

Schussel's

DOWNIZING JOURNAL

September 1992

In This Issue:

Distributed & Client/Server DBMS: Underpinning for Downsizing, Part I of IV
The first in a series of four articles that examine the functionality and structure of distributed and client/server DBMS1

Downsizing DB2 Applications Five Approaches
An evaluation of five major downsizing approaches by consultant Howard Fosdick1

So Long Ken!
A historical tour of the high and low points of Ken Olsen's 35-year career at DEC6

A Guide for Making the Transition to Collaborative Distributed Application Development
An irreverent look at the issues facing application developers in the 1990s14

Upcoming Downsizing Events16

Distributed & Client/Server DBMS: Underpinning for Downsizing Part I of IV

One of the key trends in modern computing is the *downsizing* and distributing of applications and data. This paradigm shift is occurring because companies want to take advantage of modern micro-processor technology which allows them to benefit from the new styles of software with graphical user interfaces (GUI). *Client/server* and *distributed database* technologies are two fundamental enabling technologies involved in downsizing.

(continued on page 8)

Downsizing DB2 Applications Five Approaches

by Howard Fosdick

Downsizing - the hot topic for this decade - has certainly captured a significant portion of the press. However, with the current enthusiasm about the money that can be saved through cheaper hardware, we've seen almost a soft-pedaling of the hidden costs involved in software conversion and employee training. The primary motivations for downsizing I hear most people talk about are strictly hardware-oriented. However, if you're from a traditional IS background

(continued on next page)

Downsizing DB2...

(continued from front page)

with a mainframe-centric shop that now includes PCs and networks, you already know that hardware costs are somewhat meaningless in terms of how an organization is run. My point here is that it is extremely important to include the costs of IS staffing and retraining when evaluating downsizing. To underscore the importance of this, let me tell you that most IS shops have a staffing budget that is at least triple their hardware budget.

I agree that one of the driving forces behind downsizing is less expensive hardware. However, in terms of IS shops running legacy systems, there exist better reasons for downsizing. One such reason is new and improved user interfaces. Another advantage is data accessibility — even with relational technology such as DB2 which makes data more available than with IMS, data is still relatively inaccessible. This is particularly true if you're comparing DB2 with the PC model of computing. The bottom line is that while hardware may be the reason

why downsizing is now both possible and popular, the real motivation for most IS shops should be the desire for better business systems.

Different downsizing approaches

There are several diverse approaches to downsizing. Unfortunately, very often there is no analysis available on why one approach may be better than another. It is essential to understand the different methodologies and for which environment each will work best. Following are explanations and evaluations for some of the major downsizing approaches that have been addressed by the press.

The basic five approaches most frequently followed are:

1. To take an existing mainframe application and migrate it to a PC.
2. To take an existing mainframe application and migrate it to a PC LAN.
3. To take an existing mainframe application, keep it on the mainframe, and add PC front-ends.
4. To develop applications on cross-platforms. For the past five years, people have been developing applications on PCs to be shipped to the mainframe for use. But, this style of development can work the opposite way also: applications can be

Motivating for downsizing

In this new age of downsizing, people are exploring not only new computing paradigms, but also the cost justifications of moving to the new technologies. Comparing hardware costs between traditional mainframe, channel communication architectures and micro-processor-based architectures is currently a very popular trend (see chart below for some numbers). Although there exist various measurements used for comparison — cost per MIP, cost per transaction per second, cost per megabyte, cost per megabyte per disk — anyway you look at the numbers, the hardware costs per user are quite a bit less in a PC or micro-processor-based environment than with a mainframe.

Hardware Costs

(in dollars)

	Micro	Mainframe
1 MIP	500 - 1,000	100,000
1 MB RAM	100	6,000
1 MB Disk	5	10
RDBMS Software	2,500	300,000

Source: Performance Computing Inc. (1991)

developed on the mainframe, and then shipped to PCs for use.

- To keep your existing mainframe system and add PC LANs and software to increasingly evolve the systems. Eventually, most or all of the applications will reside on the PC and this move will have been accomplished with an evolutionary approach. This is the downsizing method I believe to have the most potential for traditional IS shops.

Following are details on each of these different approaches, and some suggestions to where each architecture works the best.

