle which resulted in the production installation of legacy data extract databases available to Campus for ad-hoc use and three client/server applications serving a varying number of customers. In addition, we charged a Continuous Quality Improvement (CQI) team to explore the implementation of Object Oriented technology and methods. We have now begun the second cycle with the new development team. The new charge shifts focus from medium/ long-term projects to short-term projects with a high impact on improving processing efficiency within Campus departments. This shift in focus was, in part, brought about by budget cuts and staff shortages. Long range plans will include the development of Executive Information Systems using Object Oriented technology and methods. We are committed to continued advancement.

Developing the Plan

North Carolina State University had a backlog of requests from the customer community and a small IS staff to respond to those requests. Our customers were becoming used to working in a Windows environment and expected their new applications to use GUI standards and "leading-edge" tools. Our customers demanded easier, less restrictive access to their data. Faced with the restraints of the existing backlog, the maintenance of legacy systems, and previously committed resources, we sought innovation solutions. To address these changes, we needed to assess the feasibility of changing the way we have traditionally operated without losing sight of the need to stay responsive in the existing environment.

Based on customer demands for high-quality applications, more user-friendly interfaces, and quicker turnaround, as well as a desire to follow new industry trends, we determined that Client/Server applications were the answer to our needs. A pilot team composed of application developers and systems support personnel was formed to evaluate hardware and software needed to operate in the Client/Server environment. Over the period of a year, the pilot team, working with a small group of friendly, progressive customers, developed several small applications using a variety of tools. At the end of this evaluation period, the decision was made to use Sybase as a database engine and PowerBuilder as the primary development tool. We began researching alternatives to allow us to maintain the status quo while we moved forward into the Client/Server arena. The plan was to:

* Identify the developers to do the work
* Choose an efficient development methodology
* Expand the existing toolset
* Identify projects that were candidates for this environment
* Sell the plan to the customers

Implementing The Plan

We soon recognized that we could not respond as quickly as necessary working within the framework of the existing application development teams. A result of the work done by the pilot team was the implementation of three Client/Server applications in three different customer areas. These applications were used to sell University Administrators and Campus customers on the benefits of moving into the Client/Server arena. Management became convinced Client/Server development was the direction we needed to go and committed to support the effort. Five new positions were created and assigned to the existing application development units. The new positions allowed us to establish a new applications development team. The team, called the Distributed Applications Resource Team (DART), developed applications and provided other customer resources using emerging technology. Each of the existing development units assigned one staff member to the DART team. The team members were rotated annually, providing a means for all staff members to participate and be trained in the new environment. The first group included one senior project leader, two senior programmer analysts, and two junior programmer analysts.

As we read about and discussed the experiences of other institutions and organizations and evaluated development methodologies, we concluded Rapid Application Development Methodology (RAD) had evolved into the methodology of choice for Client/Server development. RAD offered the flexibility needed to develop applications of varied size and complexity and was conducive to mixing with more traditional methods. RAD also promoted increased customer involvement in the process and demonstrated to the customer, first hand, the commitments they would need to make to move successfully into the new environment.

Our internal management decided that the DART team would operate as a self-directed work team. The team was staffed with developers with varied skill levels and experience who were participating as volunteers. By functioning without a supervisor, all DART team members would share equally in the success or failure of the project. It was necessary for team members to evaluate new concepts and work toward meeting team, rather than individual goals. Main criteria for choosing projects for the team were:

* The task could be completed in a relatively short time
* Would have high visibility when implemented
* Would alleviate some of the stress on the existing development units.

Executing The Plan

All members of the development units within the organization had received training in Sybase, SQL, and PowerBuilder. All members of the initial DART team had participated on one of the pilot teams and had some experience in relational databases and Client/Server development. To allow the DART team to become comfortable working in a new environment, the team's first assignment was to continue developing the legacy system extract databases that were begun by the pilot team. This allowed the DART team to get to know each other and to acclimate themselves to the self-directed team concept. This process also assisted us in meeting the goals of the organization by providing easy access to legacy data for the customers which, in turn, alleviated some of the workload from the existing development units. Once extract databases were established for financial, purchasing, student, and vendor data and the Campus was provided with a user-friendly inquiry tool, the customers were no longer dependent on IS support to access their data. This reduced the number of requests to the existing development teams. The extract databases were also highly visible in most all areas of the campus community.

A division-wide application to track staff training was selected as the first Client/Server software package to be developed by the DART team using RAD methodology. The team began the process working with requirements from the primary customer. From the initial requirements document, we laid out the first draft of the database design and developed a high-level prototype for the main functions of the application. From this point through production implementation of the application, the package was developed using RAD methodology. Joint Application Development (JAD) sessions were held to allow the developers to work through the prototype and to interact with customer representatives. JAD sessions allowed developers to assess the customer's reaction to application elements and to change the prototype, as needed. Important knowledge about their preferences for the look and feel of the model was gained. The customers were able to see what the application was going to look like and how it would work before a single line of code was written.

After the first two JAD sessions, components of the package were assigned within the DART team and coding began. As programming of the major functions of each module was completed, the module was made available to the customer for testing. The customers and developers worked in a mode of iterative project development, one module at a time, and each module was installed into production following completion of testing by the customer. The application had been broken into logical pieces and each piece was delivered upon completion. This process let us incrementally install the Application Modules and let the customer work with sections of the package as they came available. A key benefit of the process was that the customer could initially populate the database through the application which provided part of the training necessary for their staff. The end results of following this process were:

* The customer began using the application much earlier in the development process than was normal with traditional methodologies
* The customer was actively involved in the development throughout the entire process
* The deliverables required much less change than normal
* The application was what the customer needed and wanted
* Delivery was made within a shorter time frame than we felt would have been the case using a traditional approach

This initial experience with RAD methodology reinforced our decision that RAD was appropriate for developing Client/Server applications. The process took longer than we had expected, but we felt the quality of the resulting product was significantly better than using traditional methods and that time was saved by not having to revisit code and enhance the product to what the customer really wanted.

Iterative prototyping was very helpful in giving the team a true feel for what the customer liked and for identifying needs and expectations. We felt that the process provided the customer with a "feel" for the new technology, and it exposed them to new terminology. The process enabled us to make changes easily, encouraged brainstorming, and uncovered elements of the application that might have been missed. The customer was much more involved in the design process, reactions to the application were easier to gauge, and pitfalls of the design were exposed very early in the process.

The JAD sessions were invaluable. They required a commitment from the customer to make available the resources necessary for the project to be successful. Consequently the DART team did not have to wait for customer availability to move forward with the project. To use JAD successfully, the right people must be involved. Decision makers, as well as people who understand and use the process the application is accommodating were required. We found the sessions to be much more successful when attainable goals were set before each one. Our experience showed it to be important for the developers to know the requirements and have a prototype before the first session. It was important that roles be assigned to each developer for each session so that everything that occurred was documented and nothing was missed. We learned to listen to everything that was said by each person involved.

Iterative project development and incremental installation of the application provided many plusses to the process. One of the biggest benefits was that this technique allowed for early discovery of problems. The DART team had the luxury of being able to deal with problems at a more leisurely pace than if many problems surfaced at once during system testing of an entire application. We were able to make logical adjustments to the design instead of applying stop-gap measures. This development method

kept the customer actively involved throughout the process and provided working modules of the application as the project progressed. This involvement boosted staff morale within the customer environment and assisted in gaining customer confidence early in the process. It helped to eliminate surprises at final installation, provided a built-in method to expose the customer to the client/server environment, and resulted in a higher quality application.

The increased involvement of the customer in the development process when using RAD methodology was also valuable from the developer's standpoint. Even reluctant customers were led into a hands-on involvement in the project from start to finish. Iterative development techniques provide them with a venue for discovering design problems, omissions, and mistakes very early in the cycle. They have more of a feeling of being in the driver's seat and have a sense of accomplishment and ownership of the application, take pride in the final product and feel a need to share "their" application with their contemporaries.

The Learning Experience

Following this process was a real learning experience for the developers and the customers. It gave us a good insight for developing with RAD, operating with a self-directed work team, and keeping pace within the framework of developing technology.

RAD does not allow for skimping in the design process. Detailed attention to the database design process is essential. The database will be the foundation from which a good application can be developed. It is critical to have a prototype available for the first JAD session. The prototype is the tool that gets the customer's attention and motivates their involvement. The development team should set reasonable goals for each JAD session so that the team and the customers leave the sessions with a feeling of success and accomplishment. Roles should be assigned to each development team member for each session so that nothing is missed. Examples are:

* One person to facilitate the session
* One to track requirement and database changes
* One to operate and change the prototype
* One to keep minutes

The DART team used laptops during sessions to perform their assignments. Roles should be rotated at each session. It is important that the right application and customer for the first project. The customer needs to understand the commitment they have to make for the process to be successful. You must take care not to create unrealistic expectations of the time frame of the project.

One of the most important factors in ensuring the success of a self-directed work team is to choose the right people to serve on the team. Members must be open to new ideas, be willing to cooperate in the self-directed environment, and be willing to commit to the concept. One of the members should be appointed to the role of integrator. This role can be rotated to alleviate the feeling that the integrator is a supervisor. The integrator's role is to:

* Ensure adherance to standards
* Coordinate the pulling together of application modules
* Oversee system testing
* Serve as the customer contact
* Direct the flow of information within the project

Client/Server technology is evolving continually. It is important to use CASE tools when they are available to facilitate and help to document the development of the application. The project team should continue to explore new tools for design, development, creating on-line help, testing, and documentation. You cannot be afraid to experiment and you should never tell a customer that a request cannot be fulfilled without first exploring for a way to accomplish what they need.

Continuing The Process

The process we at North Carolina State University have followed to move into the Client/Server arena has been very successful for us. One of the key benefits has been that management has had the flexibility to change direction when necessary. We can adapt to shifts in focus brought about by

emerging technology or customer desires without negatively impacting progress. The second generation of the DART team started out focusing on learning about data warehousing and moving towards developing a data warehouse for the University. In response to the impact of legislative budget cuts and mandates to downsize throughout the campus and to the desire of the University Financial Officers to improve processing efficiency within departments, the focus of the team shifted to taking on short term projects to achieve more efficiency. An emphasis has been placed on publicizing campus wide applications as well as departmental applications that may be sharable. Campus departments are helping by funding both contract and student staff on a temporary basis to supplement the development team's productivity. At this point, the DART team is a dynamic, flexible group with no long term commitments. In addition, there is a group operating as a Continuing Quality Improvement (CQI) team to explore and evaluate new software, hardware, and ideas. The CQI team operates under the direction of the Administrative Computing Services Research and Development unit. Members consist of staff from all areas of Business and Administrative Systems Division. They are currently evaluating Object Libraries, Object Oriented development tools, and will be recommending a future direction for applications development at North Carolina State University.

