NEAR EAST UNIVERSITY

Faculty of Engineering

Department of Computer Engineering

CLIENT SERVER

Graduation Project
COM-400

Student: Mohammad Salman (991.577)

Supervisor: Miss Besime Erin

Nicosia - 2002

ACKNOWLEDGMENT

"First I would like to thank my supervisor Miss Besime Erin/or her invaluable advice and belief in my work and my self over the course of this graduation project.

Second, I would like to give special thanks to all those who bared with me, and for those taught me the true meaning of perseverance, and I would like to thank my teachers, I am forever in their gratitude for having the believe in me and not giving upon me. I am very grateful to all the people in my life, who have supported me, advised me, taught me and who have always encouraged me to follow my dreams.

Third, I would like to thank my family specially my parents, my brothers, my sister.for moral and material supporting, and to help me to complete my study and to become an engineer.

Finally, I would also like to thank all my friends for their advice and support."

ABSTRACT

Client/server is a computational architecture that involves client processes requesting service from server processes.

Client/server computing is the logical extension of modular programmmg. Modular programming has as its fundamental assumption that separation of a large piece of software into its constituent parts ("modules'1) creates the possibility for easier development and better maintainability. Client/server computing takes this a step farther by recognizing that those modules need not all be executed within the same memory space. With this architecture, the calling module becomes the "client" (that which requests a service), and the called module becomes the "server" (that which provides the service).

The logical extension of this is to have clients and servers running on the appropriate hardware and software platforms for their functions. For example, database management system servers running 011 platforms specially designed and configured to perform queries, or file servers running on platforms with special elements for managing files. It is this latter perspective that has created the widely-believed myth that client/server has something to do with PCs or Unix machines.

Example of client/server systems:

1) Finger
2) NFS (Network File System)
3) X Window System
4) World Wide Web

TABLE OF CONTENTS

AKNOWLEGMENT ı

ABSTRACT u

TABLE OF CONTENTS ın

INTRODUCTION 1

CHAPTER 1 UNDERSTANDING CLIENT/SERVER

1.1 Wlıat is client and server system? I

1.1.1 Client process 2

1.1.2 Server process 2

1.2 Internet 3

1.3 Client/server technical issues 3

1.3.1 The different types of servers 4

1.3.2 The different client/server processing styles 5

1.4 Client/server fundamentals 7

1.4.1 Introduction 7

1.4.2 The basic feature ofC/S model 8

1.5 Middleware 10

1.6 Client/server architecture 12

1.6.1 Two-Tier architecture 12

1.6.2 Three-Tier architecture 13

CHAPTER 2 CLIENT/SERVER COMPUTING

2.1 Introduction 15

2.1.1 Wlıat is the client program do? 20

2.1.2 What is the server do? 20

2.1.3 Mainframe centric client/server computing 21

2.1.4 Downsizing and client/server computing 23

2.1.5 The real word of client/server development tools 24

2.2 Terminology 25
2.2.1 Client/server computer model 25
2.2.2 Disadvantages of client/server environment 26
2.2.3 Feature and characteristics 26
2.3 Client/server application architecture 26
2.3.1 Client/server application architecture types 27
2.4 Distributed approach to client/server computing 28
2.4 .1 Advantages of web-based distributed approach 28
2.5 application frameworks 30
2.5.1 The traditional client/server approach 30
2.5 .2 The distributed object approach 31
2.5 .3 The web-based distributed object approach 32
2.5 .4 Disadvantages of web-based distributed objects 33
2.6 General approach 35
2.7 Technology example 36
2.8HTML 38
2.8.1 Scripting and DHTML 38
2.8.2 Components 38
2.8.3 WIN 32 API 39

CHAPTER 3 CLIENT/SERVER APPLICATION THE SERVER
3 .1 Introduction 40
3 .2 The rule of server 41
3 .3 Server functionality in details 42
3 .3 .1 Request processing 42
3.3.2 File services 42
3 .3 .3 Fax/Print/Image services 43
3 .3 .4 Data Base services 44
3 .3 .5 Communication services 47
3 .3 .6 Security services 47
3 .4 The Network operating system 47

3.5 Tlıe available platforms 49
3.6 The server operating system 50
3.6.1 NetWare 50
3.6.2 O/S2 51
3.63 Windows NT 51
3.6.4MVS 51
3.6.5 OPBNVMS 52
3.6.6UNIX 52
3.6.7 Distributed computing environment (DCB) 54
3.7 System application arcliitecture(SSA} *56*

CHAPTER 4 CLIENT/SERVER APPLICATION THE CLIENT
4.1 Introduction 58
4.2 The rule of client *59*
4.3 Client services 60
4.4 Request for service 62
4.4.1 Remote Procedure Call (RPC) 64
4.4.2 Fax/Print services 64
4.4.3 Windows services 64
4.4.4 Remote boot services 65
4.4.5 Other remote services 65
4.4.6 Data base services 66
4.4.7 Network management services-alerts 67
4.4.8 Dynamic data exchange (DCB) 67
4.4.9 Object linking and embedding (OLE) 67
4.4.10 common object request broker architecture (CORBA) 68

CHAPTER 5 THE FUTURE AND ADVANTAGES OF CLIENT/SERVER COMPUTYING
5.1 The future of client/server computing 69
5.1.1 Store of networking every one's a peer 69

5.1.2 Store of software development every thing's an object 70
5.1.3 Enabling teclınology 70
5.2 Transformational systems 75
5.2;} Emergency public safety 76
5.2.2 Electronic data interchange 76
5.2.3 Financial analyses 77
5.3 Advantages of client/server computing 78
5.3.1 The advantage of client/server computing 79
5.3.2 Technology revolution 84
5.3.3 Connectivity 91
5.3.4 User productivity 93
5.3.5 Ways to improve performance 93
5.3.6 Vendor independence 95
5.3.7 Faster delivery of systems 95
CONCLUSION
REFERENCES

INTRODUCTION

The term client/server was first used in the 1980s in reference to personal computers (PCs) on a network. The actual client/server model started gaining acceptance in the late 1980s. The client/server software architecmre *is* a versatile, message-based and modular infrastructure that is intended to improve usability, flexibility, interoperability, and scalability as compared to centralized, mainframe, time sharing computing. Today, client/server has a very specific meaning tlıat has nothing to do with hardware at all. Even so, most folks still associate the tenn client with the physical computers they are using and the term server with some other physical computer to which they are connected. There's a glimmer of truth there, to be sure, but today client/server means much more.