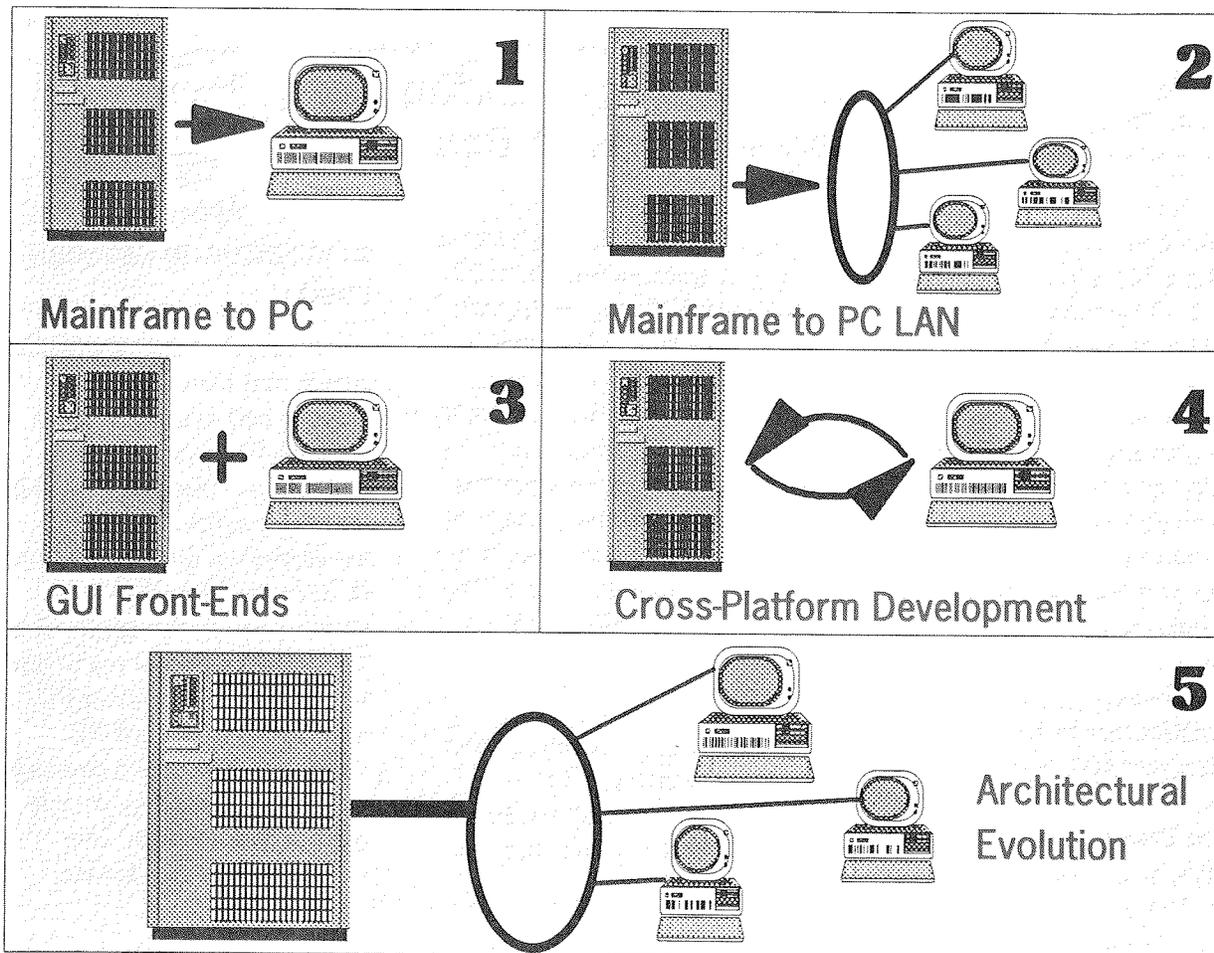
 **Approach one: mainframe to PC portability questions**

If you are contemplating porting an existing mainframe application to a PC, this will provide benefits in the form of lower hardware costs. The issue here is that with a PC, you are limited to single user applications. Therefore, scalability becomes not simply a hardware function, but is also a software concern. Also, remember that the PC and mainframe environments must be compatible unless rewriting applications is your idea of fun. You will quickly discover if you choose this form of downsizing that the world of mainframe software is divided

into several classes, some of which are portable, but many of which are not.

Another major factor to be considered when porting mainframe applications to PCs is the different architectures. First off, each machine has a unique character set. Although most software translates rather easily, occasionally you will run into problems like collating sequences. In addition, between these two different types of computing, files systems are different, transaction monitors are different, etc. There is an

(continued on next page)



Downsizing DB2...

(continued from previous page)

endless slew of small (or large) variations.



Approach two: PC LANs

Given the single user limitation of PCs, when people talk about porting mainframe applications to a PC environment, they usually intend to use a local area network of PCs. The normal assumption here is that on the PC LAN, there would be either a client/server architecture or a DBMS client/server architecture. This isn't true all of the time, but it is usually the structure that you will see.

The important issue then becomes: how compatible is that downsized, PC LAN environment with your mainframe? In most cases, these two environments are not very compatible. If you are planning to move from a mainframe to a multi-user LAN environment, you will essentially have to redesign your software. If you have a mainframe DB2 application that uses CICS as a monitor, how are you going to re-architect that to fit into a Microsoft SQL Server environment? If you use the same design as you did for CICS, it will not work. I want

to warn you about what you have seen in the press. There has been an abundance of wonderful articles about companies who have unplugged their mainframe and replaced it with a PC LAN environment, but how common (or practical) is this for the average IS shop?



Approach three: a face-lift

The third popular approach to downsizing is to keep your applications on the mainframe and add PC front-ends. This means adding a PC product

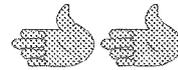
...Remember that the PC and mainframe environments must be compatible unless rewriting applications is your idea of fun....

similar to EASEL or Mozart in front of your mainframe DB2 application. Then, effectively, when a 3270 block comes down to the PC, users see a GUI rather than the traditional, less interesting, character-based display. How does this type of architecture classify as downsizing? Because users are taking advantage of the PC by using cheaper MIPS and employing a graphical interface.

The largest benefit of this method is that you're upgrading the interface which makes users happier, but you

haven't had to change the underlying application. For such reasons, this method of downsizing is non-disruptive to corporate computing.

However, there are downsides to this technique that require attention. With this scenario, the PCs are not event driven. The user may have a mouse and click on icons, but behind the scenes, the result is the same as if the user had pressed the enter button on a 3270 — the screen freezes as information is sent to the mainframe for processing, and remains frozen until information is returned. There will always be that menu-driven, 3270 computing philosophy if all you change is the interface.

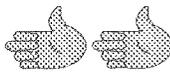


Approach four: cross-platform development

Another of the downsizing trends that has been both popular and successful over the past five years is cross-platform development. In this situation, software is developed on a PC and shipped to the mainframe for production, or the application is developed on the mainframe for production use on a PC. This first scenario of PC development is more common and has been well-proven in the last half-decade. The benefit it provides is that

software developers can take advantage of the interactivity of the PC. At the same time, by shipping the application to the mainframe for production, users can still take advantage of mainframe software and administrative functionality and organization.

The other cross-platform development option, developing applications on the mainframe for PC production, is not very common, but is done. This setup allows the user to take advantage of the standards and procedures that have been in mainframe environments for years. Then, at the PC level, users are able to benefit from the lower cost of hardware.

