From Mainframe to Client/Server - a Success Story

Stephen Patrick - Executive Director of Information Management Services, Bradley University
Ellen Watson - Associate VP, Information Services/ Dean of Library, Indiana State University

Bradley University, Peoria, Illinois
Bradley University is an independent, privately endowed, coeducational institution. Bradley’s enrollment is about 6,000 students and offers programs in Engineering, Business, Liberal Arts, Sciences, Education, Health Sciences, Communications and Fine Arts.

Abstract:

Bradley University completed the 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 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
From Mainframe to Client/Server -
 a Success Story

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 Corporation records 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, since in 1990 there were limited examples of client/ server architecture applied as we envisioned.

We recommended a phased implementation. The highest priority was the Advancement system, as 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 in-house 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 investigated 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 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. Advancement staff were delighted with the system as designed, and 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.

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 product 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 data was 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. Cross platform integration
5. Workstation configuration
6. 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, complex administrative system with a large user base. Our Advancement database consisted of 70,000 records for 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 Information Management Services (formerly Computing Services) print most reports. The Controller’s Office prints to a high volume laser printer in their office. Admissions prints most of their work on high volume laser printers, but use data center impact printers for some reporting. The Registrar’s Office uses data center printers for most output and has an office laser printer for transcripts and time sensitive reports. 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/ multitasking systems making background processing of reports a routine task. Desktop computing,
especially in a DOS environment, is a single tasking system, 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 on their “own” desktop machines. 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 database has made selection and reporting significantly faster in client/server environment, particularly for small volume reports. We provide user-controlled parameters to specify the amount of resources to allocate to foreground processes, so the user retains control of the desktop. While this situation is not ideal, it is a Windows limitation that we must accommodate.

Auxiliary Applications
While several of the systems must support auxiliary systems in a client/server environment, the Student Records system has the largest number and variety of auxiliary subsystems, including:

- 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 communicated to the mainframe through a single serial line. While we could have written 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 requires that the client/server database management system must be able to support both personal computers and UNIX systems.

- The mainframe student records system audited 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 time available. We purchased the DARS system from Miami University to replace our in-house degree audit system. DARS is an IBM mainframe, CICS-based, system available in COBOL or PL1. We initially thought we would build our own frontend to DARS, then just execute the COBOL code to actually produce an audit. We found this another difficult challenge. We now 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 resources to maintenance of the audit rules database. This appears to be the first client/server implementation of DARS, and we faced significant challenges making it work effectively. We plan to add DARS execution to our Web based Academic Inquiry system.

- 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.
We are investigating Call Center processing, where incoming calls generate a database query. The PBX sends the incoming phone number to a server. The call can be routed based on the source and the individual answering the phone is supplied with information about the caller.

Cross Platform Integration
Bradley University’s computing environment is a diverse mixture of Intel-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. We addressed this challenge by standardizing systems and configuration in the mainstream user office (the Registrar’s Office), and providing Web access for all other users.

The Web provides several advantages over other client/server architectures:
1. Platform independence. Bradley University has a wide variety of desktop computer systems including Macintosh, PC Windows (several versions), and UNIX workstations. The Academic Inquiry system runs on Netscape versions 2.0 or higher. This is supported on nearly all desktop systems on campus. The only problems are due to obsolete computers, or lack of networking.
2. Reduce training requirement - most users are familiar with the use of the web browser. The Academic Inquiry system looks like all other table based Web pages. The client needs to know only the terminology and how to push buttons.
3. Enhance security IF PROPERLY DESIGNED. Because we are very concerned with security, we prohibit Academic Inquiry access from off campus, the modem pool, computer labs and residence halls. Also, Web users are not logging into a system thus cannot shell out or wiggle through a security hole to gain access to the whole system. The Registrar maintains a table granting specific access to specific users.
4. Reduce installation, setup and troubleshooting costs. Our client/server applications require a very specific installation. Often, installation of new software will break our client/server applications. Opening access to a fragile system by a wide audience invites support problems. This problem is avoided using Web tools because we only need to be concerned with Web server, not the client.

The Web presents some disadvantages that should be considered:
1. It is more difficult to develop in this environment because database tools are immature. It takes more work to develop a Web based application than the same application in client/server architecture. We do not yet have access to easy-to-use database development tools for the Web.
2. There are potential security problems to be addressed. The standard Web browsers cache information to the clients hard disk. Thus each Web user could have 5MB of sensitive data on their computer they are not aware of. This is available to anyone that has physical access to the hardware. We avoid this problem by turning off the caching of data from the server - which is why we only support only Netscape 2.0 or greater. We also time-out users, as in our host based systems.

Workstation Configuration
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 -- but allocated no time to the process while minimized.

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

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

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, as 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 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 or mainframe environment, the benefits associated with the client/server architecture are 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 had been a several year lag between the time the mainframe began to provide unacceptable performance and the time we upgraded the mainframe, due to the high cost of upgrading. 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.

Unanticipated Outcomes
Putting control of the systems in the hands of our clients, and our Rapid Application Development (RAD) methodology led to several unanticipated outcomes.

There was a workload shift in client offices. While our new system is more efficient than the old, initially (the first year), workload increased in client offices. For example, we changed to on-line transcripts from paper copies. All information that was formerly typed or hand entered on the paper transcripts needed to be placed in the computer record. The benefits of the on-line transcript will only be realized after we complete a massive audit of our data base.
In general, the complexity of our systems have increased. Over the long term we expect a decrease in low-end clerical support, but an increase in the need for professional support (knowledge workers).

RAD methodology is ideal for providing the client with the system they want. The challenge is finishing the project. At the beginning of the project, our clients may not have a good idea of what they want and need. As we present the possibilities, clients begin to express their creativity which we incorporate into the system. At some point, we need to stop developing and implement the system.

Another potential problem with RAD is ensuring that the right individuals are involved. If the clerical support staff directs the project, the resulting system will be ideal for the support staff, but may not further the strategic objectives of the institution.

We experienced a leadership change (and corresponding change in strategic direction) in the Enrollment Management unit in the middle of the development of the Admissions system. This added to the length of the project.

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 -- Phased implementation began June 1996
- Mainframe turned off -- September 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 ... the list is endless.

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.