Schussel's
DOWNSIZING JOURNAL
May 1993

In This Issue:

Doing Client/Server Right—The First Time

Part I of II

Richard Finkelstein
Performance Computing, Inc.

This article should actually be called *"The Void in Client/Server Development."* In my experience, I have seen more projects aborted than successfully deployed when client/server development is involved. There are, of course, many reasons for such a high failure rate. I am certainly loathe to blame it on people or lack of education because I don't believe that either are the problem at this point in time. The problem is the technology involved—even if you have a very astute group of people,

(continued on next page)

Rules for Downsizing Success

As part of my job, I have had the opportunity to talk with people from various companies involved in downsizing their corporate systems. While some of these conversations involved cases that didn't work as well as expected, most people have told me about downsizing scenarios that have been overwhelmingly successful. In reviewing such cases, I always look for the common threads that seem to be good predictors of success (or failure!).

Although there exists good technical advice for downsizing, most success stories I've seen are the

(continued on page 7)

Doing Client/Server Right...

(continued from front page)

they are not going to succeed if the technology used is too immature or just wrong. This article explains what the problems in the current state of client/server hardware and software technology are, and how you can avoid the pitfalls in building a solid foundation.

Architectural objectives—homogeneity

An important architectural objective for any client/server system is a uniform, supportable environment. Uniformity and homogeneity of products is very important. If you don't have homogeneity among your hardware and software, it's going to be almost impossible to support an application over a long period of time. You need a stable foundation so that when problems arise, you know either how to solve the problem or have a methodology with which to solve the problem. Currently in the heterogeneous world in which we live, there are so many variables involved with fast-evolving and changing products that it is almost impossible to isolate and replicate problems once they occur. What is commonly happening with client/server systems is that developers are spending most of their time trying to stabilize the environment and not developing applications. An analogy for anyone with a mainframe is this: picture yourself spending 50% or more of your time trying to fix MVS, and not spending any time at all on developing code for applications.

Clearly with mainframe and Unix tools, there is a degree of stability in the operating system and the hardware which allows time for application development. In the client/server world, that stability isn't there yet. Because there are too many vendors involved, when a problem arises, every vendor points the blame elsewhere. That is the natural outgrowth of a very heterogeneous environment.

The goal in designing a stable client/server system is to eliminate this heterogeneity. We've got to mimic the mainframe world in the way it was designed for the multi-user world. This means when possible, buy your products from one vendor. Because once you have the capability of fixing problems, you will have some predictability in terms of how long applications will take to develop. The LAN administrator will also gain the ability to maintain predictable up-time and down-time once applications are deployed.

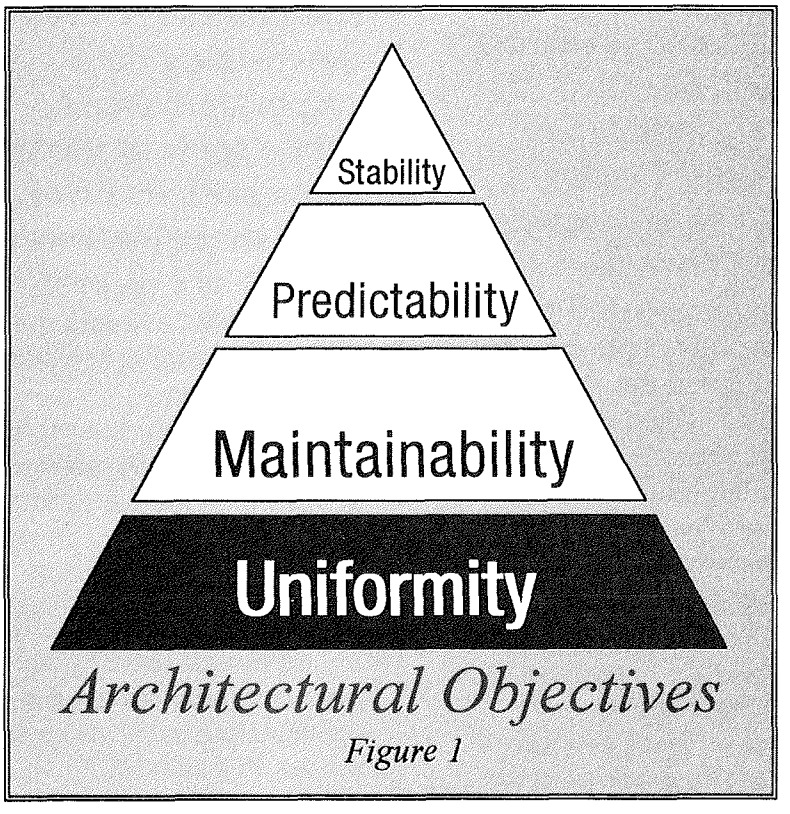


Figure 1

The worst possible situation is to have a mission critical application deployed on a client/server system, and have a problem occur, and not know how to respond in fixing it. Is it Microsoft's responsibility because it might be a Windows problem, Novell's liability because it's a network problem, or IBM's duty because it's an OS/2 problem? Listen people, you don't really want to be doing client/server development in this ad-hoc manner seven days a week, fourteen hours a day—that is no way to live. If you have a systems problem on Friday, you don't want to spend your entire weekend trying to find and fix it, and then have to face an angry manager or a user on Monday.

Why go client/server?