 **Approach five: distributed access**

The last and the most reasonable downsizing avenue to follow consists of taking a central mainframe shop and slowly evolving it into a distributed access system. This

process entails doing nothing with legacy applications residing on the mainframe, but instead progressively using a connected PC LAN or single PCs to develop and run new applications. Using this technique over time, more applications and data will be moved onto the PCs.

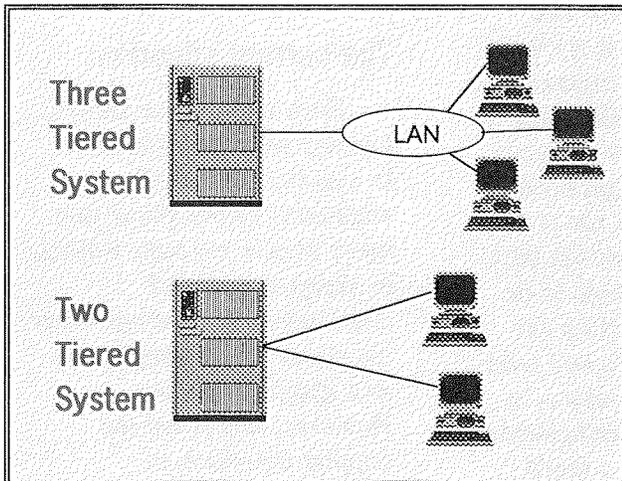
There are two popular ways that such a system could be configured: the two tiered system and the three tiered system.

The two tiered system basically provides a direct link between the PCs and mainframe. The three tiered configuration adds another layer interposed between the PCs and the mainframe. Currently, the three tiered approach is most popular among vendors. However, the two tiered approach has been proven to work well in certain situations.

As far as products for these environments are concerned, for three tiered approach, you need an operating system on the server, perhaps NetWare, OS/2, or UNIX, and depending on which operating system you choose, you need a compatible DBMS server like SQL Server or OS/2 Database Manager.

Once you have a DBMS server, you then need front-end software on the PCs that allows you to develop applications — either character-based or GUI —

(continued on page 13)



SAA to the rescue?

To aid the development of portable applications, SAA was introduced in 1987. SAA was IBM's grandiose plan to make all software portable and compatible between different platforms. Now that SAA has been around for five years, it has had a large impact. I know that this is an odd thing to say since SAA is no longer in the limelight, but its usefulness has grown tremendously. IBM has not been able to provide perfect portability between all platforms, but substantial progress in this direction has been made. Applications written on your mainframe using SAA guidelines you will find are significantly more portable than pre-SAA applications.

Another large impact of SAA has been that cross-platform development is now recognized by all independent software vendors. Although IBM propagated SAA, the fact is that other vendors have taken their cue from this — they saw IBM trying to evolve a world in which applications are totally portable, and determined that they had best do the same in order to compete.

So Long Ken!

During the week of July 13, DEC officials announced the retirement of their President and founder, Ken Olsen. This was a momentous event in the computer field. During his 35-year career, Olsen created the second largest computer company (with annual sales of \$14 billion) in the world. More than the Watsons at IBM, or any other single individual in computer science's history, Olsen's contribution was a huge, protracted success. The fact that he retired at a relatively low point in DEC's history, during a week when layoffs numbering in the tens of thousands were announced, was historically unfortunate. However, for DEC's future, those layoffs may have been the best thing that could have happened. DEC's stock price climbed six points (approximately 15%) during that week.

Many outsiders (publicly) and insiders (quietly) had been calling for Olsen's retirement. The forces in this field that

finally brought about his retirement are two subjects close to our hearts: downsizing and open systems.

The irony of DEC's fall

In what could be considered an ironic twist, Olsen actually started the downsizing movement in the 1950s. In those days, downsizing meant moving from mainframes to mini-computers. Olsen's DEC pioneered the movement of mini-computer solutions from embedded or special purpose

richer, more robust environment than UNIX.

Olsen's other major mistake — ignoring the role of PCs in downsizing — is less excusable. Some stories that I have heard (but have not been corroborated) indicate that DEC engineers developed and built several different PC models in the 1970s, but each time Olsen terminated these projects. My sources said that he didn't approve of the idea of using computers that weren't networked or time-shared. In other words, the

empire that was created by a downsizing phenomenon was resistant to the new microprocessor-driven downsizing revolution. DEC's demise has been brought on by its ill-defined, lukewarm response to downsizing as

represented by business computing on the PC and engineering computing on workstations.

...Olsen's accomplishments are both monumental and legion. However, it is clear that the difficult circumstances now surrounding DEC are Olsen's doing....

dedicated systems to general purpose engineering and business applications.

Olsen's accomplishments are both monumental and legion. However, it is clear that the difficult circumstances now surrounding DEC are Olsen's doing. His contempt for open systems has been widely reported in the press. UNIX advocates within DEC have always appeared to be treated as second class citizens. That Olsen would hold the idea of open systems in low regard is understandable. After all, VMS is a much

The current situation

Even today these anti-PC feelings persist within DEC. In sessions with DEC engineers, I have found that many people are only willing to accept the idea of client/server computing when the client is operating solely as a graphics server. However, this is not the popular definition of

client/server computing which allows for full programmability on both the client and server, with work being done where it most logically belongs.

The distance DEC has fallen can be quantitatively measured by the fact that the market valuation for DEC (share price \times number of shares outstanding) places its total value at less than that of Novell, a company that is only 1/20 the size of DEC.

There is a growing revolution going on inside of DEC. The new guard (*downsizers*) are battling for power with the old guard (*VMSers*). Downsizers endorse the various alliances that DEC has been forming with PC powerhouses like Microsoft and Novell. These revolutionaries have a vision of a DEC that will offer systems integration, software, networking, and other support services to open and downsized offices. The downsizers first approached me for assistance over two years ago. At that time, they were unable to generate much corporate support. Now, however, DEC is much more receptive to projects that showcase the company's talents in open and downsized environments. I have been assured that Olsen's departure will accelerate the firm's metamorphosis into the new, downsized world.

