Client/Server Technology: The Agony & the Ecstasy

Lessons Learned From a Pioneer

Who's Serving Whom?

Lessons Learned From a Pioneer

by John Glaser

Brigham and Women's Hospital, Boston, has been living with a client/server architecture for five years, developing one of the largest healthcare client/server implementations in the world. This daily exposure to the architecture has taught us a lesson or two, many of which were painfully learned.

The term "client/server" has become horribly abused and in some ways has lost meaning. Client/server is a class of architecture that has several characteristics:

- Applications processing is meaningfully distributed (not simply split between GUI and everything else) between at least two processors - the client that controls the application processing and a server that provides applications services requested by the client, e.g., database services.
- Servers can communicate with each other specifically to fulfill client requests for services a particular server cannot satisfy.
- A client has access to all services on the network.

From this set of characteristics, we can make several important observations.

- Client/server does not require specific technologies, such as SQL or UNIX. Meaningful distribution of an application's processing can be accomplished via a wide range of technologies. The presence or absence of specific technologies cannot serve as a litmus test of "client/serverness."
- Because client/server is a class of architecture with diverse specific implementations, one should expect wide variations in costs, performance and manageability experiences with client/server systems. Similarly, one finds diverse weights and intelligence among specific examples of animals, or mammals.
- The experiences, successful or otherwise, of one client/server implementation have no inherent relationship to the experiences at any other site. One organization's implementation of a particular vendor's approach to client/server does not mean that another organization's implementation with a different vendor will have the same difficulty, experience the same costs or achieve the same outcome. This is no different than saying the experience of one organization implementing a timesharing architecture would be the same as another's.

Client/server architecture is a software phenomenon. We talk about the architecture in terms of hardware boxes. However, the ability of a client/server architecture to function well is fundamentally attributable to the system software infrastructure, not the processor.

Client/server systems are not panaceas. They can be very helpful in achieving organizational goals but they cannot solve all information ailments any more than case tools, open systems or relational databases can. Consultant or organizational recommendations that say "we should move to client/server" without a lot of thoughtful "whys" are dangerous, and demonstrate malnourished thought.

The fundamental value of client/ server architecture is its inherent ability to enable an application, and an organization's aggregate applications, to accomplish more work in a finite period of time. Distributing processing among two or more machines, like distributing the task of stacking wood among two or more people, allows more work to be done in a period of time than if one processor or one person was given the task. We often use this work capacity to handle the demands of a GUI. Accomplishing more work in a period of time does not inherently mean that an application's response time is faster any more than having four people stacking wood implies that each individual stacks faster.

In general, client/server can help an organization achieve a goal of enabling more application software work to be accomplished. The increasing complexity of vendor-supplied

software, and organizations' needs for systems such as clinical information systems, will require that technical platforms enable the accomplishment of more work. When client/server systems are implemented on inexpensive hardware and systems software, the organization can achieve inexpensive scalability, particularly for medium or large scale platforms, and take advantage of the third party activity that surrounds commodity technologies such as UNIX and Windows.

Most of the issues associated with migrating to client/server architecture are issues that would be encountered by an organization moving to any new architecture. The systems software may be embryonic; well-developed system management strategies and tactics may be scarce; technology staff must master new concepts and skills; conversion of legacy systems is hard; thoughtful development of the organization's architecture is critical and absorption of the technology and architecture requires management skill and attention. These issues confronted those who moved to time-sharing systems and will confront those who are planning to move to object-oriented databases and development environments.

Client/server architecture is becoming a well-developed and pervasive approach to computer system architecture. We hope the lessons we learned as described above will ease the inevitable pain of making this transition.

John P. Glaser is V.P. and CIO for Partners HealthCare System, Brigham & Women's Hospital Boston.

Who's Serving Whom?

by Vincent Ciotti & Fred Zodda

This year's HIMSS conference opened the floodgates for client/server systems in the HIS industry. Every other booth was festooned with banners claiming the embracing of client/server technology by both mainframe mavens, as well as minicomputer stalwarts. Sales reps carried buttons emblazoned with "client/server" in flashing LEDs, outside street vendors hawked client/server wristwatches, while at the airport, TCBY featured client/ server frozen yogurt! HIMSS' keynote speaker was none other than Bill Gates, author of the least user-friendly program in history (DOS), recently reincarnated as King of the GUIs now that he offers healthcare a "Window" to the client/server world.

What does all the client/server hype mean to today's healthcare CIOs? Should one rush out and commit scarce dollars to this new approach, or play a waiting game and RISC being the last on your block to acquire the hottest technology this side of the Internet? This article will attempt to sort through some of the "marketechture" surrounding client/server systems today and offer some practical guidelines to healthcare facilities whose operating budgets are under more pressure to cut costs than Microsoft was to deliver Windows 95 before '96.

Historical Precedents

Many readers may not be old enough to remember, but the basic concept of client/server systems was actually launched several decades ago by long-gone HIS pioneers:

EDS - Ross Perot's original company, prior to its acquisition by GM, had a healthcare division in the '70s that marketed a revolutionary Patient Care System that competed heavily with the likes of then-dominant Technicon and Datacare. At its heart was the concept of "Distributed Data Processing," the technological answer in those days for "Model T" IBM 360 mainframes. EDS used a custom terminal with buttons on the side of the screen to help keyboard-shy nurses and physicians overcome the dreaded QWERTY keyboard. Each CRT contained a microprocessor that performed some of the application's duties locally to spare a few cycles from the wheezing mainframe.

Sentry - This successor to minicomputer-pioneer DATX Corp. in the early '80s performed similar data processing legerdemain with its "Distributed Database Controller," or DDC, which helped the Tandem CPUs live up to its Non-Stop moniker while processing hundreds of simultaneous disk reads and writes. More than a dozen hospitals installed Sentry, and although they may have lamented the firm's financial demise (first sold to Control Data, then 3M), users were unanimous in their praise of the rapid response times during a decade when waiting for a minicomputer's response was like waiting for Godot.