So, I've outlined some of the problems and the important architectural objectives of client/server development, and now you're probably thinking, why even bother with client/server? What should my motivation here be? The motivation, I think for most companies, is cost savings—the hardware cost and the software cost. I believe people downsize to a client/server arena for the substantial up-front savings in both hardware and software. These initial cost savings are definitely realizable. However, I caution you, overall client/server projects cost as much as projects developed in multi-user Unix environments. This is because you have additional costs in user and developer training, learning new application development tools, developing more complex applications, and maintaining deployed applications. It's a very complicated systems architecture to administer. This is not to say that the overall purchasing costs aren't going to be lower, because they are in most cases.

...I know of more files that have been corrupted by Windows than by the Michelangelo virus. However, while [Windows] is not a very stable platform, it is going to be almost impossible to avoid....

Where one could get stuck is in the hidden costs. This is an area, where in many cases, people end up giving away their time in order to support the application. One company I deal with supports client/server applications but didn't budget for an administrator as they should have. When I asked them who does the backup at the end of the week, they replied that it was "whoever is the last one out." So, now the motivation is to leave early on Friday so that you don't have to do the backup. You really don't want to give away your time, so budget appropriately for additional personnel. Chevron Canada's IS staff size went from nine to twenty-one when they deployed their client/server system. That increase did not surprise them. From the beginning they understood all of the costs and budgeted appropriately.

Overall, at this point in time, client/server does not necessarily save you money in terms of hardware, software, development, and administration cost. The current benefits I see in client/server are in the area of scalability, meaning that you can easily and relatively inexpensively upgrade your hardware and your network. You can go from OS/2 to NetWare to Unix to a VAX without touching your investment in applications. Such flexibility is *very valuable*. It allows you to start a system very cheaply and upgrade only

(continued on next page)

Doing Client/Server Right...

(continued from previous page)

as necessary. So in this manner, you receive more performance by virtue of the cheap hardware that you are purchasing.

When do you move to client/server?

Lots of large companies come to me and say "When do we know when to move to client/server?" You know it is time when user requirements dictate an intelligent workstation to run applications. Also, it's time when you talk to your users and they say "we need this application" and you cannot run that application without intelligent workstations. If those intelligent workstations are going to service a group of people with shared data, then you definitely have a need for client/server.

At this point in time, when you select the applications that should be moved to a client/server system, you should stay away from mission critical applications. This technology is way, *way*, ***way*** too new. Even though the concept of client/server has been around for about four years, the technology itself is in its infancy. Comparably, client/server is at the stage that the mainframe world was about 20 years ago. The products are being proven, but we're just starting. I suggest that for your first project, you take applications that are not mission critical and are mainly read-only with a low transaction rate and low visibility. The reason for this is that you don't want someone breathing down your neck as you're going through a learning process.

Some of these applications may have to service a group of users. With that type of environment, it may be difficult to tangibly justify the project. What you're dealing with is a read-only application and limited number of users. You should treat it as a proving experience, as a proving application—a low budget application that once moved will bring some immediate benefit to your users in functionality, performance, and scalability. Let it happen cheaply and learn slowly.

You might want to move a large decision support application currently on your mainframe to client/server. You'll save an enormous amount of mainframe MIPS as you bring it down to a micro, which is a very low-risk operation. Without too much to fear, you'll save the company a lot of money, and everyone will be happy. I repeat, stay away from high-transaction applications. I know everyone wants to do it, but there's plenty of time to downsize that stuff. You don't want to do it on the software currently available. There are some companies that have successfully done it, but for every one or two of those stories, there are a dozen or more projects that were unsuccessful. You're not hearing about those unsuccessful ones because practically no one wants to talk about it.

Designing and building your environment—remember homogeneity

I want to remind everyone of Murphy's Law: if something can go wrong, it will go wrong. Let me tell you, in a client/server environment, there are plenty of

chances for things to go wrong. Problems will occur more often when you're dealing with a heterogeneous environment. By heterogeneous software tools, I mean products from many different vendors and project groups, none of which have been designed or integrated to work together. With such a situation, you are the integrator and the testing site for that product mix. Being the only site that has that unique product mix, there will exist a tremendous chance for failure. Avoid heterogeneity in your client/server environment.

I'm not saying that you should go back and revamp your organization, and that all of a sudden everything is going to come from one vendor—that's nonsense. What I am saying is when you architect your client/server application, you have a chance to create a homogeneous, supportable environment, and I say do it. It's going to make your life a lot easier. Every application I've been involved with where customers have followed this advice, when they experienced problems, they knew exactly where to go for help whether it was Hewlett Packard, Sun, Microsoft, or IBM. Those who did not follow this advice were left holding the bag. Sometimes problems never got resolved. So, keep the number of vendors you use down to a minimum.

Hardware specifications

In terms of hardware, a 286 is viable as long as you're doing non-GUI, DOS-oriented work. It's a very stable platform. If you're going into a GUI environment, you'll need at least a 386 machine as your workstation and either a 386 or 486 as your server. 386s and 486s as client machines tend to perform well. Good advice for these machines is to load them up with as much as practical. Currently, 8 Mb of RAM is a bare minimum for GUI applications at the workstation. Begin to plan now, and put as much memory and other resources as you can justify into these machines. You will find that GUI applications tend to eat up these resources quickly.