Barriers to success

It will not be easy for a \$14 billion company to stem its losses. Many key executives and technical personnel have already left DEC. Besides Olsen, the most famous DEC departee probably is Dave Cutler, the chief architect of DEC's renown VMS. Cutler left DEC to join Microsoft and supervise the development of Windows NT, the product that has been nicknamed Microsoft's VMS. Windows NT is a scalable platform that runs on any size machine from any vendor (this is better than VMS). DEC has (finally) realized the opportunities that Windows NT will present, and has struck agreements with Microsoft for implementation of Windows NT on DEC's new Alpha-based processors. That's good

news for both DEC and Microsoft. However, Microsoft will pick up much of the value and account control that would have gone to DEC had it developed its own version of NT instead.

A major problem for DEC is that as a corporation, it doesn't have the ability to either attract or retain the top IS people. Also unfortunate for DEC is that Microsoft does attract the leaders in this field. The reason is simple: over the years, Microsoft has created hundreds or even thousands of millionaires through stock options and soaring stock prices. In contrast, DEC's low, book value-based stock limits employment attractiveness to the best and brightest people.

(continued on page 13)

Some historical trivia

For two decades, all DEC machines carried the initials "pdp" at the beginning of the model number. The explanation I heard for this is as follows.: it seems that early on in DEC's history, the company worked closely with the Canadian Department of Defense to provide computers for the monitoring of air defense radar. As the story is told, there was a Canadian law that required the Parliament's approval before the government could acquire any computer. That, of course, was in the days when a computer cost millions of dollars, and national debts were measured in millions, not billions of dollars. The computers that DEC bid for the Canadian job were priced in the \$100,000 range, and were well within the budget authority of Canadian Defense Ministry. Nonetheless, Canada's law specified that all computer purchases be approved by Parliament — no one at that time had thought about the possibility of "cheap" computers! This logistics problem was solved by calling the DEC machines "peripheral data processors," or pdp machines, rather than computers. So was born a name that would survive throughout the 1960s and 1970s.

Distributed & Client/Server...

(continued from front page)

Client/server approaches allow the distribution of applications over multiple computers. Usually the database(s) resides on server machines while applications run on client computers. While the type of computer used as a server varies widely (e.g. you could have a mainframe, mini-computer, or PC), most clients are PCs. *Local area networks* (LANs) provide the connection and transport protocol used in linking clients and servers.

A *distributed database* offers capabilities similar to client/server databases. The most fundamental difference between the two architectures is that the distribution of data within a distributed database is both pervasive and invisible. In this style, a database management system (DBMS) resides on each node of the network and allows transparent access to data anywhere on the network. This means that the user is not required to physically navigate to the data.

The distributed database set-up is different from the

client/server approach in which the application must be aware of the physical location of data, at least to the extent of on which server it is. With a distributed database, once an SQL query or remote procedure call is directed to the appropriate server, its query optimizer for SQL will handle the internal database navigation. Many of the advanced functions described later in this article series, such as stored procedures, triggers, and two-phase commits, are available in both client/server and distributed DBMS

...By accepting a reduction in functionality from what a distributed DBMS provides, vendors have developed client/server DBMS that run exceedingly well on modern PCs and networks....

environments.

Client/server DBMS and distributed DBMS have much in common, as will be discussed in this article series. Both are based on the SQL language, invented in the 1970s by IBM, and standardized by ANSI and ISO as the common data access language for relational databases. Both are appropriate for distributing applications.

Introduction to distributed database computing

The market for modern distributed DBMS software started in 1987 with the announcement of INGRES-STAR, a distributed relational system from RTI (now the INGRES Division of ASK computers) of Almaden, California. Most of the original research on distributed database technology for relational systems took place at IBM Corporation's two principal California software laboratories, Almaden and Santa Theresa. The first widely discussed distributed relational experiment developed within IBM was a project named R-Star. It is because of IBM's early use of the word "*Star*" in describing this technology that

most distributed database systems have "*Star*" incorporated into the name. Today, the market for distributed DBMS is almost entirely based on the SQL language and extensions. (The principal exception is Computer Associates, which inherited IDMS and DATACOM prior to relational systems and has implemented distributed versions both with and without SQL).

Distributed DBMS products can be thought of as occupying the Mercedes Benz echelon of the market-place. These products support a local DBMS at every node in the network along with local data dictionary capability. This requirement that a piece of the DBMS exist on each node is the essential difference between distributed databases and client/server systems. In a client/server approach, the DBMS resides on one (or a few) nodes, rather than all of them, and is accessed from a requester piece of software residing on the client.

The market for distributed DBMS has grown slowly for two reasons: 1) users aren't sure of how to use the products, and 2) vendors are taking the better part of a decade to deliver a full range of functionality. Another important and unanswered