A client is defined as a requester of services and a server is defined as the provider of services. A single machine can be both a client and a server depending on the software configuration, for details on client/server software architecture. We're going to cover the essentials of client/server computing in this project. The project consists of introduction five chapters and conclusion.

Chapter One "Understanding Client/Server" It's an introduction to client/server presents the client/server process, fundamentals, and its architecture.

Chapter Two "Client/Server Computing" presents the client/server programming, compuring model, application frameworks and some tecliıiology example about client/server.

Chapter Three "The Component of Client/Server Application the Server" its presents the rule of server and server services.

Chapter Four "The Component of Client/Server Application the Client" its presents the rule of client and client services.

Chapter Five "The Future and Advantages of Client/Server Computing" describe some of future and advantages for the client/server.

CHAPTER!

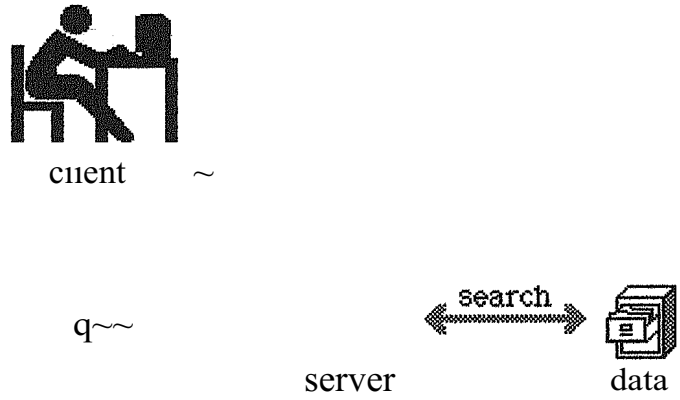
UNDERSTANDING CLIENT/SERVER

1.1 What is client and server system?

The principle is that the user has a client program. He asks information (or data) from the server program. The server searches the data and sends it back to the client.[l] Putting in another way, we can say that the user is the client, he uses a client program to start a client process, sends message to server which is a server program, to perform a task or service.

As a matter of fact a client server system is a special case of a co-operative computer system. All such systems are characterized by the use of multiple processes that work together to form the system solution. (There are two types of co-operative systems client-server systems and peer-to-peer systems.).

The client and server systems consist of three major components: a server with relational database, a client with user interface and a network hardware connection in between. Client and server is an open system with number of advantages such as interoperability, scalability, adaptability, affordability, data integrity, accessibility, performance and security.



Figuref.I client/serverrelation

1.1.1 Client process

The client is a process (program) that sends a message to a server process (program), requesting that the server perform a task (service). Client programs usually manage the user-interface portion of the application, validate data entered by the user, dispatch requests to server programs, and sometimes execute business logic. The client-based process is the front- end of the application that the usersees and interacts with. The client process contains solution-specific logic and provides the interface between the user and the rest of the application system. The client process also manages the local resources that the user interacts with such as the monitor, keyboard, workstation CPU and peripherals. One of the key elements of a client workstation is the graphical user interface (GUI). Normally a part of operating system i.e. the window manager detects user actions, manages the windows on the display and displays the data in the windows.

1.1.2 Server process

A server process (program) fulfills the client request by performing the task requested. Server programs generally receive requests from client programs, execute database retrieval and updates, manage data integrity and dispatch responses to client requests. Sometimes server programs execute common or complex business logic. The server-based process "may" run on another machine on the network. This server could be the host operating system or network file server; the server is tlıen provided both file system services and application services. Or in some cases, another desktop machine provides the application services. The server process acts as a software engine that manages shared resources such as databases, printers, communication links, or high powered-processors. The server process performs the back-end tasks that are common to similar applications.

1.2 Internet

The explosion of the World Wide Web is due to the world-wide acceptance of a common transport (TCP/IP), server standard (HTTP), and markup language (HTML). Many corporations have discovered that these same technologies can be used for internal client/server applications with the same ease that they are used on the Internet. Thus was bom the concept of the "Intranet": the use of Internet technologies for implementing internal client/server applications.

One key advantage of Web-based intranets is that the problem of managing code on the client is greatly reduced. Assuming a standard browser on the desktop, all changes to user interface and functionality can be done by changing code on the HTTP server. Compare this with the cost of updating client code on 2,000 desktops.

A second advantage is that if the corporation is already using the Internet, no additional code needs to be licensed or installed on client desktops. To the user, the internal and external information servers appear integrated.

A rapidly-disappearing disadvantage is that there is limited ability to provide custom coding on the client. In the early days of the Web, there were limited ways of interacting with the client. The Web was essentially "read-only". With the release of code tools such as Java and JavaScript, this limitation is no longer a major issue.

1.3 Client/Server Technical Issues

The basic characteristics of client/server architectures are:

1) Combination of a client or front-end portion that interacts with the user, and a server or back-end poıtion that interacts with the shared resource. The client process contains solution-specific logic and provides the interface between the user and the rest of the application system. The server process acts as a software engine that manages shared resources such as databases, printers, modems, or high powered processors.