Again, try to have the same kind of machine from the same vendor at all points on your client/server network. This will make it easier to stabilize your environment. Software is very sensitive to hardware. I've had seminar attendees approach me and say that they ran a piece of software on Brand X and it ran well, and then a few months later they bought the same Brand X hardware from the same company, but their software no longer ran. Same vendor, same hardware, same software—"nothing changed"—but the system didn't work. The catch here is that if the hardware vendor changed the control

(continued on next page)

Hardware Comparison

	Cost	Power	Scalability	Comments	Suitability
i286	Low	Low	Low	Limited life	Non-GUI workstation
i386/486	Medium	Medium	Medium	Very popular	GUI/Server
SPARC/RISC	High	High	High	Expensive	GUI/Server
Macintosh	Medium	Medium	Low	Popular	GUI Workstation
VAX	Very High	High	High	Very expensive	GUI/Server

Chart One

Doing Client/Server Right...

(continued from previous page)

board, the memory chip, the BIOS, changed anything, it will affect the environment. But since you're a customer, you wouldn't know what had been changed. So purchase machines that don't change, and preferably from a vendor who understands the need to have continuity in hardware design.

SUN SPARC RISC stations are very good for servers; they are actually very good as workstations also. However, as a workstation, they tend to be expensive and aren't user friendly. The most successful client/server applications I have seen have employed SPARC RISC machines at both points—client and server.

It is possible to integrate Apple Macintoshes into a client/server environment, but I would suggest for your first attempt that you not use them. The reason for this is that Macintoshes add 25% additional complexity to a client/server design. If you introduce them into your first environment, you will not be able to measure this additional degree of complexity. However, if you design a client/server environment initially and then add Macintoshes, you will be able to measure the impact. The cost of that addition will be visible.

A VAX server and workstation will provide a stable environment, actually one more stable than any of the other platforms I've mentioned so far. But, it is also very expensive. The companies I have seen using VAXs for client/server at both the workstation and the server ends believe in their stability. With that, the administrative costs are lowered because the VAX environment is very mature. It is a very viable option—it is usually only the hardware expense that eliminates this option. *RF*

Editor's note: Richard Finkelstein is consistently one of the most popular presenters at DATABASE WORLD and CLIENT/SERVER WORLD. The reader should keep in mind that Finkelstein is considered the "Don Rickles" of the computer industry. Whereas others may have more sanguine views on the success of client/server solutions, Finkelstein's opinions are very important in the interest of open discussion.

This article was adapted from a session taught by Richard Finkelstein at DATABASE WORLD, December 8-10, 1992. Part II, which will be published in the June 1993 issue of SDJ, contains Finkelstein's views on choosing operating systems for client/server, and what hardware works well with different software. Finkelstein is reachable at Performance Computing, (312) 549-4824.

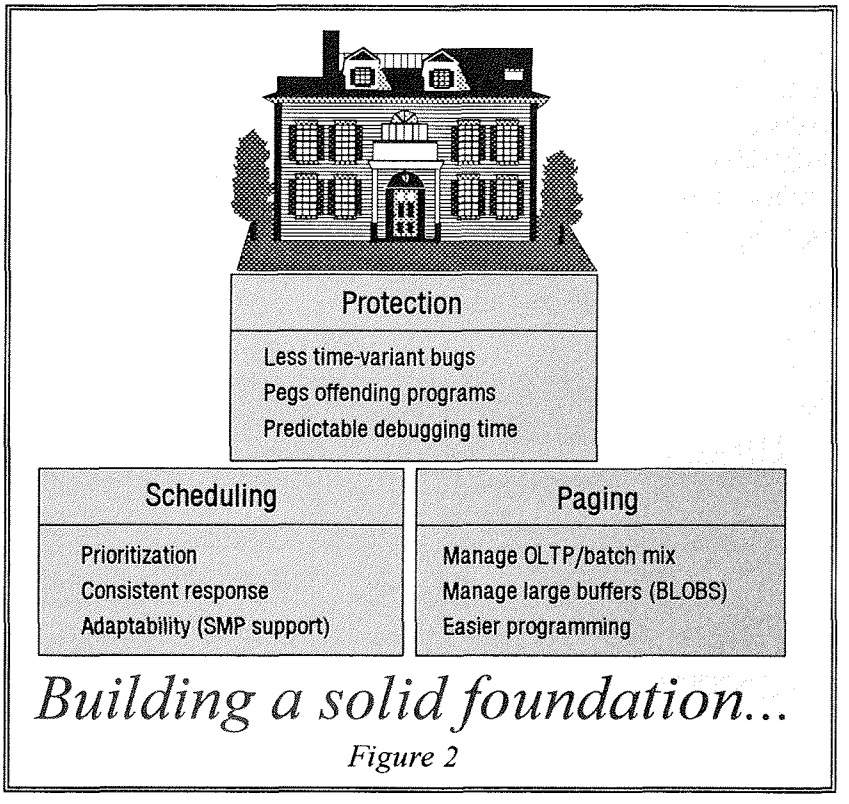


Figure 2

Rules for Downsizing...

(continued from front page)

result of good *management*. From observation, my top four management rules have emerged:

1. Make sure to secure top management support for the downsizing project.
2. Reorganize, refocus, distribute, and downsize the IS department.
3. Staff your downsizing team carefully—include both PC and mainframe people.
4. Use outside help and education including conferences, consultants, and integrators.

Some common technical rules I've seen used in successful implementations include:

5. Modularize applications to machines.
6. Use extra hardware for help in solving problems.
7. Limit the diversity of your installed base.

The remainder of this article will explore these seven rules.

1. Make sure to have top management support

One of the most important issues involved in downsizing is the need to secure the cooperation of both your technical and management staff. As Howard Fosdick of Fosdick Consulting stated at the February 1993 Downsizing EXPO in Chicago, "What we're really doing is downsizing the IS culture. This is a management issue, not a technical issue."