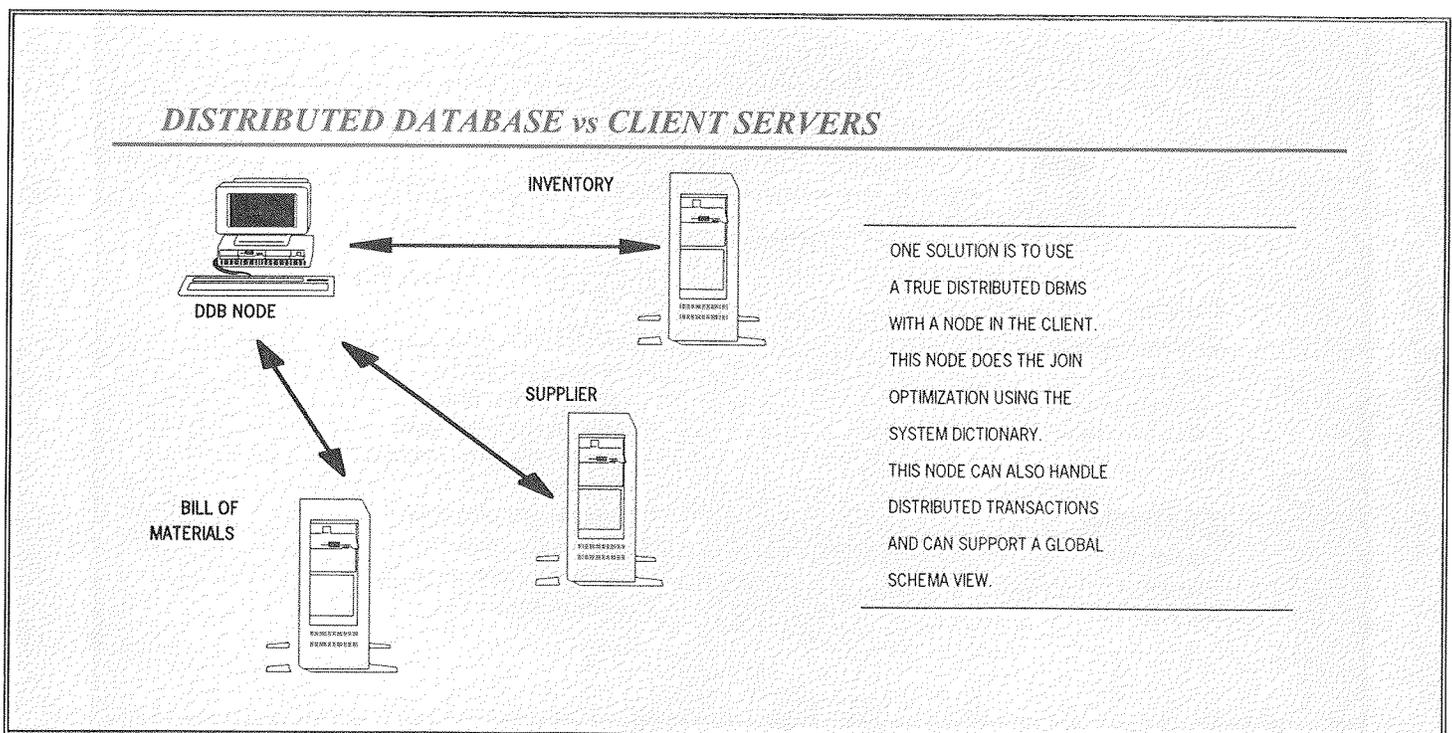
concern is that companies don't know what the cost will be for communication functions that have historically been run internally in single computers.

Introduction to client/server database computing

If distributed DBMS products represent the top tier of the market, then client/server DBMS engines are the Fords and Chevrolets. By accepting a reduction in functionality from what a distributed DBMS provides, vendors have developed client/server DBMS that run exceedingly well on modern PCs and networks. It is this author's opinion that the market place for client/server approaches is going to be far larger in dollar volume than that of distributed DBMS.

Much of the impetus for downsizing comes from the fact that many companies want to implement applications that were previously forced to reside on mainframes, onto faster, cheaper PCs. But, before committing to downsize such applications, assurances about the integrity of the data and applications are necessary. In addition, PCs, as well as LANs, have had reputations for not offering a mainframe level of security. Client/server computing is a solution that combines the friendly interface of the PC with the integrity, security and robustness of the mainframe. Server databases located on PC LANs use implementations of the SQL database access language — the standard database language used on mainframes. Once you've decided to build a

(continued on next page)



Distributed & Client/Server...

(continued from previous page)

client/server environment, you will be on your way to building an applications architecture that will be economical, flexible, and portable for a long time into the future.

The functionality delivered by today's client/server systems is not too different from that of a distributed DBMS. The key difference is that a client/server approach places the DBMS and DBMS dictionary at certain designated nodes where the data resides. The client program is required to navigate the system and find the correct server node for access to the necessary data. An important advantage of the

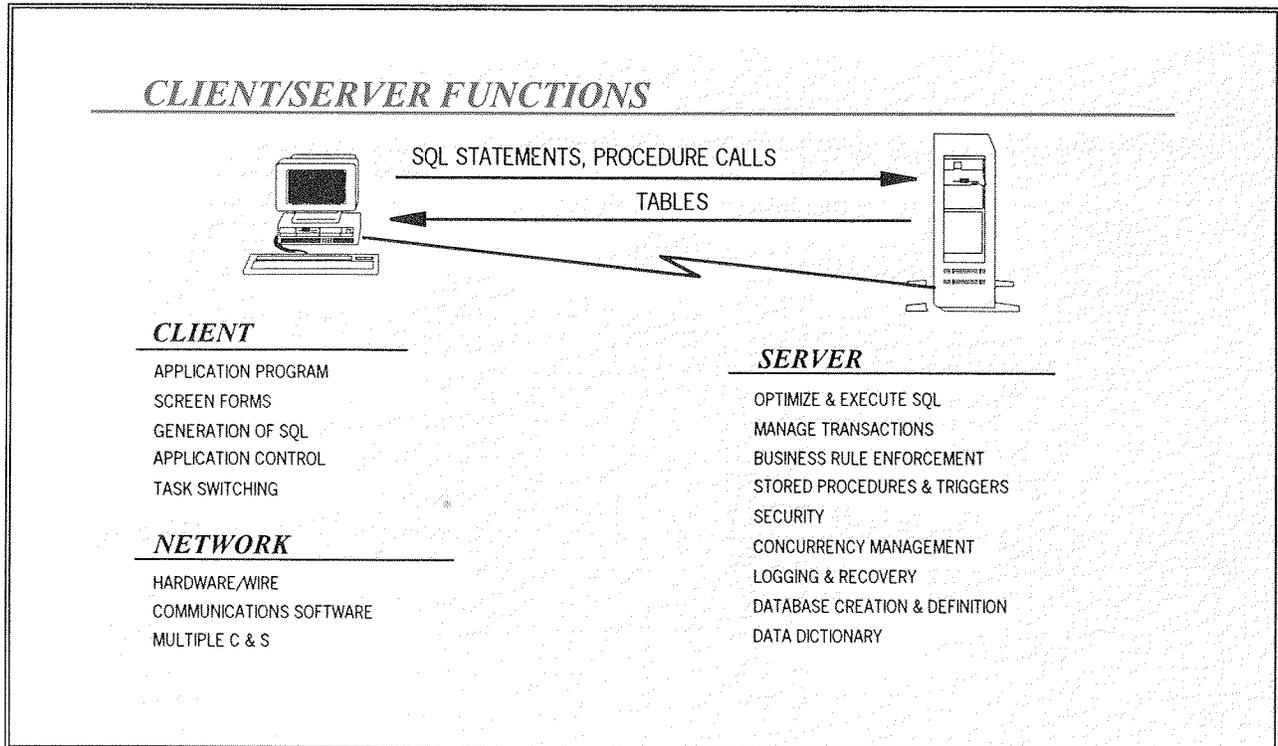
client/server approach over distributed databases is that having only one (or a few) database locations appears to be more manageable than an architecture which spreads data evenly across many nodes. Managing a distributed database properly would seem to be the more difficult challenge.