2) The front-end task and back-end task have fundamentally different requirements for computing resources such as processor speeds, memory, disk speeds and capacities, and input/output devices.

3) The environment is typically heterogeneous and multi vendor. The hardware platform and operating system of client and server are not usually the same. Client and server processes communicate through a well-defined set of standard application program interfaces (API's) and (RPC).

4) An important characteristic of client-server systems is scalability. They can be scaled horizontally or vertically. Horizontal scaling means adding or removing client workstations with only a slight performance impact. Vertical scaliiig means migrating to a larger and faster server machine or multi servers.

1.3.1 The different types of servers

The simplest forms of servers are disk servers and file servers. With a file server, the client passes requests for files or file records over a network to the file server. This form of data service requires large bandwidth and can slow. a network with many users down considerably. Traditional LAN computing allows users to share resources, such as data files and peripheral devices, by moving them from standalone PCU on to a Networked File Server (NFS).

The more advanced form of servers is the database servers, transaction server and application servers. In database servers, client's passes SQL (Structured Query Language) requests as messages to the server and the results of the query are returned over the network. The code that processes the SQL request and the data resides on the server allowing it to use its own processing power to find the requested data, rather than pass all the records back to a client and let it find its own data as was the case for the file server. In transaction servers, clients invoke remote procedures that reside on servers which also contains an SQL database engine. There are procedural statements on the server to execute a group of SQL statements (transactions) which either all succeed or fail as a unit. The applications based on transaction servers are called On-line Transaction Processing (OLTP) and tend to be mission-critical applications which require 1-3 second response time, 100%

of the time and require tight controls over the security and integrity of the database. The communication overhead in this approach is kept to a minimum as the exchange typically consists of a single request/reply (as opposed to multiple SQL statements in database servers). Application servers are not necessarily database centered but are used to server user needs, suclı as. Download capabilities from Dow Jones or regulating the electronic mail process. Basing resources on a server allows users to share data, while security and management services, which are also based in the server, ensure data integrity and security.

1.3.2 The different client/server processing styles

Gartner group came out with the five ways of describing the different c/s styles based on how they split the three components of any application: user interface, business or application logic, data management. The five styles are distributed presentation, remote presentation, distributed function, remote data management, and distributed data management.

1. Distributed or remote presentation

For people whose roots are embedded in the mainframe IBM world, client-server is essentially distributed or remote presentation. This style maps a workstation Graphical User Interface (GUI) front end onto an existing application's text-based screen. This is also called Remote, Mapping, Front-ending or HLLAPI (High-Level Language Application Programming Interface). The mode of operation is typically intelligent workstations intercepting and interrogating text-screen (e.g. 3270) data streams sent from a host for display in a windowed environment. This is "front ware" solution, where a GUI front end is added to an IBM/MVS 3270/5250 application and is placed on a workstation. However, most processing remains on the host or server, with the exception of user interface translation logic and possibly validation logic. For example, data from an application program is sent to a 3270 screen program on the mainframe to be displayed. The merged data is sent to the workstation as a 3270 data stream. The workstation interprets the data and converts it to graphical form in a window. (Typically every mainframe screen used by the application has a corresponding window on the workstation and vice versa). When the

user enters the data in a GUI window, it is transformed by tlıe front ware application into a corresponding 3270 data format and is sent to tlıe host computer.

2. Remote Data Management

In remote data management, th.e entire application resides on tlıe client and tlıe data management is located on a remote server/host. Remote Data Management is relatively easily to program for because th.ere is just one application program. The client communicates with. tlıe server using SQL, the server then responds with data that satisfies the query. RDBMS products th.at offer remote data management provide a layer of software on the client to handle the communication with the DBMS server. This style represents a more traditional LAN database server or file server approach. Workstations support tlıe presentation and function logic and interface with the data server through th.e data manipulation language. Distributed data management is an extension of remote data management and uses the distributed facilities of tlıe DBMS to access distributed data in a manner transparent to users. This is most relevant for architectures having data spread across several servers and when access to a DBMS on another server is required.

3. Distributed function processing

Here the split occurs in the application functionality, one part going to the client, other to tlıe server. Distributed function applications are the most complex of the three typologies since two separately compiled application programs must be developed. Developers must analyze where each function should reside and what type of dialog must occur between the two programs. The underlying communications facilities may implement either a message-based or remote procedure call (RPC) mechanism for transfer of dialog and data. However, there are many variants of this typology. One variant oftlıe distributed function style is where data management and application function occur at both tlıe client and server.

1.4 Client/Server Fundamentals

1.4.1 Introduction

Simply stated, an object-oriented client/server Internet (OCSI) environment provides the IT infrastructure (i.e., middleware, networks, operating systems, hardware) that supports the OCSI applications breed of distributed applications of particular interest to us.

Figure 1.2 will serve as a general :framework for discussion. This :framework, illustrate the role of the following main building blocks of OCSI environments:

- . Client and server processes (applications) that represent the business logic as objects that may reside on different machines and can be invoked through Web services
- . Middleware that supports and enables the OCSI applications (see the sidebar "What is Middleware?")
- . Network services that transport the information between remote computers
- . Local services (e.g., database managers and transaction managers)
- . Operating systems and computing hardware to provide the basic scheduling and hardware services

We will quickly scan these building blocks and illustrate their interrelationships in multi vendor environments that are becoming common to support enterprise wide distributed applications.