Ron Peri, President of Computer Support, has encountered a similar phenomenon. At a shop where a downsizing directive was issued by top management, and was neither liked nor wanted by IS management, Peri actually found evidence of employee sabotage.

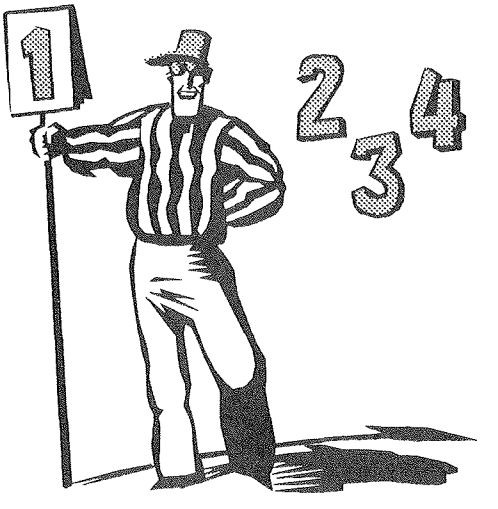
Gaining top management support where "top" is defined as company president (outside the realm of IS) should be no problem. After all, what top executive is going to reject a plan that promises budgetary savings in addition to a data processing movement toward commodity and away from cult status?

Gaining top management support where "top" is defined as Vice President, Information Systems or CIO, may be another matter. Since downsizing implies possible (and probable) staff reductions and reduced IS department budgets, we shouldn't be surprised to expect (the occasional) resistance from this segment.

The front page cover story in the February 17, 1992 ***Computerworld***, was about this issue of IS resistance to downsizing. The article raised the point that *CIOs are losing their jobs* as companies downsize and distribute processing. More recently, the March 1, 1993 ***Information Week*** cover story discussed the same issue:

> *"As companies re-engineer, downsize, and shift IS decision making to departments and business units, many CIOs and other high-ranking IS executives find they've outlived their usefulness...as client/server networks decentralize processing even further and companies lean toward departmentalized IS structures...the CIO position may soon be obsolete."*

(continued on next page)



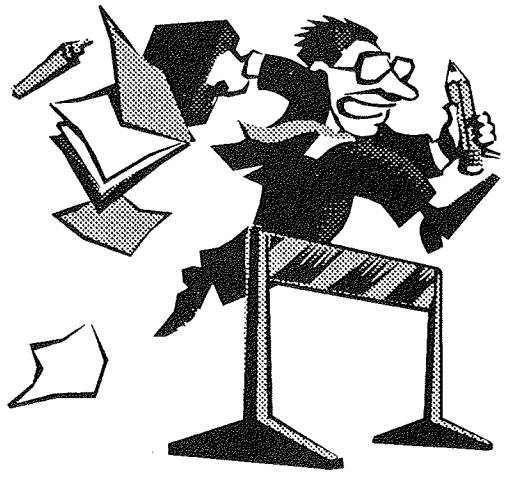
Rules for Downsizing...

(continued from front page)

And in fact, many of the prime movers and motivators behind the downsizing stories recounted in ***Schussel's Downsizing Journal*** (e.g. Foxboro, Echlin) have moved on to new jobs at different firms after their downsizing projects were completed. The kind of person who has the talent to successfully implement the changes implied by downsizing is probably not going to be interested in sticking around and running the downsized environment.

Downsizing doesn't just affect the CIO, it clearly changes the skill set and procedures of most IS employees. For example consider the following impacts:

Data Center Staff—These employees are no longer needed as the care and feeding of mainframe consoles, disk drives and tapes disappears. Larger shops may still operate a server room with air conditioning and fire protection, but it's likely to be manned by a much smaller staff than a mainframe glass house. One new job required is that of LAN administrator; depending on the skill set of your operations staff, some of these people may be appropriate for the job.



Programmer—In the late 1980s, as pioneering companies downsized, there emerged a problem in retaining experienced staff. If a typical mainframe programmer salary's was $45,000, then someone with comparable skills and more productivity on the PC could be hired for closer to $30,000. Many individuals faced with this fact, decided that their fortunes were better served by going to another company where they could continue to do mainframe types of applications. High turnover was commonplace in these situations—in some cases, I saw the IS staff turnover reach 100%!

People with mainframe skills like COBOL, CICS and DB2 are finding themselves almost unemployable. As a recruiter commented in the August 24, 1992 issue of ***Computerworld***, "These guys with just plain COBOL/CICS skills are dead in the water...the mainframe guys are becoming dinosaurs. We have not placed one of these guys in a year." Programmers who are retained will need a new skill set that includes GUIs, client/server computing, object orientation and SQL. For many programmers, the nature of their job will change from one of original development to one of component assembly and debugging. Some programmers will now develop client systems and, therefore, should be located in user departments. Others will work on the network and/or database and should be located within the central IS unit.

Users—A good friend of mine, Ed Yourdon, once pointed out that only two industries, drugs and computers, call their customers "users." I don't remember his exact point, but the parallel is definitely interesting. Computer user's lives are absolutely seriously affected by a move to downsizing technologies. For one thing, they must become more closely involved in the department systems development process. Many of the people doing the development

work will now be located in end-user departments for closer contact with the final consumers of the developed product. Also, prototyping with GUIs is becoming a more critical and common development technique which involves users. Finally, user management may now be managing a portion of what was previously controlled by the IS department.