We feel that we are on the right track for providing our customers with "Leading Edge" solutions to their problems. We have been able to assist other institutions in moving into the Client/Server arena and consider ourselves to be progressive in this area. The process we have followed to move in a new direction in applications development is one that has been highly successful for us and one that we are proud to share with our peers in hopes that we can help others to be equally as successful.

Data Warehousing Puts New Life into Legacy System

Kathy L. Fisher, Information Services & Telecommunications
Ferris State University
Big Rapids
Michigan

The major focus of this presentation is to explain how Ferris State University (FSU) is developing a data warehouse application to deliver timely, accurate student information to college-specific customers. This presentation will benefit anyone who wishes to serve academic customers by providing an PC database environment for working with legacy data.

Two underlying factors led to the development of this application. A more timely method for delivering reports to college staff is needed at a time when Information Services (IS) staff resources are diminished. The mainframe-based Student Information System presents information on an individual student basis; whereas, colleges often need to view student information in an aggregate form by college, department, course, or advisor. This application is designed to empower the customer to design and create reports and queries based on their individual needs with the ultimate outcome being an improved service to students.

PROJECT OVERVIEW

In recent years, Ferris State University (FSU) has experienced the re-engineering of many procedures that were once a common practice in daily office routines. In the fall of 1993, the University converted all academic programs to a semester calendar which necessitated the implementation of the current Student Information System (SIS). This change in technology brought about a new way of doing "business" across campus. A large majority of the batch reports previously produced by the Information Services department were no longer valid or usable. Hundreds of new on-line screens were introduced causing office staff to revise their methods for accessing student information.

At the time the University was learning to use this new system, it was also experiencing a decline in student enrollment. This led to financial constraints and reduced staffing in many areas, including both the Information Services department and the college administrative offices. At a time when the University needs to improve student services to remain competitive, it is facing a critical challenge in meeting that need. College administrative staff need to be able to easily access and summarize student information in order to serve students in a timely manner.

In an effort to re-engineer the process by which college administrators and faculty acquire information, Ferris State University is developing a data warehouse approach to working with legacy SIS data. Data delivery tools placed on client workstations allow staff to extract data from the warehouse for local data analysis and reporting.

RATIONALE FOR IMPLEMENTATION

Two underlying factors led to the development of this application. A more timely method for delivering reports to college staff is needed at a time when Information Services (IS) staff resources are diminished. The mainframe-based Student Information System presents information on an individual student basis; whereas, colleges often need to view student information in an aggregate form by college, department, course, or advisor. The data warehouse application is designed to empower the customers to design and create reports and queries based on their individual needs with the ultimate outcome being an improved service to students.

The traditional approach to accessing student information takes either of two routes: 1) a request for information is sent to the IS department, describing the data needed and the format in which it is to be printed, or 2) the data is viewed on-line through the use of CICS screens. There are limitations to both approaches that hinder timely access to information.

Limitations in the Delivery of Information

Several factors lead to a less-than-satisfactory turn around time for delivering information to the department or person requesting it. One factor is the limited number of programmers in the IS department. There are currently six full-time programmers available to respond to incoming report requests from over 16 functional departments, seven college administrative offices and various faculty and management personnel. In addition, fulfilling reporting needs constitutes only a portion of their workload. These report requests must be prioritized along with other application development, maintenance, and support tasks. This workload delays the delivery of information for days or sometimes weeks. If users need to query the IS department for assistance every time they require access to critical data, vital decision-making time is consumed and opportunities can be lost.

A second factor that often leads to frustration for the user/customer is the process of printing and delivering the needed report. The majority of reports currently generated are printed centrally through the Computer Operations department and delivered by the campus mail service. This type of delivery adds at least another day to the process. Recent implementation of an on-line Report Distribution System has alleviated some of this frustration. However, most users continue to request the information in hard copy form.

One last item that needs to be discussed is the effect budget limitations has had on this traditional approach. With the recent restructuring of academic programs and the drop in previous levels of enrollment, the university has had a difficult time in maintaining staffing levels -- not only in the IS department, but across campus. As other departments become lean and need to "do more with less", they are turning to the already downsized IS staff for technological solutions to enhance the effectiveness of their workflow. This puts an additional strain on IS resources to meet user needs for timely information.

Limitations of Mainframe Transaction-Processing System

In 1993, the university purchased and implemented a mainframe-based Student Information System (SIS). This system offered the ability to process semester-based data and added greater functionality than our previous student record system which had been in use for over 15 years. While this new system meets the needs for processing admission, financial aid, academic program and student account information, it lacks the capabilities of providing information that is readily available or useful for management decision-making activities.

This SIS system is a basic transaction-processing system utilizing CICS screens for data entry and verification. The application code is written in traditional COBOL language which is used to process data stored in VSAM file structures. This structure prohibits easy access for the average user and requires programmer intervention to turn this data into

useful information. In addition, all on-line screens are based on individual student name or identification number. This system works fine when specific student data is needed but does not provide executive information capabilities for summarizing the data into easy-to-digest formats, eye-catching graphics, or analytical trend data.

THE DATA WAREHOUSE APPROACH

The initiative for the data warehouse approach resulted from several circumstances coming together at the right time. As an IS manager, responsible for application systems and development, I was well aware of the current limitations of our present Student Information System and the ineffective process for delivering timely information. At the same time, I was enrolled in FSU's graduate program for Information Systems Management and was contemplating ideas for a thesis project. The two situations converged during a meeting with the Assistant Deans. Representatives from the Information Services department, including myself, were presenting a PC-database approach for accessing university financial data. This meeting evolved into a discussion of student information reporting needs with the Assistant Deans expressing their desire for a similar PC-based system for accessing student data. Considering the current backlog of requests in the IS department, it was decided that the best approach would be to develop this student data warehouse as my graduate program thesis project. Development activity could occur after normal work hours and would not interfere with other priorities in the IS department.

Implementation Highlights

The main objective of the thesis project is to define and develop a student information data warehouse on a SQL server that will be easily accessible through the use of client/server technology and PC-based reporting tools (e.g. MS-ACCESS). These features will provide selected end-users in the college offices with the capability to create their own reports, perform search and sort functions, produce mail-merge documents, and view data on-line in an Windows environment -- all from their individual client workstations. Thus, end-users will be empowered with effective management tools for planning and decision making.

In the initial planning phase, several discussions with the Assistant Deans from each college demonstrated the need to have multiple types of information relating to student data (e.g. recruiting and admission data, degree requirement information, and alumni relationships). However, after considering the resources available and the timeframe in which a useable application was needed, it was determined that a subset of this information would be used as a pilot project. Therefore, the initial data warehouse will be populated with data pertaining to currently enrolled students only. An eight-step implementation plan was developed for creating the data warehouse application (see Table 1).

Table 1

Step	Activity	Target Date
1	Determine immediate data needs for development of warehouse database.	April, 1995
2	Evaluate and select data warehouse platform.	June, 1995
3	Select PC-based reporting tools.	July, 1995
4	Determine workstation hardware requirements.	October, 1995
5	Extract mainframe student data for import to data warehouse.	October, 1995
6	Develop reporting applications for initial use and training activities.	December, 1995
7	Train pilot group and solicit feed-back.	February, 1996
8	Develop database administration procedures and system reference manual.	March, 1996

Step 1 involved a review of the IS department's most commonly received requests for information. The outcomes of this review were discussed with the Assistant Deans and were refined to include only biographic and enrollment information for currently registered students. The initial data that will be stored in the data warehouse includes the following:

- student biographic information such as name, birth date, handicap, citizenship, gender, and ethnic origin
- local mailing address of student
- permanent mailing address of student
- enrolled program of study, including college, degree, department and major
- current enrolled courses, credit hours, etc.
- current GPA, attempted hours, etc.

Step 2 covers the evaluation and selection of an appropriate platform to use for storing the files associated with the data warehouse. A Windows NT server running Microsoft's SQL Server software was selected for this purpose. This decision was based on a combination of factors. Previously, a Windows-NT server had been purchased and was controlled by the IS department for research and development purposes. Its only use at the time of the data warehouse implementation, was for a similar development project. Therefore, it could easily be allocated for the new project without conflicting with other university needs. In addition, the SQL Server software would provide a centralized relational database that could be queried to transmit only the requested data to the user workstation.

Step 3, the evaluation and selection of PC-based reporting tools, was based on three factors. These include its ease-of-use, integration to other Windows-based software, and its technical sophistication and potential for growth. The MS-ACCESS database product was selected as meeting these requirements and went on to become our university standard for PC databases.

Step 4 involved the selection of workstation hardware that would support the MS-ACCESS requirements for memory, disk capacity, and processor type. A specification sheet comparing these requirements with current workstation configurations was distributed to each of the seven colleges. This comparison chart was used to determine the upgrades needed for each workstation that would be equipped to access the data warehouse information.

Step 5 was the easiest step for me to accomplish. Being the manager of Application Systems, I was frequently involved in extracting data from our mainframe system This same type of extract process is used daily to fulfill the many report requests received in IS. The only learning that needed to occur was how to the import the data into the SQL Server warehouse to be queried with MS-ACCESS.

On the other hand, Step 6 is probably the most training-intensive and time consuming piece of the development. Not having worked with either SQL Server or MS-ACCESS before, I was starting at ground zero. Intense self-training exercises are conducted in "off" hours to prepare myself for development of the front-end, user-interface portion of the application.

Step 7 will involve training selected end-users from each college on the use of the delivered applications and the types of further reporting capabilities that can be accomplished. The goal is to develop four report/query applications and one mailing label application to deliver with the final product and to use for training purposes. This step is currently in the planning phase. The pilot group of users has been selected and includes one or two key users from each of the college administrative offices. Training sessions will be conducted by the application developer and will concentrate on two specific areas: 1) a demonstration on how to generate the delivered reports through the user interface, and 2) other report, query, and analysis functions available in MS-ACCESS.

Step 8 includes the development of database administration procedures along with a system reference manual. This process, although included as the last step of the project, will be ongoing throughout the development phase. Items contained in the reference manual will include the following:

- Schedule of data warehouse "refresh" procedures
- Procedures for adding new fields to the data tables
- Copy of extract programs used to transfer data from the mainframe SIS system
- Extract file definitions
- Copy of table layouts and definitions for each table in the data warehouse
- A data flow diagram of the system

The completion of the data warehouse along with the deliverables specified in each step is scheduled for March, 1996.

Impact on Customer Functions

The implementation of this data warehouse approach to student information will have a significant impact on customer functions related to data gathering and analysis. More timely information is provided for review by administration since information is available as soon as the data is queried. An improvement in data quality results from direct feeds of data to the warehouse, eliminating errors produced by manual data entry and the consequences of making decisions based on erroneous information. Hours of staff time that were involved in manual data collection can be freed up to add valuable interpretation and background information to reports. Data presentation can be tailored to the needs of different audiences.