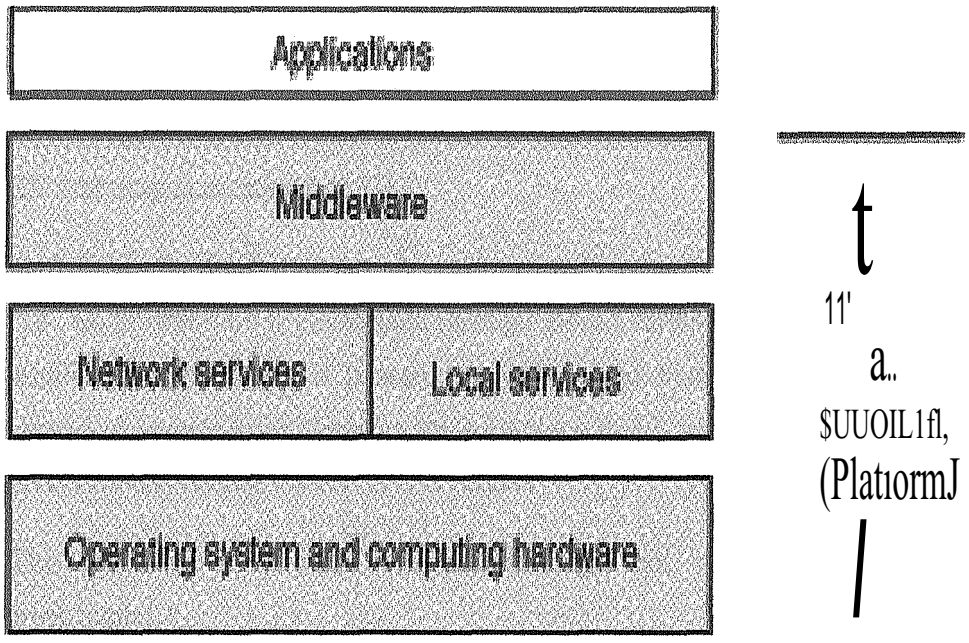


Figure 1.2 Object-Oriented Client/Server Internet Environments

Client/server model is a concept for describing communications between computing processes that are classified as service consumers (clients) and service providers (servers). Figure 1.3 presents a simple C/S model.

1.4.2 The basic features of a C/S model are:

1) Clients and servers are functional modules with well defined interfaces (i.e., they hide internal information). The functions performed by a client and a server can be implemented by a set of software modules, hardware components, or a combination thereof. Clients and/or servers may run on dedicated machines, if needed. It is unfortunate that some machines are called "servers." This causes confusion (try explaining to an already bewildered user that a client's software is running on a machine called "the server"). We will avoid this usage as much as possible.

2) Each client/server relationship is established between two functional modules when one module (client) initiates a service request and the other (server) chooses to respond to the service request. Examples of service requests (SR) are "retrieve customer name," "produce net income in last year," etc. For a given service request,

clients and servers do not reverse roles (i.e., a client stays a client and a server stays a server). However, a server for SR R1 may become a client for SR R2 when it issues requests to another server (see Figure 1.3). For example, a client may issue an SR that may generate other SR.

3) Information exchange between clients and servers is strictly through messages (i.e., no information is exchanged through global variables). The service request and additional information is placed into a message that is sent to the server. The server's response is similarly another message that is sent back to the client. This is an extremely crucial feature of C/S model.

The following additional features, although not required, are typical of a client/server model:

4) Messages exchanged are typically interactive. In other words, C/S model does not support an off-line process. There are a few exceptions. For example, message queuing systems allow clients to store messages on a queue to be picked up asynchronously by the servers at a later stage.

5) Clients and servers typically reside on separate machines connected through a network. Conceptually, clients and servers may run on the same machine or on separate machines. In this book, however, our primary interest is in *distributed client/server systems* where clients and servers reside on separate machines.

The implication of the last two features is that C/S service requests are real-time messages that are exchanged through network services. This feature increases the appeal of the C/S model (i.e., flexibility, scalability) but introduces several technical issues such as portability, interoperability, security, and performance.

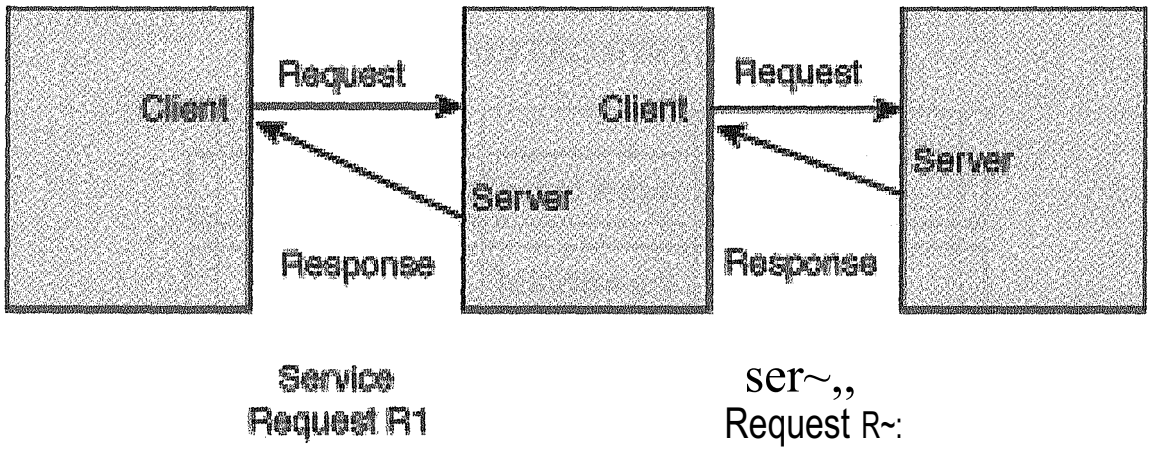


Figure 1.3 Conceptual Client/Server Model

1.5 Middleware

Middleware is a crucial component 0f modem IT infrastructure. We will use the following definition of middleware:

Deflnition: Middleware is a set of common business-unaware services that enable applications and end users to interact with each other across a network. In essence, middleware is the software that resides above the network and below the business-aware application software.