2. Reorganize, distribute, refocus and downsize the IS department

The nature of an organization is likely to change substantially as distributed, network-based computing takes hold. Changes will occur in the business units using new systems as well as in the IS department that is developing (at least partially) the new environment. The concept of business process re-engineering concerns what happens in the business units as new communications and computer systems take hold. Typically, the end result is an organization that has fewer management layers in the organization hierarchy. Walter Wriston, former chairman of CitiCorp has written about this in his 1992 book ***The Twilight of Sovereignty***.

Data processing techniques are now coming full circle and changing the culture, organization, and approaches used by IS professionals. Over time, the size of the typical corporate IS department will shrink as users become more intimately involved with computer-based systems. In addition, the focus of the IS department has to change—away from applications and towards networks and databases. In other words, the successful data processing department of the future will focus on databases, connectivity and standards. The development of most of the client side work will be left to the consumers of these new systems. This new approach is graphically illustrated in *Chart 1.*

"The middle managers who used to convey information and the upper managers who 'owned' it and held power thereby are losing that power and in some cases their jobs. Information systems flatten the management hierarchy. They change the very meaning of management and the skills needed to do it well."

—Walter Wriston, 1992
The Twilight of Sovereignty

In the past, the IS professional was concerned with the mainframe. That concern is now replaced by the management of shared data and networks. Previously, the typical user department was concerned with a dedicated minicomputer that ran the department applications. Now the user department is dealing with a series of applications that run on the client workstations and are, in turn, supported by data accessed locally over a LAN server or remotely through wide area networks, distributed databases, information warehouses, or other appropriate technologies.

Lastly, and of most importance, the consumer of these technologies and data is now dealing with something very different. Depending on the user's background, he/she will be

(continued on next page)

Rules for Downsizing...

(continued from front page)

accustom to working with mainframe slave terminals or independent PCs. That user will have non-personal computers that sport graphical interfaces and run client side applications against shared data bases or sources.

3. Staff carefully—include both PC and mainframe people on your team

A downsized system, the way I define it, includes important elements from both mainframe and PC technologies. Word processing is not downsizing. Word processing has always made more sense on a PC, stand-alone or networked. *Downsizing means the use of networked workstation and PC technology to build enterprise-oriented, transaction processing types of applications.* Such applications must be supported by robust systems. Robust means that data integrity and security are built in features. A robust system has recovery techniques to insure that no transactions are lost or half completed; transactions are to be either fully completed or all interim changes should be backed out.

The essence of mainframe systems has been robustness. PC systems have focused more on usability and immediate responsiveness. And now with Windows and Macintosh GUIs predominating, PC systems provide ease of learning and adaptability as the user interface remains similar across applications.

In many shops there has been a great wall separating the PC support staff and the mainframe systems people. However, successful downsizing requires skill sets from both camps. The mainframe group can bring expertise in large scale systems; they understand the requirements for applications that must support large numbers of people or must operate in a 7 X 24 mode. PC people bring experience with GUIs, DOS, Windows, NetWare, etc.

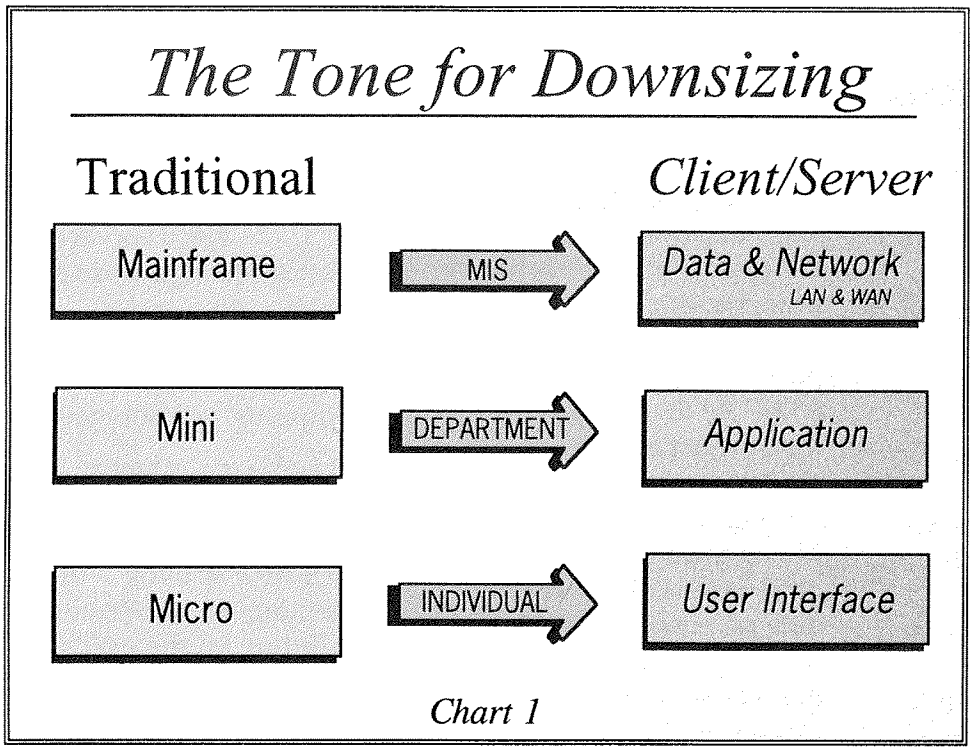


Chart 1

Those organizations that build development teams out of both groups will do better when downsizing. It is important to do the necessary to make sure you have communications going on between your PC and mainframe people. Even something as obvious as staggered, mixed offices between the two groups can help.