The implementation of this proposed data warehouse application will enable the college administrative offices at Ferris State University to make effective management decisions regarding their enrolled student population. They will be able to readily access the information needed to verify and analyze college, program, department and course enrollment statistics.

The use of PC-based software on the client workstation adds the potential to access, manage, analyze, and present the warehouse data in various formats. Class lists, course registration audits, and departmental reports can easily be produced. Search and sort capabilities will result in faster data lookup and make specific student information more readily accessible. The ability to summarize, report, or project specific college or program data will be available. The integration with other Windows-based tools offers the potential to graphically analyze and view trends in enrollment data.

Local printing capabilities will be available to produce mail-merge documents, hard copy reports, or student mailing labels. This will relieve some of the need for the Information Services department to produce, print, and distribute ad hoc reports. The ability to quickly access and view student data and/or reports in an on-line Windows environment could result in a significant savings in paper costs.

Changes in Information Services Role

The implementation of this data warehouse approach to student information will significantly change the role in the Information Services department when fulfilling report requests and other information needs across campus. By empowering the college administrative staff and faculty with the ability to generate their own reports, mail-merge applications, and student mailing labels, the backlog of report requests received by IS can be significantly reduced. Technical staff time can be freed up to work on more complex programming issues. Other technological advances can be developed to meet university needs, resulting in a more challenging and rewarding work experience for the technical staff.

Reduced operational support will be needed for printing and mailing reports, resulting in less overtime pay to an already overloaded operations staff. Paper costs for the IS department can be greatly reduced while resulting in more attractive reports being produced locally in the colleges.

FUTURE PLANS

The primary objective of a data warehouse is to give end-users faster, easier, direct access to institutional data so that they can make quicker, more-informed and better decisions. This initial data warehouse project only targets a small percentage of the reporting and data analysis needs across campus. Used as a pilot research project concentrating on currently enrolled student data only, it is hoped that the outcomes will provide significant insight into the methodology, impact, and usefulness of this approach for meeting data access needs in other areas. These same techniques can be used to access data in the financial, human resource, or alumni/development systems.

Setting the Stage for Client/Server Architecture

As technological advances progressively move to client interfaces for ease-of-use and direct access to decision-making data, so, also, must the underlying architecture support these client needs. In general, the mainframe provides a great deal of reliable processing power and storage space for large database systems. However, it does not provide adequate interfaces for easy access by the average end-user. On the other hand, microcomputers have relatively friendly and powerful user interfaces, but they lack the storage and processing power of larger computers.

In a true client-server system, computers share the processing load. The data warehouse approach allows each computer to do what it is best at. The administrative functions of the university are based on mainframe systems that are tuned for high-volume transaction processing (e.g. registering students, crediting and debiting university accounts, and issuing paychecks and W2 forms). By placing the data warehouse on a separate SQL server platform, the efficient processing of the University's operational systems will not be compromised by ad hoc queries against the data they contain. In turn, the data warehouse can be tuned to provide optimal performance for ad hoc queries from the client workstations.

Improved Customer Relations

The growth of this data warehouse to include additional stores of data from multiple sources will lead to improved customer relations throughout the university. Staff will be motivated to re-tool, data accuracy will be improved, and many workflow and decision-

making processes will be re-engineered for improved effectiveness. Several other factors relating to this potential include the following:

- **Decision Support**. FSU is currently engaged in developing an Executive Information System (EIS). Decision-support capabilities will provide all interested parties with the information necessary to make the best possible decisions regarding the status and progress of FSU. Included in the benchmarks relating to the development of this EIS are the considerations for existing databases, accurate and reliable data, and alignment with the university's Information Technology Strategy. The development of this data warehouse will add to and strengthen our portfolio of EIS tools.

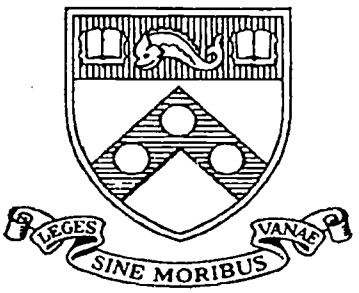
- **Decentralization of Ad Hoc Reporting**. With ad hoc reporting responsibilities currently placed in the IS department, customers are not always fully aware of where their requests fall in the prioritization process or what currently exists in the programming backlog. The development of the data warehouse will provide end-user departments with the ability to produce their own ad hoc reports. The "ownership" of the programming backlog will shift from the IS department to the colleges and customer departments. Prioritization of reporting needs will be done within individual departments and will not conflict with other campus needs.

- **LAN Server Consolidation**. The promotion of a central server for data warehouse processing lends itself to a streamlined process for data entry and administration. Many departments currently maintain data on local servers and individual workstations. This often results in the duplicate entry of data originating from the same source. By consolidating LAN servers into a central database server, this redundancy of effort can be eliminated. The central server can handle the database processing while the client receives only the data that was requested. This model causes minimum network traffic.

- **Continuous Quality Improvement**. This data warehouse approach will provide for more accurate, reliable data. Duplicate departmental databases that may not be systematically maintained to assure accuracy will be no longer be needed. In addition, this approach will empower the customer to detect exceptions from the norm by providing new perspectives on production data. These exceptions can be noted and reported to the owner, building a continuous effort to provide quality at the source.

CONCLUSION

Ferris State University is committed to improving customer service. By providing easier methods for accessing institutional data, customers can have needed information at their fingertips for improved decision-making, analysis, and reporting. The implementation of this data warehouse approach for student enrollment information, will move the university one step closer to that goal. To date, the project is well underway, and is currently being evaluated for use in other areas.



University of Pennsylvania

CAUSE 95

Assessing Risk: Developing a Client/Server Security Architecture

November 22, 1995

Dave Millar, University Information Security Officer (millar@isc.upenn.edu)
Noam Arzt, Director Information Technology Architecture (arzt@isc.upenn.edu)
Bill Ramirez, Systems Programmer (ramirez@isc.upenn.edu)

Introduction

The University of Pennsylvania's new Data Warehouse and Financial Systems are exploiting technology to bring to Penn flexible new ways to organize and manage data, making it readily-available for both operational and planning needs. Along with the benefits of the new technology, however come risks which the University must address to ensure the integrity of its information assets. The decentralization of data and computing, and the use of open networks, open systems and open standards exposes Penn to new vulnerabilities. Penn can no longer rely on the use of obscure operating systems and networking protocols to protect our systems and our information.

The Client/Server Security Standards Task Force was created to identify the threats to information security posed by the new technologies being adopted for the Data Warehouse and Financial systems. This report documents the threats which the group feels are most serious, and provides a rationale for the group's recommendations.

Bringing Together the Right Players

As Penn planned the implementation of Oracle-based Financial Systems and the Data Warehouse, it was apparent that careful thought would need to be given to how to assure the security of information in the new technical environment. For that purpose, the Client/Server Security Standards Team was formed, with the following mission:

> *Ensure that as Penn moves from mainframe-based computing to distributed, client/server computing, it can ensure the security of administrative systems and data.*

Care was taken to ensure that other planning efforts were taken into consideration, and were involved as necessary in the task force's work (e.g., campus-wide Distributed Computing and Network Architecture task forces).

Scope

Figure 1 shows that the initial focus of the team is on the implementation of Financial systems and the Data Warehouse; subsequently, attention will be given to the broader implications for administrative computing University-wide. The first phase of the Data Warehouse application was implemented in December, 1994, and additional phases will be implemented over time. Additional modules of the new Financial System are expected to be available in July of 1996.

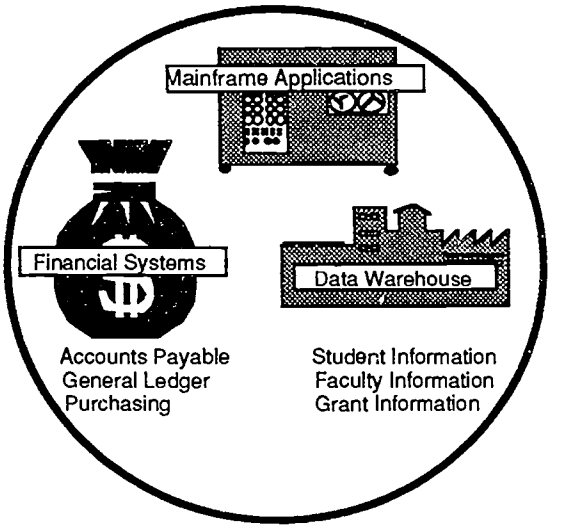


Figure 1 - Scope

While the group's charge did not initially include mainframe computer applications, the group came to believe that the following mainframe security issues should be included in the project scope:

- *Sniffing* To the extent that ethernet is used to connect to the University mainframe (about 40% of all mainframe connections), the legacy applications are no more immune to password or data sniffing than the new Financial Systems and Data Warehouse applications running on the server. Any data which travels over ethernet is subject to sniffing. The team felt that while the primary focus of the group is solving the problem of sniffing for the Financial Systems and the Data Warehouse, consideration should be given to how such solutions may apply to mainframe applications.

- *Single Sign-on* The new Financial and Data Warehouse applications will create new accounts and passwords for users to remember. The team felt that any recommendations for strong user authentication must include an assessment of the costs of mainframe integration, permitting a single sign-on for administrative system users.

Technical Environment

The technical environment for Project Cornerstone consists of a large, multi-processor Unix server connected via the campus' baseband enterprise network to a variety of desktop computers. Users will either connect via Telnet , or by making a client/server connection to an Oracle server using Oracle's proprietary SQL*Net client/server product via the TCP/IP protocol. Data will be exchanged between the Unix server and the mainframe. Financial system implementation plans and timing are presently under review.

Methodology

The team took the following approach to fulfilling its mission:

1. Survey other institutions for information
2. Identify trends in information security technology and practices
3. Identify the information assets we are trying to protect.
4. Identify the threats to those assets. Who might the actors be? How might threats be carried out? What are the implications of the threats.
5. Rank all of the threats. Consider only those threats which are credible and which could inflict significant harm to the University.
6. Validate the threat analysis with a knowledgeable focus group
7. Develop solutions to the credible/harmful threats.

Information from Other Institutions

Based on an informal survey conducted by Penn State in the Winter of 1993/1994 and reported verbally to members of College and University Information Security Professionals (no written report available) of eight private and five state educational institutions:

- 77% of the respondents have not updated their policies to address the migration to client/server architectures.
- 38% of the respondents require that all backbone-attached devices must have User ID/ password or other authentication mechanism.
- 54% of the respondents are planning to migrate to more secure authentication mechanisms than passwords - rated from most to least important: Kerberos, tokens, encryption, signatures, DSS.
- 54% of the respondents required badging or card systems to gain physical access to telecommunications installations.
- 24% of the respondents are now using tokens or smart cards.
- 54% of the respondents are employing a firewall on at least some systems/networks. The effectiveness of firewalls was ranked as somewhere between "effective", and "somewhat effective" (the choices also include "ineffective").
- 77% of the respondents are not encrypting data in local LAN environments.