The services provided by these routines are available to the applications through application programming interfaces (APis) and to the human users through commands and/or graphical user interfaces (GUis).

A common example of middleware is e-mail because it provides business-unaware services that reside above networks and interconnect users (in several cases applications also). Other examples are groupware products (e.g., Lotus Notes), Web browsers, Web gateways, SQL gateways, Electronic Data Interchange (EDI) packages, remote procedure call (RPC) packages, and "distributed object servers" such as CORBA We will briefly discuss these middleware components in this chapter.

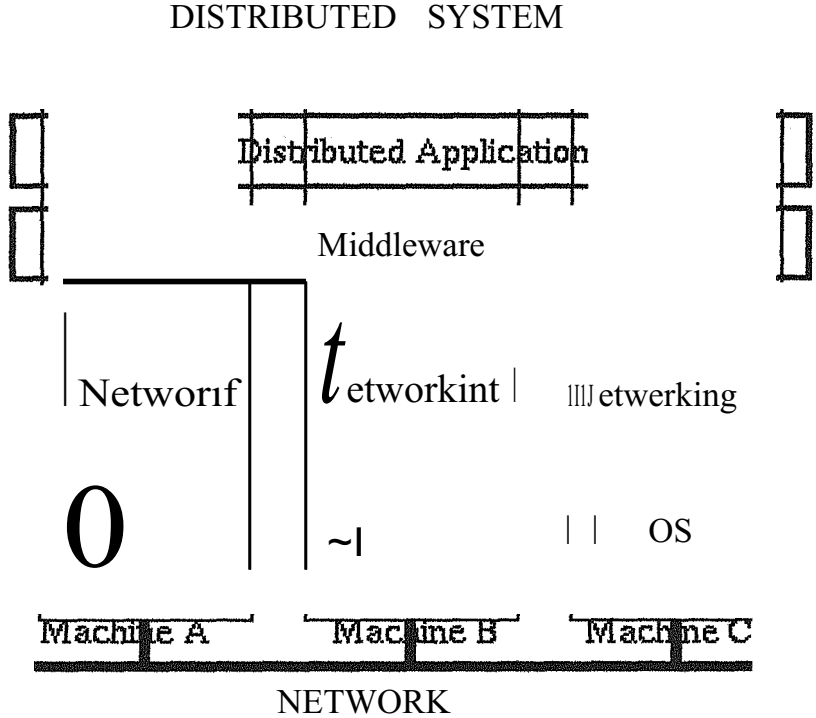


Figure 1.4 middleware

Client/server applications, an area of vital importance to us, employ the C/S model to deliver business aware functionalities. C/S applications provide a powerful and :flexible mechanism for organizations to design applications to fit the business needs. For example, an order processing application can be implemented using the C/S model by keeping the order processing databases (customers, products) at the corporate office and developing/customizingthe order processing logic and user interfaces for different stores that initiate orders. In this case, order processing clients may reside on store computers to perform initial checking and preprocessing, and the order processing servers may exist at the corporate mainframe to perform final approval and shipping. Due to the critical importance of C/S applications to business enterprises of the 1990s and beyond, we will focus on C/S applications in this book.

1.6 Client/Server Architectures

Client/server architecture provides the fundamental framework that allows many technologies to plug in for the applications of 1990s and beyond. Clients and servers typically communicate with each other by using one of the following paradigms:

Remote Procedure Call (RPC). In this paradigm, the client process invokes a remotely located procedure (a server process), the remote procedure executes and sends the response back to the client. The remote procedure can be simple (e.g., retrieve time of day) or complex (e.g., retrieve all customers from Chicago who have a good credit rating). Each request/response of an RPC is treated as a separate unit of work, thus each request must carry enough information needed by the server process. RPCs are supported widely at present.

Remote Data Access (RDA). This paradigm allows client programs and/or end-user tools to issue ad hoc queries, usually SQL, against remotely located databases. The key technical difference between RDA and RPC is tlıat in an RDA the size of the result is not known because the result of an SQL query could be one row or thousands of rows. RDA is heavily supported by database vendors.

Queued Message Processing (QMP): In this paradigm, the client message is stored in a queue and the server works on it when free. The server stores ("puts") the response in another queue and the client actively retrieves ("gets") the responses from this queue. This model, used in many transaction processing systems, allows the clients to asynchronously send requests to the server. Once a request is queued, the request is processed even if the sender is disconnected (intentionally or due to a failure). QMP support is becoming commonly available.

1.6.1 Two-Tier Architecture

Two-tier architecture is where a client talks directly to a server, with no intervening server. It is typically used in small environments (less than 50 users).

A common error in client/server development is to prototype an application in a small, two-tier environment, and then scale up by simply adding more users to the server. This approach will usually result in an ineffective system, as the server becomes overwhelmed. To properly scale to hundreds or thousands of users, it is usually necessary to move to a three - tier architecture.

1.6.2 Three- Tier Architecture

Three-tier architecture introduces a server (or an "agent") between the client and the server. The role of the agent is many fold. It can provide translation services (as in adapting a legacy application on a mainframe to a client/server environment), metering services (as in acting as a transaction monitor to limit the number of simultaneous requests to a given server), or intelligent agent services (as in mapping a request to a number of different servers, collating the results, and returning a single response to the client.