4. Use outside help including conferences, consultants and integrators

It should be obvious that a move to any new technology requires major investments in learning and training. And yet, this is something that is frequently underestimated, even by the most experienced people.

Much of the technology in this field changes so rapidly that it should all be considered current events. Before I made the decision to publish ***Schussel's Downsizing Journal,*** a book publisher asked me to write a textbook on the subject of downsizing. I rejected that idea because it seemed to me that the usefulness of a book on this subject would be limited—its content could (and probably would) already be dated by the time it hit the streets. Conferences and industry literature are a good way to keep current, and it's essential to keep current if your downsizing investments are to make any sense.

Probably the most frequent "morning after" comment I've heard about successful downsizing projects is something along the line that consultants or systems integrators should have been used more frequently than they were. We are dealing with new technologies and bringing in knowledgeable advice is usually cheaper in time and money than bull-headed trial and error discovery approaches. My advice here is to identify the technologically tough areas and have consultants available who can help when they are needed.

5. Modularize applications to machines *and* 6. Throw extra hardware at the problem

The best thing about downsizing is that hardware is cheap. At the current time, a PC instruction cycle costs less than 1/1000 of the same function on a mainframe. Since these cycles are so cheap, you should not hesitate to use or even squander them if it provides any benefit.

There are many ways that extra hardware can help:

- ☑ One of the problems with NetWare is that no protection is provided between various NLMs (NetWare Loadable Modules—applications running under the control of the server) in the same machine. One method of obtaining application protection in this environment is to use a different machine for each major NLM application. For example, a database server such as Sybase, Gupta, or Oracle should always be run on its own machine with no other significant live applications.
- ☑ If extra machines and machine parts are kept stocked, then trained user personnel can substitute working parts (or machines) for the down hardware. This will lower your requirement for on-site, emergency maintenance support. Slow turnaround maintenance is far less expensive.
- ☑ If your database response is slow, buying additional servers to split up the database, obtaining a faster server, or purchasing additional network cards are all easy, relatively inexpensive, and fast solutions.
- ☑ Adding more client terminals so there is never a wait for service will go a long way towards improving the image of the system.

7. Limit the diversity of your installed base

One of the nice things about the downsized world is that "open systems" rule. There are plenty of technologically-based answers for almost any data processing problem. And, for those "answers" to be

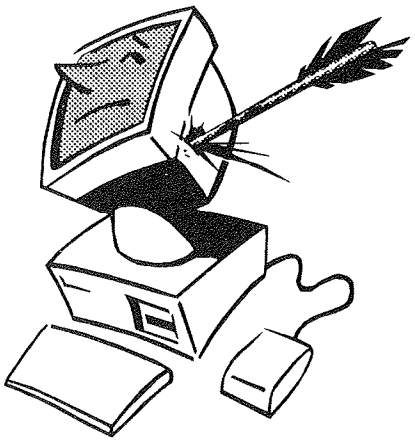
(continued on page 15)

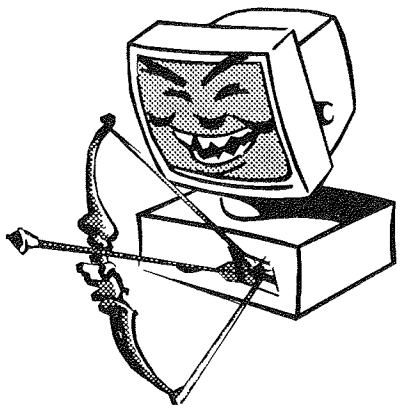
Operating System Wars—The Latest Status

Things have changed recently in the market for "open" operating systems. Novell has acquired USL, the caretaker of the UNIX standard. In addition, Novell is now shipping NetWare 4.0, their first product targeted at the enterprise-wide computing market. And, of course, NetWare 4.0 now offers protected mode computing, an omission in NetWare 3.11 that some consultants criticized severely.

Microsoft is about to ship Windows NT, an advanced client/server, protected mode operating system, in two different versions—one for the client and another for the server. In addition, Microsoft is currently claiming that Windows 3.1, the latest version of DOS/Windows, is shipping at the rate of over one million copies per month! That is why Bill Gates is the richest man in America; his holdings in Microsoft stock are currently valued at over $7 billion. Recently, Microsoft has begun talking in public about next year's Windows 4.0, the important new version of Windows 3.1—but, more on this later.

Not only does IBM have a new CEO, Lou Gerstner, but it has shipped over two million copies of OS/2, an operating system that has now matured into a useful, stable PC product—it is probably the best PC product IBM has shipped since the PC/AT.





When the stock of such companies as AST, Gateway, Dell, and Compaq fluctuate significantly in one or two cycles per year, it is clear that hardware isn't the controlling issue in computer systems anymore. There are simply too many companies capable of building superior hardware quickly. Toll free (800) number marketing and the direct response channels means that the barriers to entry in the hardware field have come tumbling down. This has all been forecast by consultants (including us here at ***Schussel's Downsizing Journal***) for several years now. DEC's and IBM's very public agony is only too visible proof of the descent of hardware manufacturers (and integrated suppliers) and the shift of power to software producers. Intel and Microsoft, each, have gross sales of 10% or less of IBM's, yet their market capitalization values (# shares outstanding X price/share) each is about the same as IBM's. In the new world, software dominates. And in the software kingdom, the lion's share belongs to those who control the operating system (O/S) market.