- 24% of the respondents are encrypting data between buildings and facilities.
- None of the respondents are encrypting data over WAN communication links.

Based on informal discussions with peer institutions, the following themes emerged:

- Some institutions which continue to run their administrative computing over SNA or similar proprietary protocols have not had to face the security challenges of open systems computing and may choose to wait for the solutions to catch up to the problems before wading in.
- Of those institutions facing the problem of open systems, client/server computing, many applications are still in their infancy, as are the solutions to the corresponding security problems. We spoke to a large number of organizations that were conducting limited studies or pilots, or limited implementations of security solutions. Concerns expressed about the solutions included scalability, performance, and the long-term viability of the solutions given confusion over what standard(s) will emerge.

Trends in Information Security Technology and Practices

The following trends were identified:

- Movement away from re-usable, plain text passwords to one-time and/or encrypted passwords using the following technologies:
    - DCE/Kerberos
    - Authenticator tokens
    - S/key - one time password algorithm
    - Public key digital signatures backed up with proof of authenticity "certificates"
    - Digital signature (and encryption technology) embedded in hardened PCMCIA-compatible smart cards: e.g. Fortezza card.
- Movement away from embedding security features (authentication, encryption, non-repudiation) in high-level applications (e.g. PGP, Kerberos) to lower, more pervasive levels in the Internet protocol, making security more transparent to users:
    - Secure Sockets Layer (SSL) from Netscape
    - Private Communication Technology (PCT) from Microsoft
    - SKIP (Simple Key Management for Internet Protocols)
    - Internet Protocol version 6
- Increasing use of firewalls
    - External firewalls

- Internal firewalls
- "Private" internets with firewall address translation

- Network-based tools for auditing host security

  - SATAN
  - ISS
  - Pingware

- We found no single, unified and comprehensive solution to the security problems posed by client/server computing in the heterogeneous technical environment we face. Organizations working on these problems are probably best served by accepting that problems may have to be solved on a piece-meal basis with solutions that may be obsolete in less than two years.

Assets and Threats to Those Assets

Figure 2 shows some of the assets that the Client/Server Security Standards team was charged to protect, and the threats against the assets. Information and infrastructure assets are at risk from threats which may originate either from within Penn, from the outside Internet, or which may simply be the result of an accident. The group was charged with developing recommendations for solutions to protect against the most credible and harmful threats.

The project team developed a complete list of the threats to the information assets. These threats were divided into three categories: **threats against the desktop** (7 identified); **threats against the server** (15 identified); and **threats against the network** (8 identified). A rationale was then articulated to explain the team's reasons behind its assessment. Finally, the threats were ranked along two dimensions: the **likelihood** that the threat might take place, and the **harm** that would be experienced if it *did* take place. Both these rankings were done on a "high/medium/low" scale.

The most credible and harmful threats include the following (rationale follows in italics):

1. Desktop Computer Data Disclosed: *Someone discloses sensitive information (e.g. payroll information, student grades), which was obtained without authorization from a desktop computer. As more data is available to be downloaded , and as it becomes easier to download, this risk will increase.*

2. Desktop Computer Data Altered or Destroyed: *Either intentionally (someone gains unauthorized access to a desktop computer) or unintentionally (hard disk crash, flood, fire, accidental file deletion by the user) , data is altered or destroyed. Since altered data is difficult to detect, it presents a special threat.*

3. Passwords Compromised by "Trojan Horse" Program on Desktop: *A "Trojan Horse" is a program that performs unknown, and unexpected as well as expected tasks. It is either placed on a system without the owner's knowledge, or it is placed there by the owner, ignorant of any harmful effects. One example is a program which logs the user onto a system (expected) while at the same time storing the password in a hidden file (unknown, unexpected) for later collection and illicit/inappropriate use.*

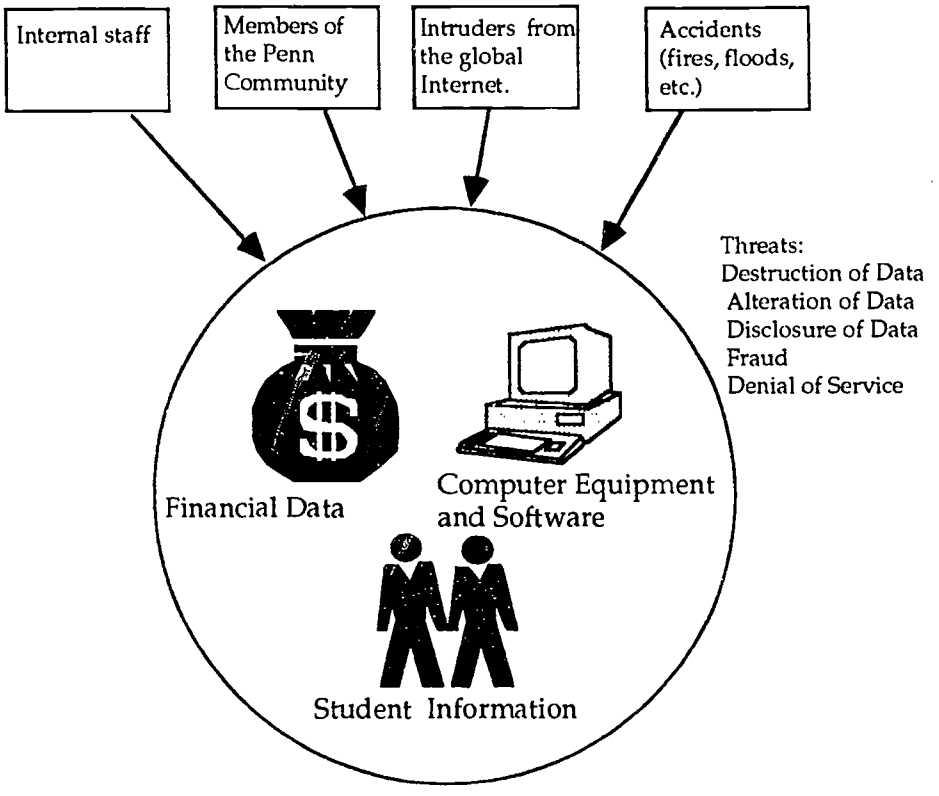


Figure 2 - Assets and Threats

4. Users Logged in on Unattended Desktop Computer: *Anyone who comes upon an unattended desktop computer , which is logged onto a server application, has all of the privileges of the user. This problem may become worse as the client/server architecture eliminates the incentive to sign off of mainframe applications to save on connect time charges.*

5. Employee Fraud/Sabotage: *Data or programs are altered, disclosed, or destroyed either by a properly-authorized employee abusing his/her access privileges or by an employee who obtains excessive access privileges. Such incidents are more common during restructuring/layoffs.*

6. Improperly Protected Server Accounts Used to Gain Access: *Privileged accounts on the server (those with broad capabilities normally only given to system administrators) are targets of intruders for obvious reasons. Poorly protected user accounts , though less powerful, also make tempting targets, since they provide a toehold to gain further privileges. Poorly protected accounts include those with easily-guessed passwords, or no password at all, accounts held by users who write down or script their password for easy sign-on, and dormant accounts belonging to inactive users. An intruder on a server can disclose sensitive information, destroy or*

alter data or programs, and leave behind hidden programs which steal passwords from unsuspecting users of the system, and which make future intrusions easier.

7. **Intruder Exploits Server Vulnerabilities to Gain Access:** *New security vulnerabilities in the Unix operating system and in Internet network services appear almost weekly. Detailed scripts for how to exploit these vulnerabilities are circulated widely within the hacker community, and patches to fix the problem are sometimes not available from vendors for months.*

8. **Privileged Employees Accidentally Delete or Alter Data or Software:** *In performing their duties, an application system administrator, a production control employee,, DBA or system administrator accidentally causes harm.*

9. **Server Destroyed in an Accident:** *Leaking pipes, power failures and equipment failures are not uncommon.*

10. **Address Spoofing:** *Someone spoofs network addresses to gain access to servers, and uses that access to read/alter data or set up future access. Someone "spoofs" a network address by using their host computer to "impersonate" a trusted computer, thereby improperly gaining special permissions and access that only the trusted computer should have.*

11. **Sensitive Data Disclosed Through Packet Sniffing:** *Someone uses a packet sniffing tool (software which allows a computer connected to the network to view data intended for another host computer on the network) to read sensitive data being transmitted over the network.*

12. **Accounts Compromised Through Packet Sniffing of Passwords:** *Someone uses a packet sniffing tool to capture accounts and passwords providing access to the server. In Spring, 1994, the Computer Emergency Response Team reported a high incidence of such attacks, including compromised accounts numbering between the tens and hundreds of thousands. CERT recommends that organizations stop sending unencrypted passwords over the network and move to encrypted, one-time password authentication schemes. Extensive harm could done with unauthorized access to numerous users' or system administrators' accounts.*

13. **Network Unavailability:** *Network connectivity unavailable due to accidents such as fiber cuts, flood, fires, power outage, etc.).*

After these threats were developed and articulated, a campus-wide focus group, consisting of both information technology professionals *and* functional managers responsible for the management of processes that use the information assets being protected, reviewed the threats and rankings in detail and suggested changes and additions.

Proposed Solutions

The following solutions were developed and proposed to try to mitigate the serious threats detailed above:

Policy: A comprehensive administrative information security policy should be developed which defines the information assets that the University wishes to protect, and defines the responsibilities of both computer users and computer administrators to ensure protection of those assets.

Server Protection: It is recommended that University administrative computing servers be protected by a firewall. This would allow Penn to better control how they are accessed and used.

Password Hardening: The Kerberos network authentication service was recommended for users of the Data Warehouse and Financial applications. A fair amount of time was spent looking for, testing, and evaluating Kerberos servers and client software. The project team was concerned about vendor support for these products, so commercial versions were evaluated wherever possible. In the end, the team was disappointed by what it found. Commercial products did not have even the most basic features required for a large-scale implementation among a wide range of users.

It was additionally recommended that privileged users will be required to supplement their Kerberos authentication with token authentication. In addition, some users unable to use Kerberos authentication (primarily due to insufficient desktop resources) will instead be required to use a token to authenticate themselves. A token generally uses either a challenge/response algorithm, or a clock-driven algorithm to create one-time passwords, which can not be re-used later, even if they are disclosed by network sniffing. A token authentication server provides token authentication services for the server.