Why didn't such early efforts catch on? There were none of today's cheap and ubiquitous PCs enabling vendors to use off-the-shelf hardware to build these distributed processing prototypes. Even IBM and Apple's earliest machines, circa 1982, were ludicrously under-powered to ever be an active part of an HIS. It took the introduction of real CPUs like Intel's Pentium and Motorola/IBM/Apple's "Power PC" family before PCs had sufficient processing cycles, about 100 Mhz, to play pivotal roles in large system architecture.

Semantic Confusion

One of the biggest challenges with client/server systems today is defining just what they are, and are not. The term is used to describe a whole host - forgive that pun - of systems:

- Networks - any LAN or WAN that has PCs connected to servers.
- GUIs - systems using a Graphical User Interface front end.
- Emulators - systems that can utilize a PC as a terminal emulator.

In truth, these concepts were with us years before George Forrester coined the phrase "client/server" in 1987. As if it isn't confusing enough, people use such terms interchangeably with client/server, and clever vendor marketing gurus intentionally drop the phrase as loosely as other hot buttons like "seamless" and "Enterprise." Indeed, as a litmus test for true client/server systems, consider this definition from Anderson and Armstrong of Deloitte & Touche:

"Client/server is an application design concept. An application is broken into two components. One part of the application, the server, provides data or services when asked. The receiving component of the application, the client, receives data or services for its own local use."

That's something almost normal beings can understand. We hope it deflates the sales hype that claims any use of Windows on a PC as a front-end to an HIS system suddenly makes it "client/server."

Some "True" Client/Server Systems

Rather than debunk abuses of the term, let's give tribute to some of the client/server pioneers who deserve far more credit than the recent "me-too" crowd:

PeopleSoft - As its name implies, originally a Human Resources, or HR, and payroll vendor when formed in 1987, now expanded to include a host of applications, including Accounts Payable and General Ledger. This Walnut Creek, Calif.-based vendor has long dominated HR and payroll systems in other industries and recently has made many sales in healthcare as well.

HMDS - Health Micro Data Systems, recently purchased by Citation, started in the early '80s when President Frank Poggio first dreamed up the idea of building an HIS on IBM PCs. Granted, his early network was "sneaker-net," and only when it became Novell-based did sales and acceptance take off, but the system deserves credit for truly running applications on PCs and servers, since it never had minis or mainframes connected to it, unlike most recent HIS "wannabes."

Client/Server Advantages

Any readers who recently traded up to Windows from DOS, or to Windows 95 from 3.1, know all too well the power of "point and click" computing that Macintosh users have enjoyed for years. True client/server systems combine the best of both worlds: the user-friendliness of icons, windows and pulldown menus, combined with the near-limitless storage of today's servers, with mass storage now running about 50¢ per meg, removable media like magneto-optical drives squeezing a half-gig on a 3.5 inch floppy, and CD-ROM writers now priced under $3,000. Perhaps most crucial is the far greater access to data client/server systems offer, with users' PCs not only accessing data stored on servers, but processing the data how and when they want to.

Client/Server Disadvantages

Cost - The president of one client/server firm recently said it can cost from six to seven times the license fee to install his systems (software and install fees are about equal in most HIS systems today). Much of the excess goes to outside consultants who assist in the extensive re-engineering of work flows such technology requires, which may explain the popularity of client/server systems among the RFP crowd.

Time - A survey of 400 CIOs revealed that those who believed client/server systems significantly reduced the time needed to develop applications dropped from 46% last year to 33% this year. Splitting applications between servers and clients does take more programming time.

Performance - Pundits knock "legacy" mid-range and mainframe systems, infamous for their history of response time problems and interminable month-end down-times. But, according to Avery Jenkins in his article "The Performance Conundrum" in *Computerworld Client/Server Journal*, April, 1995, "Performance management and capacity planning tools . . . are virtually non-existent for client/server systems." Some scalability.

Controls - All those ancient "legacy" systems with their archaic batch controls over input. What we need are systems where we can down-load a file, make changes on the fly, upload it, print a management report, make more changes, try another report and send one to management.

In closing, we offer a few simple rules to guide healthcare CIOs through the maze of marketing hype and Madison Avenue hyperbole about this brave new world:

1. Watch the semantics. Don't let glossy brochures or slick demos convince you that an HIS is client/server because it has a Windows front-end or uses PC terminals. Look beyond the surface to see if all or part of the application programs actually run on the PC/client, or you may be buying the sizzle instead of the steak.

2. Serve your clients. What good would it do an R.N. or M.D. if your HIS was converted to client/server tomorrow? It might make your life more interesting and advance the IS staff into '90s technology, but if the functionality doesn't improve ancillary department technicians' productivity or facilitate outcomes measurement for managers, what good is the technology for the hospital?

Client/server is surely the hottest "paradigm" in IS and undoubtedly represents the wave of the future. However, to help your facility survive to be there, a prudent CIO will scrutinize these "emperor's new clothes." After all, who is serving whom? M

Vince Ciotti is a Partner with HIS Professionals, a management consulting firm headquartered in Boonton, N.J. Fred Zodda is a Partner in the firm's Framingham, Mass., office.

Reprinted by permission of *healthcare informatics*, Vol 12, No 10, October 1995.

Call for Articles

The April issue of *M Computing* will focus on Object Oriented Systems. If you have experience with M and Object Oriented technology and would like to submit an article for publication, please send your ideas to:

Managing Editor, M Computing
M Technology Association
1738 Elton Road, Suite 205
Silver Spring, MD 20903-1725