So that is what this article is about. The in's and out's of the O/S wars, how each of the leaders is doing and what some forecastable results will be.

So what is an "Open" O/S anyway?

There is no single widely-accepted definition for an "open operating system." In general, I've found five different definitions used frequently:

1. *De Jure Standards*—These are definitions published by some type of, typically, non-profit group. ANSI, ISO, SQL Access Group, and X/Open are all examples of such groups. Their standards are normally built on top of the lowest common denominator of existing products. And, usually, de jure standards are published some years after leading vendors have brought such capabilities to customers. De Jure standards are becoming less and less relevant to most buyers in the 1990s, because they are too slow in coming and can't keep up with fast moving computer technology and changes.
2. *Interoperability*—For many years, DEC talked about it's VAX/VMS systems being "open" because of extensive facilities for communication and data interchange with a wide variety of non-DEC systems. Other companies such as Sybase, Sequent, and IBM have also been leaders in providing facilities for interoperability. There is nothing wrong with providing such data interchange facilities, but in today's market, that just isn't enough for your system to be considered *open*.
3. *Flexibility to change hardware vendors*—Some software vendors have long championed their ability to run on a wide variety of different hardware environments. Examples of leaders in this approach include tool and DBMS vendors such as Cincom, Sybase, Oracle, and Informix. The argument is that a commitment to that software vendor's architecture frees the user from being tied to any individual hardware vendor. That is true to a significant extent, however, this path will lock you into the software vendor's products.
4. *Flexibility to change Software Vendors*—The exact counter point to the idea of "open hardware" is "open software." Some hardware vendors, including Pyramid, Compaq, and Sequent, have championed the fact that their hardware runs standard software systems like UNIX, Windows and DOS. The point is that by using hardware from one of these vendors, you are then free to choose from a wide variety of different software vendors. To the extent that these hardware vendors support many popular software standards, it is true that the user has many options, and is therefore "open" to choose.
5. *Marketecture Standards*—I am finding that the most important definition of "open systems" has to do with the marketing of ideas, systems, hardware, and software that exists for that environment. Years ago, I considered the IBM 360/370 environment to be the most *open* because its size encouraged other firms to build clone hardware and software products to operate in an IBM compatible environment. Nowadays, the largest O/S vendors for the new downsized and distributed computing culture are Microsoft and Novell (with IBM possibly joining the list). The huge and rapidly growing base of NetWare and Windows sites means that tens of thousands of independent hardware and software vendors have been drawn to these environments and are developing improved and cost effective solu-

(continued on next page)

Operating System...

(continued from previous page)

tions for them. That is why you want to be a user of NetWare, UNIX, DOS, and/or Windows. With these O/S, your choice of capabilities and the price that you pay for those features are going to be better than you can find anywhere else. Needless to say, any software manufacturer who wants their product to become a *marketecture* standard has to freely license to all qualified VARs under policies that generate favorable business partners.

Vertical integration is gone!

The subject of open O/S is interesting now because the marketplace for computing solutions is going through a fundamental paradigm (sorry, but it fits) change. Computer-based solutions have always been assembled by single vendors such as Hewlett Packard, IBM, or DEC. Each vendor would assemble the necessary hardware, software, and network pieces to solve a customer's problem. The vendor would stage the platform in their shop before it shipped, and make whatever adjustments were necessary to insure that the delivered solution worked. About the only piece of the computing solution that wasn't from the vendor was the application software. Historically, computer vendors have not been good at building application software. Therefore, application software has been built by specialized software vendors who do just that job or by the users, themselves.