The project team tested the SecureID token extensively and found that it met the project's needs for supplemental authentication. Given the poorer results finding Kerberos-enabled software, consideration is being given to wide-spread deployment of a token card to reduce password exposure even though it does nothing to enable data encryption.

In addition, warehouse users would be required to select strong Oracle passwords using the Braintree SQL*Secure software.

Data Encryption: An additional feature of Kerberized telnet products is the ability to encrypt the full stream of data during a telnet session. Since initially most users will be accessing the financial applications via telnet, this feature is highly desirable. Once again, however, the team could not identify products (commercial *or* public domain) to support these functions to their minimum specifications.

The warehouse users are accessing using client/server products. Oracle's Secure Network Services product is a piece of the Oracle product family that layers on top of SQL*Net and provides end-to-end data encryption with no modification of the client or server application. The project team hopes to deploy it as more financial applications migrate from host-based to client/server.

Desktop Computers: Users storing sensitive data on their desktop computers will be required to protect the data using desktop access control and encryption software unless the computers are otherwise physically secured. Policy should be developed to define the terms and conditions.

Lessons Learned

Be precise with scope: We faced a complex array of security issues to focus on. Product restrictions make it too difficult to select a single solution. Changing requirements blur the focus on the problem. All of the above factors combined demand constant focus on the scope of the project in order to limit the energy to the task at hand (securing the Cornerstone environment).

Stay on top of the vendors: Constant developments in products and technology offered requires constant review of available products that meet requirements. The configuration being secured differs from what other users implemented, hence the vendor requirements differ. This makes it hard to draw on the experience of others.

Limitations of Technology: The technology the project required is just not there yet. Products were usually inconsistent across vendors, and some product families were internally inconsistent. Limitations with the Kerberos components abound. On the other hand, new products are emerging constantly.

The process as an end in itself: Better understanding of how the tested security products work including their strengths and weaknesses. Increased awareness of network topology, network vulnerabilities, and Unix operating system vulnerabilities would be very helpful. The creation of the task force (made up mostly of technical people) and the focus group (made up mostly of business people) greatly increased the awareness of the risks of open systems and client/server computing. The process also helped introduce a risk management philosophy, informing business managers of the technical risks in terms they felt comfortable with. As a result, business managers responsible for applications and data are better able to determine acceptable levels of risk, and to make appropriate tradeoffs.

The recommendations have been presented to the project sponsors and have been accepted where there are products to implement them. Deployment of the firewall has begun. Use of token cards is still being assessed financially. Policy development will begin shortly.

Saving Enterprise Data from the Client/Server Grinch

John D. Porter
Susan D. Moore
John J. Rome
Arizona State University

Abstract

In the world of client/server technology, where enterprise databases are distributed, how do postsecondary institutions keep from losing control of their data sources? One university's answer was to create an enterprise data infrastructure to ensure the integrity and integratability of enterprise data. The data infrastructure serves as the "hub" of a portfolio of distributed enterprise applications. This paper describes the implementation of the first subject area of the hub, the Affiliate Management System, which contains enterprise data on the organization's customers.

SAVING ENTERPRISE DATA FROM THE CLIENT/SERVER GRINCH

In the world of client/server, where enterprise applications and databases are distributed, how do postsecondary institutions keep from losing control of enterprise data? This was the question asked by Arizona State University's data administration and information technology managers as they observed the outward flow of enterprise data as business units pushed to implement client/server solutions.

ASU's Computing Environment

Arizona State University is a Research I, multi-campus university consisting of three campuses in the Phoenix metropolitan area. Total enrollment exceeds 45,000. The administrative computing environment centers around IBM 3090 MVS architecture. The student information system, developed by ASU, is an on-line and batch application using a large integrated data base (IDMS). The human resources system is a purchased package running on DB2. The university's financial system is also a purchased package running on IDMS, but migrating to DB2 this year. Another IBM product supports electronic mail and FOCUS applications. FOCUS, which was used for end-user reporting and departmental applications, is being replaced by client/server applications. The first client/server implementation was a data warehouse. ASU is exploring other client/server applications in a re-engineering project involving the student information system. Client/server applications use Sybase on a UNIX server.

ASU's Difficulty in Implementing Client/Server Technology

Implementing client/server technology at ASU proved more difficult than promised by industry. Client/server technology is less reliable, secure, and timely than its mainframe predecessor. Networks add new layers of complexity, and monitoring performance and tuning of servers is imperfect. The result is that for complex organizations like ASU, quickly moving mainframe legacy systems to client/server on a major scale may be too difficult. ASU's information and technology managers came to this conclusion re-engineering the organization's student information system. Given the problems in piecing together the new and old technology, ASU realized that mainframe systems were going be around a long time, coexisting with client/server applications until the industry solves some of the problems inherent in distributed and shared computing. For ASU, slowing the pace of client/server development provided data administration and the information technology mangers the time to devise a solution to the critical data problems posed by client/server.

The Need for an Enterprise Data Infrastructure

ASU's information managers view client/server as a desirable computing solution because of the potential for new and improved services for less cost (see anticipated client/server benefits in Table 1). However, client/server also raises many technical, data and security risks. One big risk is that distributed databases and

systems will degrade the integrity of enterprise data and lessen the ability of the organization to integrate data between computing systems.

These concerns lead ASU's information managers to conclude that the organization needs a client/server enterprise data infrastructure (EDI). The vision of EDI is that ASU captures the critical enterprise data elements required by *multiple business processes* in a single data model and database. EDI functions as the hub of

Table 1

Benefits of Client/Server

- Flexibility
- Potential of Technologies
- Scalability
- Increased Productivity
- Long Term Cost Savings

a portfolio of mainframe and distributed enterprise applications (see enterprise hub at Figure 1). Some of the applications interface with the hub via client/server over ASU's network backbone. Independent systems interface with the hub via program bridges between the two applications.

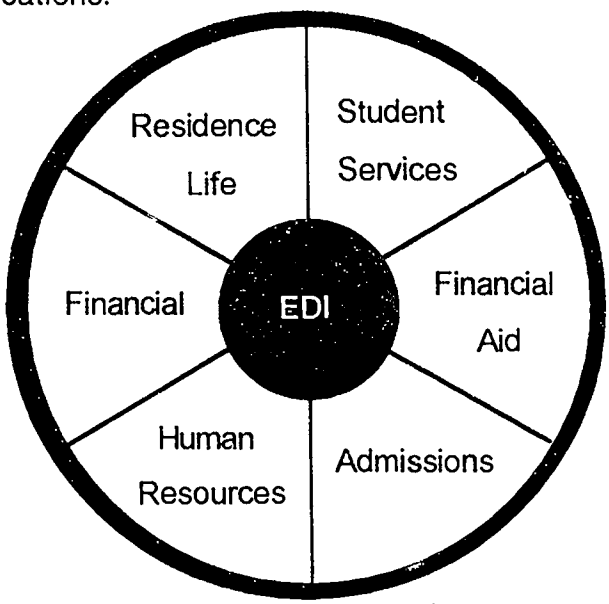


Figure 1. Enterprise Data Infrastructure is the Client/Server Hub.

The advantages of the EDI are many. As ASU implements more and more client/server applications, EDI serves as a seamless data foundation, containing the enterprise data needed by distributed applications. Tremendous economies flow to ASU by managing enterprise data through a single model and updating these data in a single database. Distributed systems benefit by having the most recent occurrence

of data, and information engineers benefit by having enterprise formats and standards to guide their work. Through EDI, ASU is banking on improved data integrity, lower costs in updating enterprise data, less data redundancy, and improved services to customers (see benefits listed at Table 2).

Table 2

Benefits of Enterprise Data Infrustructure

- One-stop update
- Data foundation
- Contributes to data integration
- Ensures data integrity
- Distributed developers know enterprise formats and standards

Pieces of ASU's Enterprise Data Puzzle

Advocates of data administration maintain that enterprise data need high-level or corporate management. However, ASU found that defining enterprise data is problematic because almost every data element is "enterprise" to someone or something! In creating ASU's EDI paradigm, data administration and the information technology departments conceptualized not only the data but *what the data were about*. For example, improving customer service is a major goal at ASU; therefore, much of ASU's enterprise data concerns customers.

Following this logic, ASU identified four major subject areas ("puzzle pieces") to EDI: (1) persons or organizations that do business with ASU, (2) work units or departments that comprise the organizational structure, (3) facilities used by both ASU's customers and work units, and (4) programs (academic, research, public service, etc.) that impact ASU's customers (see EDI subject areas at Figure 2). These four high-level subject areas are the foundation of ASU's enterprise data infrastructure. ASU re-designated the customer subject area as the "affiliate" after defining customers more broadly, e.g. students, employees, vendors, donors, constituents, etc. Affiliates are anyone with a business relationship (affiliation) with ASU.

ASU's vision is that through a single model and database, ASU enforces the business rules for the data in each subject area. For example, if an affiliate updates their address, the business rules for address are enforced by the affiliate data model and the update is stored in one location. Distributed or mainframe databases may replicate the address in affiliate, but the update occurs once.

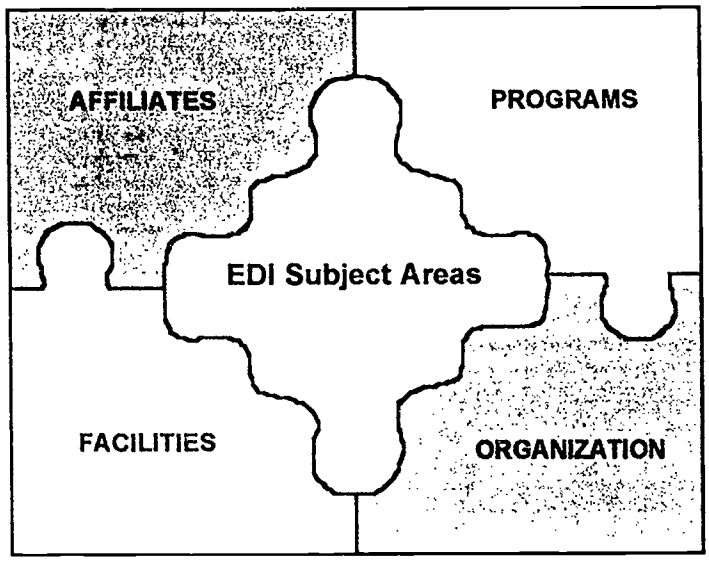


Figure 2. ASU's EDI Puzzle.

ASU decided that the first EDI subject area built should be the area impacting the organization the most. The data administration and information technology departments agreed that the subject area with the most pervasive impact concerns affiliates. This decision was made in part because of ASU's student information re-engineering project. ASU wants to maximize the number of new client/server applications and plans to establish the interface with the Affiliate Data Model from the beginning for these applications. Also, ASU's business units plan on implementing a "single" campus card for all business transactions. The single campus card requires affiliate data services, such as a single affiliate ID for both students and employees. By implementing the Affiliate, the campus card initiative has the ability to interface more broadly with other applications at ASU.