The history behind the client/server

The idea for client/server computing grew out of database machine approaches. Sybase's Robert Epstein was working for Britton Lee when he envisioned creating a database machine environment with a server that was a virtual machine rather than a physically unique piece of hardware. The systems software, then, was separated

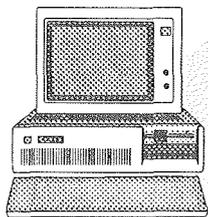
into a front-end (client) which ran the program (written in a 4GL), and a back-end (server) which handled the DBMS chores. The advantage of this idea was that the back-end (the virtual database machine) could physically be moved out onto a different piece of hardware if desired. What made this different from Britton Lee's traditional approach was that Epstein planned for the server to be a generic VAX, UNIX, or PC machine, rather than a unique, custom built database machine. By moving the database machine onto a standard piece of hardware, Sybase picked up the advantage of a vastly improved price performance for generic small systems.

About the same time that Epstein was starting Sybase, Umang Gupta (at that time a



Senior Oracle executive) had pictured the same situation and left Oracle to form Gupta Technologies, a company which has emerged as a leader in PC-based, client/server DBMS and tools. Bing Yao, the former University of Maryland professor who founded XDB Systems, was another early developer of client/server approaches to database computing.

By now, most SQL DBMS vendors have jumped into the client/server game. One exception is IBM; when IBM talks about client/server computing, what they are really referring to is distributed computing. IBM is in the process of building a fully functional, distributed architecture for all of its SQL products: DB2, SQL/DS, SQL/400, OS/2EE. IBM is taking several years to develop this approach.



A client/server computing environment consists of three principal components: Client, server, and network.

The client

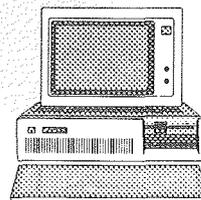
The client is where the application program runs. Normally, client hardware is a desktop computer such as an IBM PC, PC clone, or Apple Mac. The application program itself may have been written in a 4GL or third generation language such as C or

COBOL. There is an entire new class of Windows 4GLs that allows the painting of applications under leading desktop, Windows-based, operating systems.

Such Windows 4GLs support both windows-oriented application development and execution. Leading examples now on the market include: Powersoft's PowerBuilder, INGRES's Windows 4GL, and Gupta's SQL Windows. Using any of these application building approaches will result in a runtime configuration where the I/O and application controls come from the client, while the database and associated semantics run on the server. At the desktop level, most software will support the emerging windows-based standards: Macintosh, Windows 3.x for DOS, Presentation Manager, Open Look, and Motif for UNIX.

The network

The network connects the clients and server(s). Normally, networks are based on either Ethernet or Token Ring topologies, and have appropriate interface cards in both the client and server boxes. The communications software typically handles different types of transportation protocols such as SPX/IPX, LU6.2, and TCP/IP. Most network environments provide support



for multiple clients and servers.

The server

The server is responsible for executing SQL statements received from a client. Sometimes data requests are not communicated through SQL, but through a remote procedure call which triggers a series of pre-compiled, existing SQL statements.

The server is responsible for SQL optimization, determining the best path to the data, and managing transactions. Some server technologies support advanced software capabilities, such as stored procedures, event notifiers, and triggers. The server is also responsible for data security and requester validation.

The server will also handle additional database functions such as concurrency management, deadlock protection and resolution, logging and recovering, database creation and definition. The idea of managing data on a separate machine fits well with the management approach of treating data as a corporate resource. In addition to executing SQL statements, the server handles security and provides for concurrent access to the data by multiple users.

(continued on next page)

Distributed & Client/Server...

(continued from previous page)

The benefits of using SQL

An important benefit that the set-oriented SQL language provides is network efficiency. When using traditional, file-serving, PC LAN approaches, the entire data file must be transmitted across a network to the client machine. Using SQL as a basis in the database management system on the server solves this problem since only the necessary query response data (a table) is transmitted to the client machine.

Having SQL on the server also allows the database implementation of advanced facilities such as triggers and automatic procedures. As relational DBMS evolve, they will confer the ability to build rules directly into the database engine. Systems that are built with this approach will be more robust than traditional application-based logic approaches.

Although client/server computing is being planned for environments which use mini-computers and mainframes as servers, the largest market likely to develop will have a mix of OS/2, Macintosh, Windows 3.x, Windows NT, and MS-DOS on the client and either UNIX, Windows NT,

NetWare, or OS/2 for the server. Server software will provide mainframe levels of security, recovery, and data integrity capability. Functions such as automatic locking and commit rollback logic, along with deadlock detection and a full suite of data administration utilities, are available on the server side. Another way of looking at this, then, is that SQL client-server technology allows cheap PCs to be made into "industrial strength" computing engines. **GS**

In next month's issue, the second article in this four part series will delve into the features and functionalities of distributed DBMS technology.