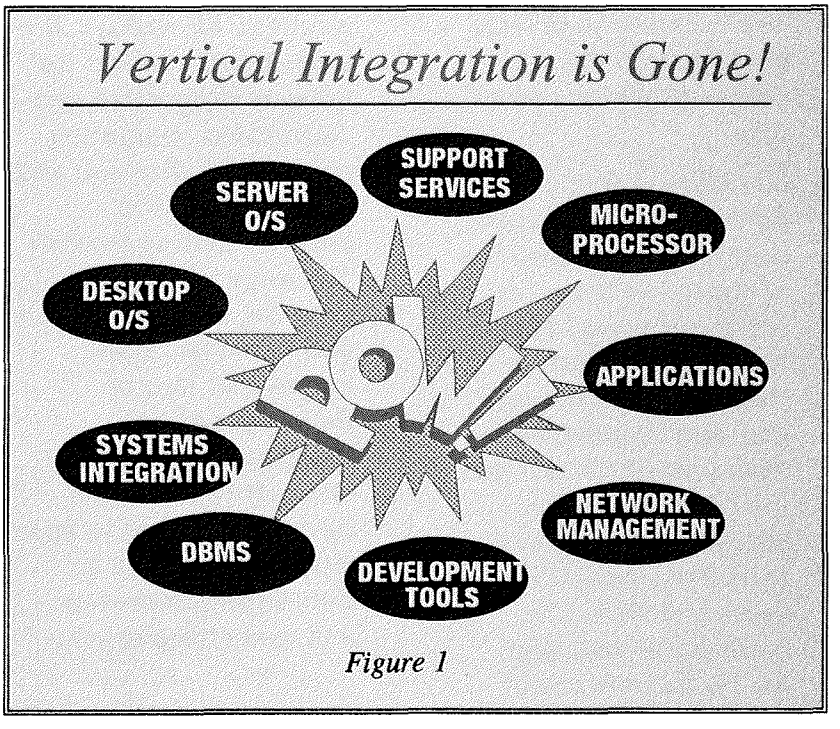


Figure 1

This traditional method of doing business is becoming increasingly unpopular. For various reasons, in the 1980s, a transition among vendors began as each vendor became more specialized and more expert in small segments of the total computing market. For example, Microsoft became the dominant expert in the development *and marketing* of PC operating systems, while Novell developed expertise in PC networking. In a similar fashion, various experts and consultants appeared for small market segments including microprocessor CPU's (Intel, Motorola, Sun, etc.) DBMSs (Sybase, Oracle, Informix, etc.), network software (Novell, Microsoft, Banyan, IBM, DEC), and so forth. Companies are now picking the "best of breed" in each market segment and integrating those solutions either themselves or with the help of contractors. The large, integrated suppliers of the 1970s and 1980s like IBM and DEC are attempting to recast themselves into loose collections of independent companies. Their hope is that this approach will make them more re-

sponsive to quick changes in market trends and at the same time lower the overhead attached to having huge corporate staffs.

The way that a company achieves market dominance in its segment now is to have a reasonably good product combined with superior marketing. Critical to the marketing approach is developing a complete list of companies that support your product. When that list gets long enough, you then declare your product to be an industry standard. Usually, such industry standards are purchased for the "popularity" reason alone—the wide availability of compatible products—as long as they remain reasonably competitive and well-priced.

New markets for operating systems

No where is the change in business procedures more obvious than in the market for O/S. Proprietary O/S running on proprietary hardware was the rule in most corporate data processing shops until very recently. Today the attention of most application builders has switched to products like MS-DOS, Windows, UNIX and NetWare. These products are made by software, not hardware, vendors and are equally licensed to all hardware VARs, most of whom choose to bundle in these software products with their hardware.

IBM, with its OS/2, is playing in the open O/S market, but OS/2 has been partially hampered by its reputation for not supporting as wide a variety of hardware systems as IBM's competitors Microsoft and Novell. GS

In Part II of this article, to be featured in the June 1993 issue, Schussel will look at the current playing field, analyze who is set up to win the battle for dominance among operating systems.

Rules for Downsizing...

(continued from page 11)

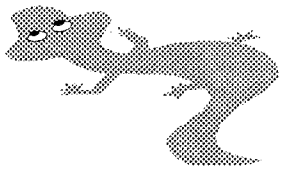
successful, they have to interoperate with established industry standards. When I say *standards* I mean market, not de jure, standards. The Macintosh, Windows, DOS, MVS, NetWare, and UNIX all are examples of market standards. In other words, these products are so successful that independent developers have added functionality and designed products for these architectures. Having such a wide choice of products is one of the best things about the networked, downsized world.

An important point here is that it's nice to have this choice at selection time. One shouldn't, however, expect a reliable, robust environment if too many of these alternatives are selected and then, in turn, supported in a live production system.

We are now starting to see some stories where severe maintenance problems are caused by a lack of attention to implementation standards during the fielding of a system. Variations in operating system release levels, interrupt controls, slot availability, add-in boards, disk drives, video displays, etc. can cause network or compatibility problems.

A wide variety of both software and hardware is good at selection time. At implementation time, however, limit the number of different system components to the minimum possible. I know that this is easier said than done. William Connor of Motorola, who supervised a downsizing of Motorola's General Systems Division, mentioned to me at Downsizing EXPO that he now has 45 software vendors. He compared that number to before downsizing when his division had only four! GS

UPCOMING **downsizing** Events...



DATABASE WORLD and **CLIENT/SERVER WORLD** are once again being jointly held in Boston, June 14-16, 1993. There are nine conference tracks between both shows: *Object-Oriented Technology Conference, Database Technologies Conference, DB2/Information Warehouse Conference, Xbase Conference, Database Connectivity Conference, Client/Server Databases Conference, Managing the Client/Server Environment Conference, Client/Server Networking Conference, Building Client/Server Applications Conference.* Keynotes are being delivered by several renowned industry figures including: Chris Date, Michael Stonebraker, George Schussel, Larry DeBoever, and William Zachmann. In addition, Philippe Kahn of Borland and Charles Wang of Computer Associates are to be plenary speakers.

This June, there are two back-to-back seminars being held in Dallas: *Implementing Client/Server Applications and Distributing Data* with Herbert Edelstein, June 8-9, and *Cheryl Currid: Implementing Downsizing,* June 10-11. Edelstein's seminar shows attendees how to use client/server technology to effectively distribute data throughout their organization. The differences between cooperative processing and client/server will be covered in detail, as well as open systems, network considerations, relational DBMSs, database integrity, and interoperability. Currid, in her two-day course, will provide, though case studies, the knowledge you need to assess your company and implement the best strategy for downsizing. In addition, DCI also offers several one and two-day downsizing seminars with such industry notables as Larry DeBoever, Richard Finkelstein, Jeff Tash, and George Schussel.

For more information on any of these classes or conferences, call DCI at (508) 470-3880.

Schussel's DOWNSIZING JOURNAL
May 1993
Volume 2 Issue 9

Editor: Dr. George Schussel
Managing Editor: Stacey S. Griffin
Subscription rates:
- $199 annually for U.S. residents
- US$225 annually elsewhere

SDJ is published monthly by:
Digital Consulting, Inc.
204 Andover Street
Andover, MA 01810 USA
508-470-3870 FAX 508-470-1992