The Affiliate Management System and Its Implementation

The Affiliate Problem

Arizona State University has many business relationships with persons and organizations. Capturing information about the affiliations these persons and organizations have with ASU is an important factor in developing new and improved services. In many cases, ASU affiliates have multiple relationships with the organization which are not understood.

In the past, ASU's business practices and computing systems processed information about affiliates based on the needs of the department and not the needs of the enterprise. Information about affiliate characteristics such as address, name, ethnicity, etc., is captured and stored in different formats using different definitions. When the same data are carried on multiple databases, the data are frequently out of sync with no way to know which occurrence of the data are correct. As client/server technology advances, it is likely that the resulting distributed databases and

applications will make this problem worse, unless enterprise data models and policies are implemented.

The end result is reduced levels of service, more expensive systems maintenance, a loss of data integrity, and redundant storage of the same or similar data. However, the most strategic loss is the failure to fully understand the myriad of relationships that a single affiliate may have with the organization. Understanding such relationships could result in a totally new view of ASU and how it serves its customers. ASU hopes another big benefit will be significantly reduced computing and information costs.

The Affiliate Solution

The Affiliate Management System is an enterprise application and data model which solves these problems by defining and capturing in a single data base the attributes and relationships of affiliates that are important to more than one ASU departments. Since affiliate data is stored in a single data base, in a single format, and based on a single definition, the data are more easily managed for the benefit of the organization. When affiliate data are updated, all distributed applications interfacing with the Affiliate System immediately have the benefit of the new information.

The Affiliate System has the advantages common to all EDI subject areas, i.e., reduced maintenance, update and storage costs, and increased data integrity and integratability (review benefits at Table 2). However, Affiliate brings unique benefits by providing ASU a single customer ID that can be linked to other IDs, giving an ASU affiliate the capability of maintaining their "personal" data, making it less complicated for ASU to build applications to support sophisticated customer services (e.g., single-campus ID card) or affiliate to affiliate relationships (e.g., advisor/advisee), and making it easier for ASU to strategically comprehend the full extent of the relationships it's customers have with the organization (see unique affiliate benefits at Table 3).

Table 3

Benefits of Affiliate Management System

- Single ID (could be linked to other Ids)
- One stop update to personal info
- Supports sophisticated services (i.e. mailings, e-mail, fax, etc.)
- Ability to study affiliate relationships with ASU
- Supports affiliate relations with each other

The Affiliate Model

The high-level Affiliate Data Model contains nine entities (see model at Figure 3). The most important entity in the model is the Affiliate. To be an affiliate, a person or organization must have a name and a location (a way for ASU to communicate with the affiliate, i.e., address, phone, or e-mail, etc.). Affiliates are randomly assigned a unique ID which is a permanent identification number. Business units decide when to create an affiliate record. For example, the student recruiting office could decide to create an affiliate record for prospective students or wait until the "recruit" submits an application for admission.

In the Affiliate Model, there are two mandatory relationships between the affiliate and other entities. One mandatory relationship is between the affiliate and an entity called "Affiliation". Affiliations are the "first-order" relationships affiliates have with ASU, e.g., student, employee, donor, vendor, etc. Second-order relationships (relationships which are not mission critical, e.g., a dorm student) are not included in the Model. ASU's data administration and information technology managers believe relationships which exist because of previous relationships are best captured in departmental applications.

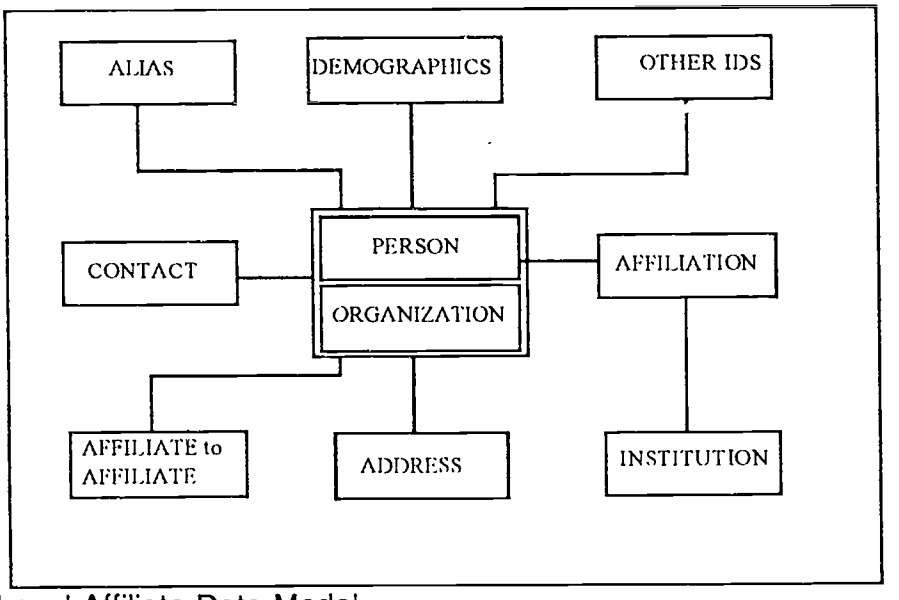


Figure 3. High Level Affiliate Data Model.

A second mandatory relationship exists between the Affiliate entity and Address. In order to become an affiliate, a person or organization must have a way of being contacted by ASU. This may be a mailing, phone call, or e-mail message, but the organization must have a way of communicating with the affiliate. This information is captured in the Address entity, where an affiliate can have multiple addresses, phone numbers, and e-mail locations with differing formats and differing uses.

Alias (other affiliate names), Demographics (ethnicity, birth day, gender, etc.), and Other IDs are entities which may or *may not* exist for an affiliate. These contain data which describe or identify the affiliate to ASU. The entity "Contact" contains information about persons who represent the affiliate. For example, a corporation may

have multiple contacts that represent the organization with ASU. In this capacity, these persons are not directly affiliated with ASU, except through their relationship with the affiliate. This is also true of a student's emergency contact. The Contact entity contains the contact's name, address and phone number, and nature of the contact's relationship with the affiliate.

Affiliate to Affiliate relationships is an entity which provides for non-mandatory relationships between affiliates, some of which may be important for ASU to capture. For example, if a group of donors are related in some way, that relationship may be strategically important for ASU to capture. More probable examples are faculty with similar research interests, or advisor/advisee relationships. By capturing this information in the Affiliate Model, it is easier for ASU to build distributed applications which support the business needs that emerge from these relationships.

Only one entity in the Model is related to an entity other than Affiliate: Institution. At ASU, there are three campuses (Main, East and West) which comprise ASU as an Institution. Affiliates can be affiliated with one or more campus, but this relationship only has meaning when the nature of the affiliation is known. This relationship is mandatory so that it is clear what relationship an affiliate has and where (i.e., campus) that relationship exists.

Creating the Affiliate Management System meant that ASU developed new business rules to govern the Model and the data captured by the System. These rules (see list at Appendix A) are still in the process of developing, but do indicate a significant shift in ASU's computing culture.

Data Trustee Program

Building the Affiliate Management Systems resulted in many data issues which are not completely resolved (see list of issues at Appendix B). ASU is creating a data trustee program to manage enterprise data and resolve the data issues posed by client/server. This policy states that a data trustee is assigned to each enterprise data element in the Affiliate Model. The data trustee is responsible for managing these data according to University data policies, including facilitating access and insuring data security. This program is administered by the Data Administration Office. Data Administration selects the data trustee, coordinates communication between data trustees, and facilitates all activities related to managing Affiliate data. Data Administration is the ultimate authority over the use and management of all enterprise data at ASU.

Developing the Affiliate Management System

Phase One (March to June 1994)

The initial development phase occurred as part of a three month pilot project. The purpose of the pilot was to test a new development methodology (Information Engineering), a new development tool (Composer by Texas Instruments), and a new computing environment (client/server). The pilot consisted of developing the Affiliate Model and an application that processed a specific type of affiliate called constituents. During this phase, an overall affiliate model was developed but detailed implementation only included those pieces of the model that were needed for the

constituent application. Business analysts from human resources, the registrar's office, institutional research, and data administration participated in the project as well as information technology analysts.

Second Phase (November 1994 to January 1995)

The second development phase included a larger and more diverse group of business analysts with representatives from admissions, financial aid, the graduate college, the registrar's office, sponsored projects, data administration, and the law college. Their charter was to review the "pilot" Affiliate Model, business rules, entities, and attributes and make any modifications needed to implement the model.

Third Phase (May 1995 to May 1996)

The third development phase is to implement the model validated in phase two. This involves completing all entity and attribute definitions, and building windows to maintain what was determined to be "core" affiliate information. The idea is to create a generic application to add and maintain affiliate information, knowing that additional windows will be needed later to enter affiliate information pertinent to a specific affiliation type.

Conclusion

ASU believes that building an EDI is essential if client/server is to produce the benefits promised by industry. The technologists will eventually solve all of the implementation problems of client/server. Interfaces will be smooth and system performance not a problem. What will be a problem is the loss of data integrity and ability to integrate data strategically as distributed applications capture and store enterprise data based on departmental definitions and standards. In building the Affiliate Management System, ASU is hoping to avoid this problem for critical segments of its business. At this point, the jury is still out on whether this approach will work. But be sure to stay tuned for future announcements!

APPENDIX A

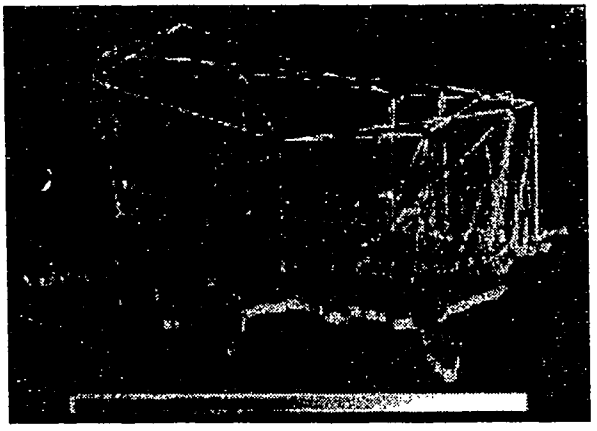
Affiliate Business Rules

1. Who is an affiliate?
   An affiliate is a person or organization doing business with ASU
2. What entities and attributes of objects belong in the Affiliate Data Model?
   Entities and attributes of objects belong in the affiliate data model if they are enterprise, common to more than one "business process", and a "characteristic of an affiliate".
3. What is required information to create an affiliate record?
   To create an affiliate record, an affiliate must be separately identifiable, and have a means to be contacted.
4. What are valid affiliation types?
   Affiliation types are fundamental business relationships with ASU that are not concurrently dependent on other relationships. Examples of affiliation types are student, constituent, vendor, employee, and donor.
5. How will ASU format address?
   Affiliate addresses conform to the U.S. Postal Service's format. The affiliate address structure is extremely complex. It allows for multiple address types and multiple addresses for each type. It provides the capability to designate an address as confidential or to be used for a specific purpose, such as mailing grades.
6. What data is excluded from the Affiliate Model?
   Highly sensitive relationships are excluded from the affiliate model to insure the privacy of those relationships. Examples are HIV status, disability, etc.