<i>CLIENT/WINDOWS 4GLs</i>		
<i>END USER</i>	<i>OCCASIONAL PROGRAMMER</i>	<i>PROFESSIONAL PROGRAMMER</i>
QUEST GUPTA TECHNOLOGIES	DATAEASE DATAEASE	OPEN INSIGHT REVELATION TECHNOLOGIES
ORACLE CARD ORACLE	FOREST & TREES CHANNEL COMPUTING	SQL WINDOWS GUPTA TECHNOLOGIES
OBJECTVISION BORLAND INTERNATIONAL	INFOALLIANCE SOFTWARE PUBLISHING CORP	ORACLE SQL FORMS ORACLE
NOTEBOOK LOTUS	FOCUS INFORMATION BUILDERS	PARADOX BORLAND INTERNATIONAL
Q+E PIONEER SOFTWARE	VISUAL BASIC MICROSOFT	POWERBUILDER POWERSOFT
IMPROMPTU COGNOS	WINDOWS 4GL ASK/INGRES	UNIFACE UNIFACE
		ELLIPSE COOPERATIVE SOLUTIONS
		dBASE IV, Server Edition BORLAND INTERNATIONAL

So Long Ken!...

(continued from page 7)

DEC: An open systems vendor?

Even if DEC's stock prices were higher, there is the question of whether it's believable that a hardware vendor like DEC could create a viable, open software environment. The world is quickly moving to a future where even key software such as operating systems, application development languages, tools, and database management systems

come from independent software vendors rather than the old familiar hardware vendors.

Such imminent proliferation of key vendors stands in stark contrast to the forecasts of the middle 1980s. During the past decade, DB2 was running like a steam roller over the other mainframe DBMS competitors such as ADABAS and IDMS. At that time, it appeared that the future would be dominated by Hewlett Packard, DEC, and IBM, each with a proprietary environment. What was not foreseen by many analysts was

the tremendous move to open and downsized systems. These same analysts also clearly underplayed the fact that smaller, software-only suppliers could create better, more exciting and desirable software than could the industry leviathans.

So Ken, I wish you only the best in your retirement. You had a fabulous career. You made some mistakes, but, haven't we all? Now it's time to relax, enjoy yourself, and maybe even start a new venture. Best wishes. GS

Downsizing DB2...

(continued from page 5)

such as Windows, DOS, or OS/2. And finally, not to leave anything out, communication software is another piece in this puzzle. I see a lot of shops having trouble with these user-related, interdependent decisions. It is difficult to determine which system to pick first.

If you decide to pursue the development of such an environment, it will be easiest to place read-only applications on the PCs as a first step. Applications with update needs can be more difficult to establish on PCs and, therefore, should be implemented at a later date

once your IS shop has more experience the new system.

One of the advantages of slowly and gradually pushing applications down from the mainframe to PCs is that you can proceed at your own rate. Such migration can be non-disruptive if you carefully choose which application to port first. This type of environment is great for query applications, and starts to capitalize on PC hardware and software for more user-friendly environments at lower costs.

For this setup, compatibility is nice, but NOT required. Since the applications are migrated at your own pace, you're not going to necessarily unplug your mainframe DB2 application. Instead, you will

increasingly upgrade the application and put both the upgrades and the data on the network.

There are disadvantages with this form of downsizing. You need to examine how IS is going to partition the data — by location, department, etc. Also, it will be very hard to either estimate or predict the system's performance. There are not many tools available to help you do this. Since this is not a dramatic, instantaneous restructuring of your entire computing system, you can feel confident that you won't run into any serious trouble. Trouble shooting and monitoring can be a concern in such a distributed access

(continued on back page)

A Guide for Making the Transition to Collaborative Distributed Application Development

On April 14 at DCI's Software World in Toronto, Dave Thomas of Object Technology International gave a presentation on distributed applications, and the object paradigm's relevancy in building such applications. Thomas, who is well respected within the object (OO) community, offered some wise and witty suggestions for distributed application developers. Following are some quotes and wisdom:

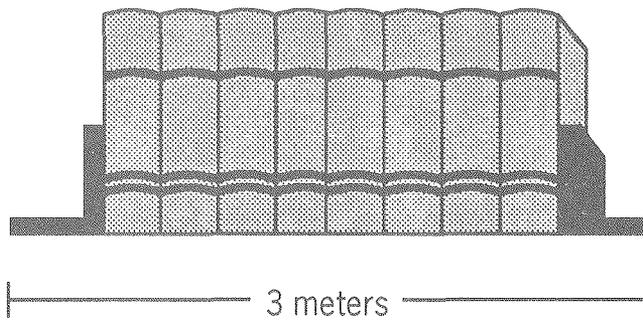
Non-starters

Beware of silver bullet technologies being proposed by OOPSers among others. Their advice requires you to take "non-

starter" actions to succeed. Examples include:

- Replace your main-frame.
- Hire programmers that hold either a MSc or PhD in C.
- Stop maintaining existing systems.
- Rewrite in C all existing applications.

If you decide to follow such advice, you will find yourself competing for available programming talent with the likes of Borland and Microsoft. As a matter of fact, Thomas is currently subcontracting work to India since this level of talent is not available in North America.



UNIX-phobia

Some UNIX facts that Thomas believes everyone should know are as follows:

- UNIX is a four letter word.
- A set of UNIX manuals occupies 3 meters of shelf space.
- Your computer needs 16 MB of RAM to run UNIX successfully.
- Your computer also needs a 20 MIPS CPU

and 150 MB of free disk space to run UNIX successfully.