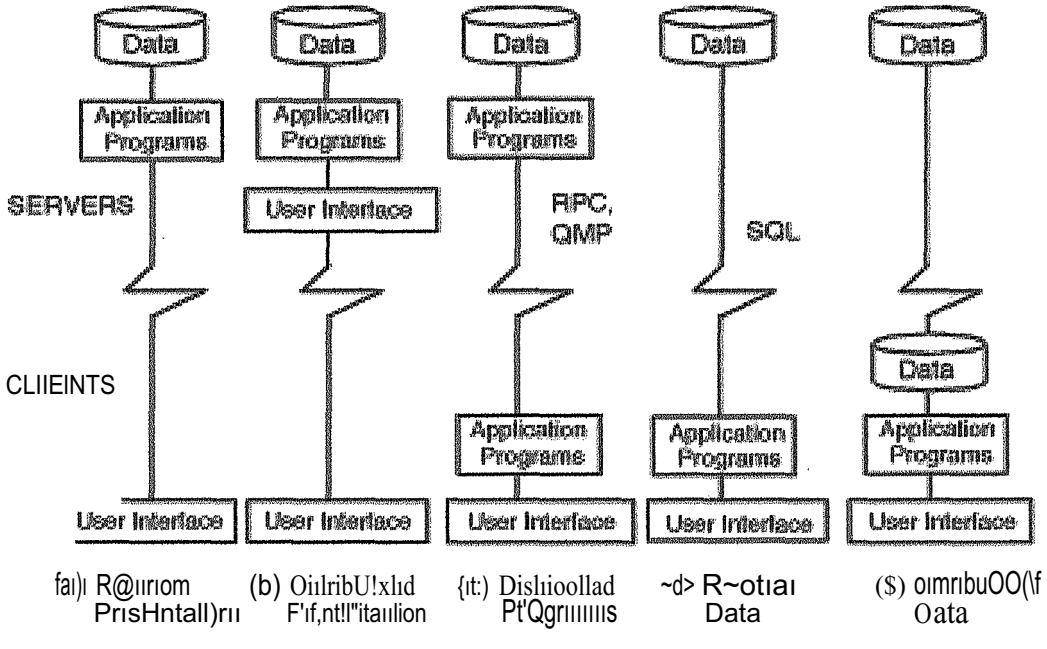


Figure 1.5 Traditional Client/Server Architectures

Although a given C/S application can be architected in any of these configurations, the remote data and distributed program configurations are used heavily at present. The

remote data configuration at present is very popular for departmental applications and is heavily supported by numerous database vendors (as a matter of fact this configuration is used to represent typical two-tiered architectures that rely on remote SQL). Most data warehouses also use a remote data configuration because the data warehouse tools can reside on user workstations and issue remote SQL calls to the data warelıouse. However, the distributed programs configuration is very useful for enterprise wide applications, because the application programs on both sides can exchange information through messages.

CHAPTER2

CLIENT/SERVER COMPUTING

2.1 Introduction

The single-system image is best implemented through the client/server model. Our experience confirınsthatcHent!server computing can provide the enterprise to the desktop .. Because tlıe deskt&tfcôıtıpttteris' tlıe· user,s view int<r tlıe, enterprise, there is no better way to guarantee a single: imağe,tlıaıı.tô start atthetdesktop:Unfôıtunately, it often seems as .if the number of definitions of client/server computing depends on how many organizations you survey, whether they're hardware and software vendors, integrators, or IS groups. Each has a vested interest in a definition that makes its particular product or service an indispensable component.

Throughout this book, the following definitions will be used consistently:

- *Client:* A client is a single-user workstation that provides presentation services and the appropriate computing, connectivity, and database services and interfaces relevant to the business need.
- *Server:* A server is one or more multi user processors with shared memory providing computing, connectivity, and database services and interfaces relevant to the business need.

Client/server computing is an environment that satisfies the business need by appropriately, allocating the application processing between the client and the server processors. The client requests services from the server; the server processes the request and returns the result to the client. The communications mechanism is a message passing inter process communication (IPC) that enables (but does not require) distributed placement of the client and server processes. Client/server is a software model of computing, not a hardware definition.