APPENDIX B

Ongoing Affiliate Data Issues

1. ASU's data administration function and data trustee policies play an integral role in the new client/server environment. These functions and policies are evolving and represent a cultural shift in how ASU's computing community views enterprise data.
2. ASU needs new policies and protocols to insure fair information practices. Since the integration capabilities of the Affiliate Management System make enterprise data more powerful, it is paramount that employees understand the appropriate and ethical uses of the new capabilities.
3. In order to successfully implement the affiliate concept, business units need to agree on data definitions and usage. The user community must abandon the "ownership of data" mentality that characterized past development practices.
4. ASU needs a way to secure certain sensitive or private data in the affiliate database. Due to the nature of the information or the position of the affiliate, some data should not be accessible by the campus community. The client/server environment opens a new set of security issues that were partially resolved in the mainframe world and need resolving in the client/server world.
5. ASU has several large purchased applications that need to interface with the Affiliate Management System. These applications carry name and address information about employees and vendors that must be incorporated into the Affiliate Database. ASU has yet to build the bridges linking these applications with the Affiliate, but is considering data replication.
6. Affiliate information needs to be updated by programs created with different application development tools. ASU needs to maintain the integrity of the data by enforcing business rules about how data is updated. This becomes more difficult when multiple application development tools are involved. The client/server industry needs to address this problem.



UNTANGLING THE WEB

Taking Advantage of WWW Technology

Note: This is an HTML-only track presentation in three segments. For the benefit of the CAUSE95 printed proceedings, the URL's of the three documents are:

- **Title page** http://www.lehigh.edu/~intel/cause.html
- **Untangling the Web** http://www.lehigh.edu/~tjf0/cause95/cause.html
- **Taking Advantage of WWW Technology** http://www.duke.edu/~savage/cause95-summary.html

This track presentation will feature significant present and future WWW development activities at both Duke University and Lehigh University. Taken in total, the session will provide a broad glimpse at creative use of World-Wide Web technology.

In *Untangling the Web* , Tim Foley and Roy Gruver from Lehigh University will discuss how Lehigh University has centrally *supported* the implementation of web-based services. Lehigh's goal is to simplify the development effort for campus constituents, to provide a broad range of information services, and to create a positive external image for the institution.

In *Taking Advantage of WWW Technology*, Bill Savage from Duke University will discuss how Duke has used WWW as an application development environment. Bill will discuss the advantages and disadvantages of the web and will describe how web technology can be used to implement sophisticated client-server database systems.

Presenters:

Timothy Foley, Lehigh University *(tjf0@lehigh.edu)*
Roy Gruver, Lehigh University *(rag3@lehigh.edu)*
Bill Savage, Duke University *(savage@acpub.duke.edu)*

Untangling the Web: Taking Advantage of WWW Technology

Timothy J. Foley
Associate Director, Computing Center
Lehigh University
Bethlehem PA 18015
tjf0@lehigh.edu

Roy A. Gruver
Director, Administrative Systems & Telecommunications
Lehigh University
Bethlehem, PA 18015
rag3@lehigh.edu

ABSTRACT

For many, the World Wide Web is a place of amazement and confusion. Like a spider's web, its architecture is flexible and its applications are often intriguing and picturesque. Also like a web, applications can exist without much substance, they are sometimes sticky and difficult to navigate, and the links may not connect important parts. Lehigh University has exerted great effort into untangling the Web and using it as an interactive presentation environment. This environment presents a coherent institutional image and a broad range of campus services. A key focus of this paper will be the effort Lehigh has placed on "marketing" WWW: demonstrating its ability to support relatively sophisticated departmental and institutional applications; providing campus-wide development training; involving key departmental partners in early development; and providing institutionally-funded development assistance.

INTRODUCTION

The past half-decade has profoundly transformed the way campuses have conceptualized and developed information systems. The rapid acceptance of client/server technology, and specifically the World-Wide Web (WWW), has created a new paradigm for campus information systems.

Lehigh's implementation of World Wide Web (WWW) technology began as a grass roots effort by a few departments. This seminal effort bore positive results. It also made Lehigh realize that a concerted effort to build university-wide consensus on the value and usefulness of WWW was needed. The process at Lehigh has been referred to as "controlled anarchy", but the results have been impressive -- the entire campus-wide information system is available on the Web; student prospects can "visit" campus through the online Viewbook; numerous departments have created their own homepages with some even running their own WWW servers; hundreds of Alumni have registered under the Alumni homepages and an Alumni Web page has been created to link Alumni to Lehigh from around the world. The Admissions Office feeds web-based prospect information forms into the institutional admissions and recruiting system; Administrative Systems is developing an Oracle-based "virtual kiosk" application for the Web; and they are also working on improving information flow throughout the campus using the online forms facility.

EVOLUTION OF LEHIGH'S INFORMATION ENVIRONMENT

Over the last five years, Lehigh's information environment has been shaped by Lehigh's five-year strategic plan for computing and communications. This plan was developed in the spring of 1990 and finalized in the fall of 1991. The plan called for movement from a mainframe environment to a client/server environment utilizing the Unix operating system. It also stressed the importance of providing high-speed network connections in every office, classroom, and residence hall room. The removal of the mainframes was accomplished with the installation of over 150 IBM RS/6000 computers including numerous high end IBM RS/6000 computers to act as Compute, File, and Communication Servers for the campus. Campus high-speed network connections have been evolving rapidly with major projects underway to connect all offices through the recently established Network Support group. High-speed network connections are also available to all students living in residence halls through the WIRED (World-wide Information Resources in Every Dorm) project which is a joint undertaking of the Computing Center, Telecommunications, and Residential Services. An extensive network of publicly available microcomputers is also in place to provide web access to everyone on-campus.

Another positive force in the evolution of Lehigh's Information environment is the merging of the Information Technology groups with the University's Libraries. Lehigh has always had a strong working relationship between these two groups with many joint ventures such as the installation of over 75 networked microcomputers and workstations in the libraries; the installation and system support of an IBM RS/6000 running the OCLC's SiteSearch software and the Library's web server; the implementation of many tools on Lehigh's Campus-wide Information System that were directly inspired by Library requests; and the installation and support of the Library's on-line catalog system running on an IBM RS/6000.

Transforming Lehigh's Campus-wide Information System (CWIS)

Another important element in untangling the web at Lehigh is the transformation of Lehigh's CWIS interface to a web-based interface. Lehigh's CWIS provides the following applications to a user base of over 7000 active accounts: electronic mail, bulletin board and conferencing facilities, access to national and international networks, on-line forms and survey processing, fax delivery and retrieval, and an access point for library services. This system is widely used by the campus with 95% of the community using the system on a regular basis. Lehigh's CWIS was developed in 1986 running on an IBM 4381. The IBM 4381, which utilized flat ASCII files that were accessed by a control file, was replaced in 1991 with a distributed database model using the Oracle database management system on a cluster of RS/6000s. This distributed database model has allowed the client portion of the CWIS interface to run on the web. Faculty, departments, and campus groups automatically have their existing information placed on the web through this Oracle/WWW interface. Lehigh is currently working on improving this interface to provide many of the same features as Lehigh's current terminal-based interface.

Lehigh's CWIS was based on the concept of "distributed services" encompassing both the management and the location of databases and applications. The "distributed services concept has been easily adapted to Lehigh's WWW. Data management activities are the responsibility of individual topic coordinators. Topics on the CWIS are fed, nurtured by, and are the responsibility of, the information providers. Another aspect of distributed services is the ability to access designated hosts or data directories. Distributed applications are being explored to allow selected applications to transparently run on another host. The overall goal is to move from a tightly coupled cluster supporting all available services to a more diffuse system.

MARKETING STRATEGIES

The marketing of the web to the campus community has been an integral part of Lehigh's web strategy. The first web server was setup in November of 1993 with the Computing Center developing a prototype which included an image map (see figure 1) and links to each of these areas. The prototype also included an interactive campus map utilizing scanned images of the campus which were already in place on Lehigh's gopher server. Parts of the existing University catalog were scanned in to develop a hypertext version of the catalog. On the lighter side, experimental pages were developed such as the "Chocolate Milk Guide of the Lehigh Valley" with milk ratings in terms of number of cows. The Computer Store established a web server with a video camera to scan the store's main floor every 60

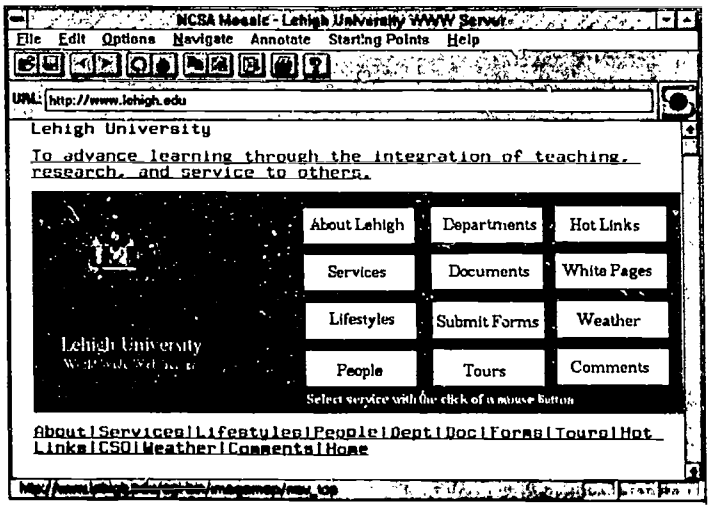


Figure 1 - Original Lehigh University Homepage

seconds and display the image on the web so customers can check if the store is busy before coming to the store. Once this was done, the University's Media Relations group became involved with design suggestions and also by placing on-line many of Lehigh's current publications.