For those who believe that the combination of UNIX running on X-terminals is the perfect solution, Thomas proposes the following question: While running interactive graphics, how many X-terminals can you support on a 50 MIPS workstation? The answer: 1.

Furthermore, Thomas is convinced that the C++ language is so difficult to learn, that most UNIX programmers aren't capable of mastering it.

CASE-phobia

- CASE has shown itself to be barely adequate for 3GL development, and totally useless for 5GL.
- CASE is useful for turning your best and brightest developers into draftsmen.
- In terms of application development languages, Thomas's favorite successes (in terms of achieving their stated goals are):
 1. RPG
 2. APL
 3. 1-2-3
 4. SQL
 5. Forth (Postscript)
 6. Hypercard

7. Visual Basic
8. Smalltalk

Object oriented

Thomas offered the following logic:

Object Oriented is Good.

God is Good.

∴ God is Object Oriented.

Following are promises that OOPS moonies like to make. According to Thomas, they are all lies:

1. Your first OO project will cost less.
2. Your first OO project will take less time.
3. Your first OO project will produce a library of reusable components.
4. Your first OO project will be infinitely flexible.
5. Your first OO project will satisfy 100% of user requirements.

The OO truths that Thomas does believe include:

- Those who are E/R modelers now will convert and become OO modelers in the future.
- OO modeling without OO implementation is a waste of time.
- OO modeling is most effective when it is enterprise wide.
- An OO commitment is best if it's complete and total for the

environment under consideration.

- For most organizations, it takes a period of years to build a useful library of reusable components and to achieve real productivity gains from OOPS.
- In Thomas's experience, OO developers have had to restructure their class libraries at least *three* times to achieve adequate quality.
- To be most useful (reusable), objects need to be small and plentiful. Some of Thomas's

touch data entry screens.

- Make sure the real user, not just his/her manager, likes the program. (Editor's note: This is a *good* suggestion.)
- Easel's name, to reflect its true personality, should be changed to Chisel.

Thomas's tenet in life

The person who thought up the idea of renaming the existing mess of systems as "legacy" is a marketing genius.

**..Object Oriented is Good.
God is Good.
Therefore,
God is Object Oriented...**

clients have actually developed libraries with thousands of classes.

GULizing

Thomas was skeptical about the benefits to be achieved from the rush to GUIize (his word). He analogized it to the colorization craze of TNT. He offered the following advice:

- GUIize only when it will give you a real competitive advantage.
- Don't use applications that require the user to

Software World

DCI's **Software World** in Toronto drew slightly over 10,000 attendees and is now established as

one of the few major Canadian DP shows. Quite a few of the conference sessions were packed with several hundred to over a thousand attendees.

Next year's **Software World** will be held in Toronto, May 4-6, 1993. The next large DCI trade show scheduled for Canada is **DOWNSIZING EXPO**, October 19-21, 1992. **GS**

Downsizing DB2...

(continued from page 13)

environment. On a mainframe, you can choose between three or four monitoring tools. But, who has the equivalent of those tools that works for a mainframe with a PC LAN plugged into it? *HF*

This is the first article in a two-part series that is based on a lecture given by Howard Fosdick at DATABASE WORLD, Tuesday, June 30, 1992. Mr. Fosdick is a nationally-known industry analyst who has authored six books, including bestsellers on VM/CMS, ISPF Dialog Manager, and the OS/2 Database Manager. He is a popular speaker and lectures internationally on a variety of industry-oriented topics. Mr. Fosdick can be reached at Fosdick Consulting, Inc., Villa Park, IL, (708) 279-4286.

Schussel's

Downsizing Journal

September 1992
Volume 2 Issue 1

SDJ is published monthly by:

Digital Consulting, Inc.
204 Andover Street, Andover, MA 01810 USA
(508) 470-3870 FAX (508) 470-0526

Editor: Dr. George Schussel

Managing Editor: Stacey S. Griffin

Subscription rates:

\$199 annually for U.S. residents
\$225 annually for foreign addresses

Copyright © by *Schussel's Downsizing Journal*. This newsletter is fully protected by copyright. Reproduction or photocopying — even for internal use — without the publisher's permission is PROHIBITED BY LAW. Violators risk criminal penalties and up to \$100,000 damages. *Discounted multiple-copy subscriptions are available.*

For permission to photocopy for internal use or to inquire about special reprints, contact Stacey Griffin, *Schussel's Downsizing Journal*.

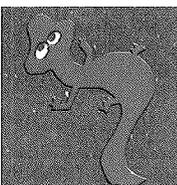
UPCOMING downsizing Events...

Pen-based computing is one of the revolutions for the 1990s, and promises to be as important as the PC revolution was a decade ago. Industry experts predict that within a few years, more than half of the laptop computers sold in the U.S. will be pen-based. All the information you need on this new field will be available at the **Pen-Based EXPO**, being held September 21-23, 1992, in Los Angeles. Keynote presentations will be given by Ed Yourdon, Portia Isaacson, and Kirk Cruikshank.

Being held in Washington D.C., September 21-23, 1992, is **Windows Client/Server Workshop, Test Driving Tools for the New Application Development Environments: Windows, Presentation Manager, Apple, Motif, and OpenLook**. This workshop is based on the premise that the evaluation of applications development software is strictly a qualitative process. The kick off of this three day event will be the Chairman's Address by Jeff Tash. Additional keynote presentations will be made by Larry DeBoever, Pieter Mimno, and Ted Klein.

Coming to Canada for the first time is **Downsizing EXPO Canada**. This conference/exposition will be held at the Sheraton Center in Toronto, October 19-21, 1992, and will feature over 75 different speakers participating in five separate conferences: Downsizing, Client/Server, Business Re-engineering, Interoperability, and Windows. Running concurrently will be the Solutions EXPO with over 50 participating vendors.

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



For your enjoyment and ease of reference, like a chameleon, **SDJ** will be changing its colors monthly. Our changing color scheme will make it easy to spot each new issue at a glance, and quick to reference past issues.