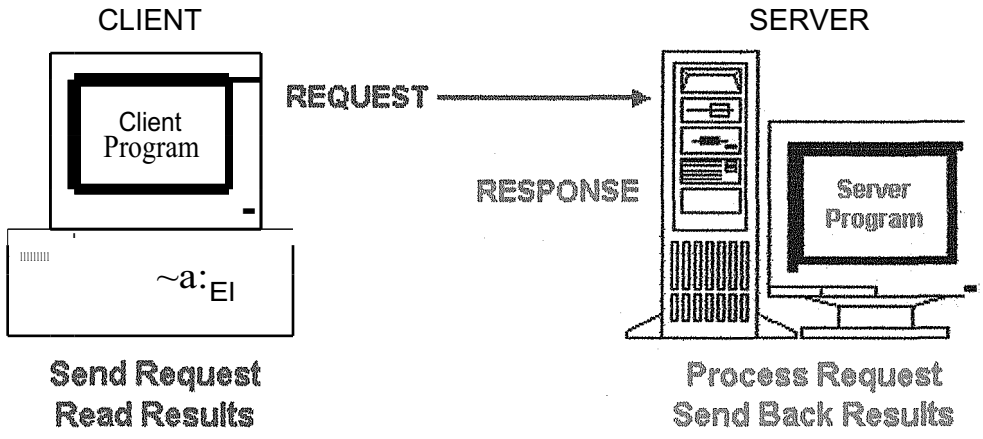


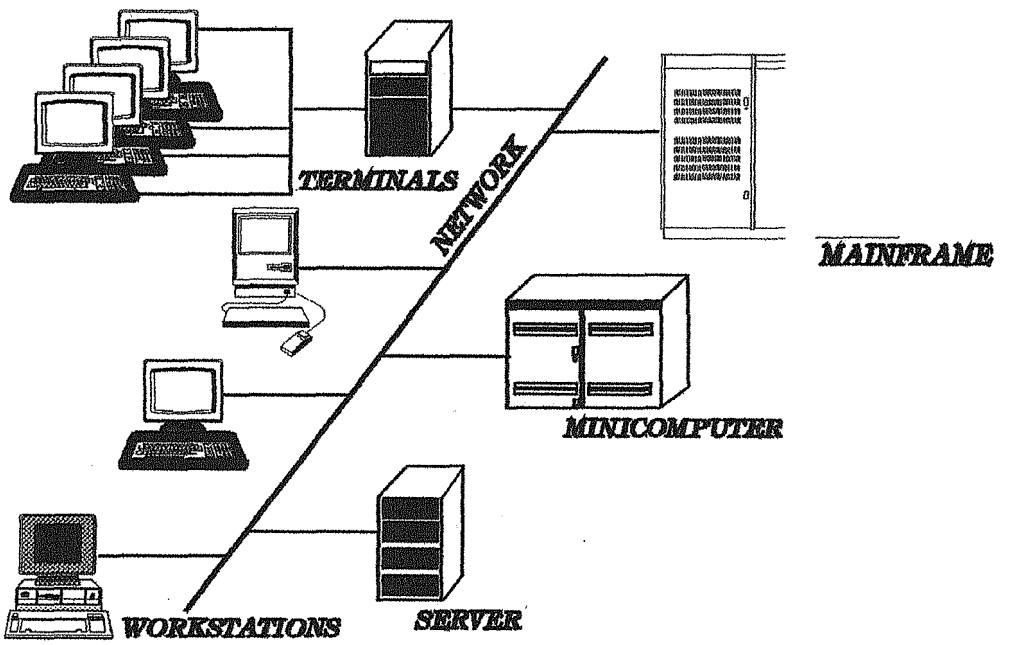
Figure 2.1 client server

Because the client/server environment is typically heterogeneous, the hardware platform and operating system of the client and server are not usually the same. In such cases, the communications mechanism may be further extended through a well-defined set of standard application program interfaces (APis) and remote procedure calls (RPC).

The modern diagram representing the client/server model was probably first popularized by Sybase. A client-user relies on the desktop workstation for all computing needs. Whether the application runs totally on the desktop or uses services provided by one or mote servers be they powerful PCs or mainframes is irrelevant.

Effective client/server computing will be fundamentally platform-independent. The user of an application wants the business functionality it provides; the computing platform provides access to this business :functionality.There is no benefit, yet considerable risk, in exposing this platform to its user.

Changes in platform and underlying technology should be transparent to the user. Training costs, business processing delays and errors, staff frustration, and staff turnover result from the confusion generated by changes in environments where the user is sensitive to the technology platform.



Figu.re2.2 Modem client/server architecture

It is easily demonstrated that systems built with transparency to the technology, for all user, offer the highest probability of solid ongoing return for the technology investment It is equally demonstrable that if developers become aware of the target platform, development will be bound to that platform. Developers will. use special features, tricks, and syntax found only in the specific development platform.

The use of technology layers provides this application development isolation. These layers isolate the characteristics of the technology at each level from the layer above and below. This layering is fundamental to the development of applications in the client/server model. The rapid rate of change in these technologies and the lack of experience with the "best" solutions; implies that we must isolate specific technologies from each other. This book will continue to emphasize and expand on the concept of a systems development environment (SOE) as a way to achieve this isolation. Figure 2.3 illustrates the degree of visibility to specific technology components required by the developers.

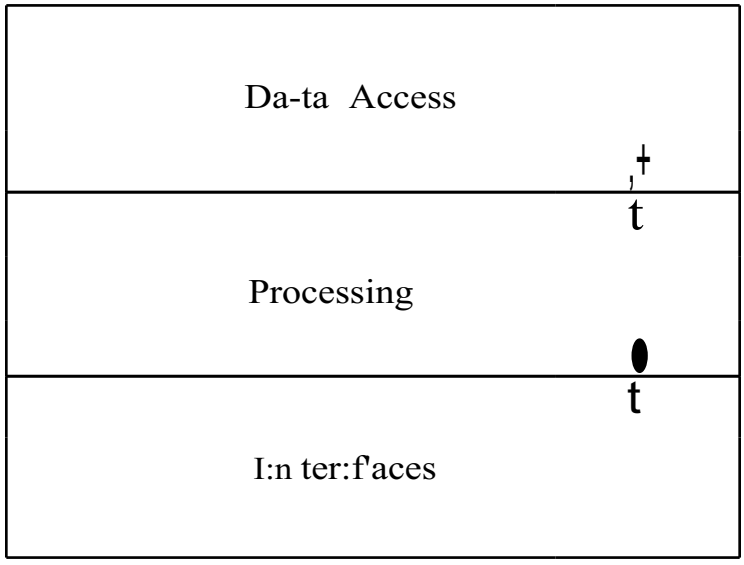


Figure 2.3 Simplifiedapplication model

Developer tools are by far the most visible. Most developers need to know only tlıe syntax of these tools to express the business problem in a format acceptable to the technology platform. With the increasing involvement of noıı computer professionals, as technology users and application assemblers, technology isolation is even more important. Very few perhaps none of an organization's application development staff needs to be aware of the hardware, system software, specific database engines, specific communications products, or specific presentation services products. These are invoked through the APls message passing, and RPC generated by tools or by a few technical specialists.

As organizations increase the use of personal productivity tools, workstations become widely installed. The need to protect desktop real estate requires that host terminal capabilities be provided by the single workstation. It soon becomes evident that the power of the workstation is not being tapped and application processing migrates to the desktop. Once most users are connected from their workstation desktop to the applications and data at the host mainframe or minicomputer, there is significant cost benefit in offloading processing to these powerful workstations. The first applications tend to be data capture and edit. These simplify but still use the transaction expected by an already existing host application. If the workstation is to become truly integrated with the application, reengineering of the business process will be necessary. Accounting functions and many

customer service applications are easily offloaded in this manner. Thus, workgroup and departmental processing is done at the LAN level, with host involvement for enterprise-wide data and enforcement of interdepartmental business rules.