To spread the word of WWW capabilities and provide training to the campus community, a number of initiatives were undertaken. Web seminars were added to the Computing Center's seminar schedule. Separate faculty seminars were also developed which included the use of a digital camera to get faculty pictures on-line and placed on their individual homepages. From the outset, Lehigh's web initiative encouraged individual homepages and also encouraged the linking and creation of homepages for Alumni. Departments were given individual presentations on the value of the web to both the internal functioning of the campus and as a mechanism for presenting Lehigh to the outside world. The President's council and the Board of Trustee were given presentations on the uses of the web at Lehigh and around the world. Special training sessions were given to the Library Staff who also setup their own web server which is linked from Lehigh's main page. This summer, special seminars were given to train one person from every administrative office on the development of HTML documents. These seminars were requested by the Administrative Vice President.

Policies, standards, and guidelines were developed for the web. The issues associated with computing policy are magnified by the ease of use and access to world-wide information through the web. The extensive use of graphics, video, and sound caused concerns about copyright issues and also network bandwidth issues. Policies were in place to manage these issues and are placed prominently on-line. At Lehigh, information providers agree to abide by these policies when web accounts are issued to departments or campus groups. Individuals who create their own homepages must agree to abide by Lehigh's policies before being given an account to create a homepage.

Style guides and suggested formats are made available on Lehigh's web, but departments are on their own to design and develop what they want. The Computing Center decided at the start that departments and groups would be responsible for the design and creation of their individual pages with training provided by the Computing Center. Student Web Authoring Teams (SWAT) have been formed to work with the Computing Center staff and also with individual departments to develop and implement homepages. The departments actually pay for these students out of their own budgets once the students have received training through the Computing Center. A perfect example of this is the work done by one of the SWAT members to place the current University catalog on-line. The SWAT student recruitment process revealed that many students wanted to become involved with creating HTML documents, and the Center had to turn away many qualified students.

Another part of marketing was the tracking of usage and monitoring of comments in Lehigh's guestbook. The usage statistics reveal the global nature of the web, especially when they show that the University's homepage is being accessed by people from around the world on a regular basis. Comments to Lehigh's guestbook such as:

"Lafayette Class '93. Lafayette 52 Lehigh 20 better luck next time....if it is any consolation your WWW page is the best I've seen"
"Just a senior from Menlo Park, CA , interested in possibly attending Lehigh in the fall of 95. I would appreciate any information you could send me."
" '74 Alumnus. If this stuff had been around when I was there I may have finished my engineering degree instead of switching to Biology"
"Grad Student from Berkeley. I almost went to Lehigh now you have a better homepage than my Alma Mater. Cal."

have been used to illustrate to the University the diversity of the people viewing Lehigh's homepage. It also should be pointed out that Lehigh's guestbook, and also student's individual homepages, have been abused at times and Lehigh has moved from a rather open policy concerning postings to a more controlled environment. For example, last October when the Computing Center was presenting an overview of the web for the entire Administrative wing of the University, looking at the guestbook produced two images of Barney along with a couple links to dating services. Students have posted inappropriate materials on their homepages and have been referred to the Dean of Students for disciplinary action. The entrepreneurial students at Lehigh have also gone into the homepage business to advertise for companies for a charge and they have also set up server sites for distributing adult materials. The student homepage and server issues are covered in the University's Computing Policy.

Another marketing strategy was the involvement of the entire campus in the design of Lehigh's current homepage (see figure 2). It was felt that the campus departments would have a sense of ownership if they were involved in the design and implementation of the University's homepage. This page was designed with input from all departments on campus as to its organization and content. It is currently undergoing another round of revisions and will probably turn into a yearly review task. The amount of suggestions about our current homepage were minimal with most ideas coming from the Computing Center and Media Relations. Other departments were still getting use to the web as a tool for marketing the University and their services. Lehigh's next revision should have more input from other departments as they become more familiar with the web.

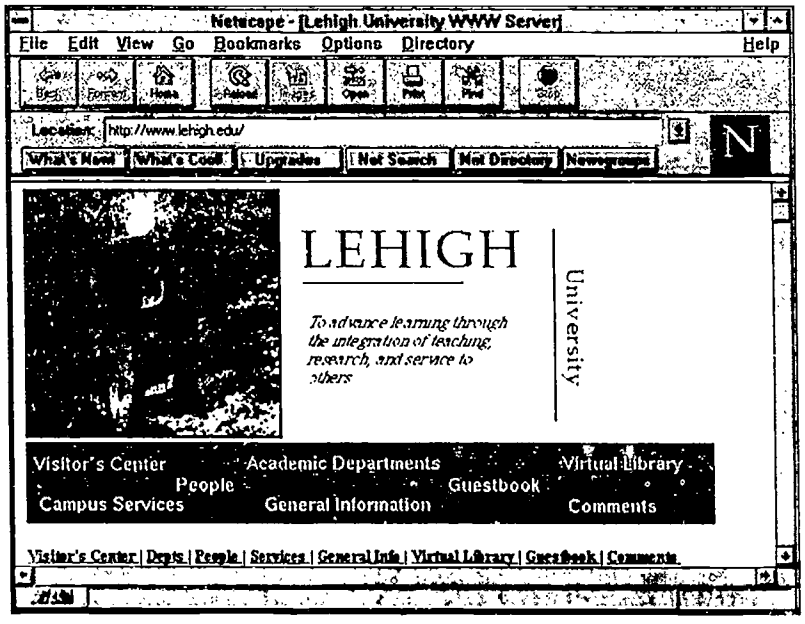


Figure 2 - Current Lehigh University Homepage

And finally, when marketing the web, it is important to get recognition of its value from the highest level in the University as possible. The Dean of the Engineering College has created a homepage for the College and encourage all departments to develop their own homepages. Similar initiatives are occurring at the other Colleges and departments on campus. Lehigh's President has recorded several messages which have been placed at appropriate places on the web. Audio and video players, such as Real Time Audio, are being investigated to provide a better mechanism than is currently available to distribute sound and video. The Center first saw Real Time Audio when the Athletics department made arrangements with Taylor Subscription TalkTM out of Tulsa, Oklahoma to broadcast its football games on the web.

ADMINISTRATIVE STRATEGIES

The creation of virtual kiosks for the University is a high-priority of the Administrative Systems group. The abundance of workstations and PCs connected directly to the high-speed network make the virtual kiosk concept the logical solution for access to student, faculty, and staff information. Students should be able to access their own personal and academic information from anywhere on-campus, especially from their own rooms. Faculty and staff should also be able to access their personal and payroll information through web-based screens. Administrative Systems has explored several different implementations for Lehigh's virtual kiosk and believe that a web-based approach has the most promise. To this end, a number of pilot applications have been developed to demonstrate the flexibility, simplicity, and richness of the web as an application development environment. People from outside the University should also be able to request and receive information. Web-based software that automatically generates mailings and responses is currently being developed. For more than a year, web-based requests for general and admission information have been received from interested students and parents. Further, the web is being used for a repository of operational information and the forms facility has been used to simplify on and off-campus communications.

SUMMARY

The word "web" no longer evokes thought of spiders on most college campuses and one can hear references to it on a daily basis in news and television media. As the web and the Internet have continued to grow at exponential rates people are just beginning to determine the many ways that they can use these new technologies. Cooperation between all departments on campus is essential for providing up-to-date information for both on and off campus users. Training and marketing of the current and future capabilities of WWW technologies is a necessary and daunting tasks that needs to be done to allow campuses to fully exploit the web.

3-8-8

Using Web Technology for Client-Server Database Systems

Bill Savage
Duke University
Durham, NC
savage@acpub.duke.edu

Abstract: *Designing, building and supporting client-server database systems is usually a complicated and costly process. Choosing an architecture, tools and vendors can involve considerable risk. World Wide Web (WWW) technology offers an attractive alternative to conventional solutions by providing a simplified and open architecture, often at considerably reduced cost.*

Conventional Architecture

The typical client-server system attempts to enhance productivity and provide timely access to important data by providing a sophisticated graphical user interface (GUI) on the user's desktop. These systems generally provide database access (both inquiries and updates) for client computers which are most often PCs or Macs. Client-server systems utilize a separate server machine, usually running Unix or Windows NT, on which is placed a relational database management system (DBMS). Sometimes a three-tier architecture installs a portion of the application on a third computer positioned logically between the client and the database server.

The client-server architecture usually is based on the bulk of the application residing and executing on the client machine, interpreted and/or executed by proprietary runtime software; PowerBuilder is a prominent example of this approach. Database access is often provided by use of calls to the ODBC library, which are then mapped to an ODBC driver specific to the client platform and DBMS. A DBMS-specific client software layer (such as Sybase Open Client) manages the database access, in conjunction with the network protocol software on the client (such as TCP/IP).

Although this architecture is widely used and supported by many vendors, it introduces issues of complexity, cost, and risk. Some areas of concern are:

- *Vendor selection.* It is difficult to select from among the many vendors of development tools. The tools and languages are generally proprietary and costly.
- *Complicated installations.* The several layers of required client software complicate the process of system testing, installation, and support.
- *Cross-platform support.* Many client-server development products only support PCs running Windows, which is problematic in heterogeneous environments.

Web-Based Architecture

A client-server system can be implemented using a Web architecture. This architecture consists of a web browser (such as Netscape Navigator) running on the client machine, in conjunction with the TCP/IP network software. A web server package (such as Netscape Commerce Server or

NCSA HTTPd) runs on the server machine. The server also includes application programs, implemented in a standard language (often Perl), which are connected to the web server through the Common Gateway Interface (CGI). These programs access the database residing on the server, and dynamically create the graphical interface using HyperText Markup Language (HTML). In a three-tier architecture, the web server and application programs would run on an intermediate machine separate from the database server. Using this model offers these advantages:

1. *Simple architecture.* Because this model is significantly more simple, system installations and updates become easier and less costly.
2. *Platform independence.* The application code is independent of the client platform, as is the language (HTML) used to create the graphical interface. Virtually any client type can be easily supported without system modification.
3. *Standard languages.* Several different standard languages (such as C, C++, Java, Perl and Tcl) can be chosen for the application, thereby removing the dependence on and special learning required for vendor-proprietary languages.
4. *Low cost.* A sophisticated client server system can be built using free software on both the clients and server, including even the DBMS.
5. *Internet Access.* Use of a web-based architecture enables the creation of a system which can be accessed over the Internet. This can be valuable for systems which rely on public usage, or for which remote access would otherwise be difficult.

There are potential disadvantages in some situations when relying on a web-based client-server architecture. Additional planning and expense might be required to address security concerns. The tools and standards are still evolving, and it may not be possible to create a database interface as robust as one created using conventional technology. Because web-based systems are relatively new, it is often difficult to find experienced assistance and good training.

Summary

It is possible today to implement a sophisticated client -server database system using web technology. This web-based approach addresses some of the significant problems that can make conventional client-server development difficult. An increasing number of organizations have successfully adopted web technology in a variety of client-server systems, often with significant cost savings. Web server solutions will continue to gain acceptance as the technology, tools, and standards evolve, and the advantages of this approach become more widely known.