Figure 2.4 illustrates the existing environment in many organizations wherein desktop workstations have replaced the unintelligent terminal to access existing host-based applications.

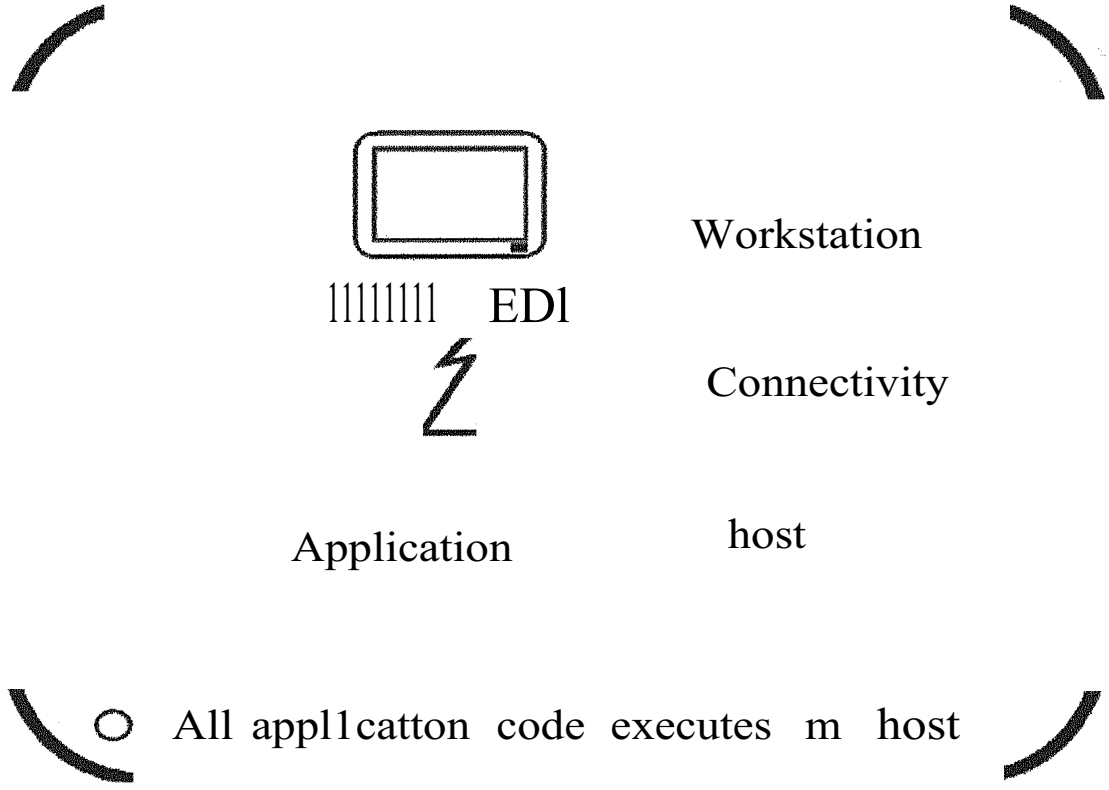


Figure 2.4 Existing environment

In this "dumb" terminal (IBM uses the euphemism nonprogrammableto describe its 327x devices) emulation environment, all application logic resides in the minicomputer, mainframe, or workstation. Clearly a $5000 or less desktop workstation is capable of much more than the character display provided by a $500 terminal. In the client/server model, tlıe low-cost processing power of the workstation will replace host processing, and the application logic will be divided appropriately among the platforms. As previously noted, this distribution of function and data is transparent to the user and application developer.

2.1.1 What is the client program do?

They usually deal with;

- Managing the application's user-interface part
- Confirming the data given by the user
- Sending out the requests to server programs
- Managing local resources, like monitor, keyboard and peripherals.

The client-based process is the application that the user interacts with. It contains solution-specific logic and provides the interface between the user and the rest of the application system. In this sense the graphical user interface (GUI) is one characteristic of client system.

Administration Tool: for specifying the relevant server information, creation of users, roles and privileges, definition of file formats, document type definitions (DTD) and document status information. Template Editor: for creating and modifying templates of documents. Document Editor: for editing instances of documents and for accessing component information. Document Browser: for retrieval of documents from a Document Server. Access Tool: provides the basic methods for information access.

2.1.2 What is the server do?

Its purpose is fulfilling client's requests. What they do in general is:

- Receive request
- Execute database ret rival and updates
- Manage data integrity
- Sent the result to client back

When the servers do these, it can use common or complex logic. It can supply the information only using its own resources or it can employ some other machine on the network, in master and slave attitude.

In other words, there can be,

- single client. single server
- multiple clients, single server
- single client, multiple servers
- multiple clients, multiple servers

In a client server system, we can talk about specialization of some compoııeııts for particular tasks. The specialization can provide very fast computation servers or high throughput database servers, resilient transaction servers or for some other purpose, nevertheless what is important is that they are optimized for that task. It is not optimal to try to perform all of the tasks together. Actually it is this specialization, therefore optimization that gives power to client and server systems.

2.1.3 Mainframe Centric Client/Server Computing

The mainframe-centric model uses the presentation capabilities of the workstation to front-end existing applications. The character mode interface is remapped by products such as Easel and Mozart. The same data is displayed or entered through the use of pull-down lists, scrollable fields, check boxes, and buttons; the user interface is easy to use, and information is presented more clearly. In this mainframe-centric model, mainframe applications continue to run unmodified, because the existing terminal data stream is processed by the workstation-based communications API.

In contrast, the workstation GUI provides facilities to build the screen dynamically. This enables screens to be built with a variable format based conditionally on the data values of specific fields.

Although it is a limited use of client/server computing capability, a GUI front end to an existing application is frequently the first client/server-like application implemented by organizations familiar with the host mainframe and dumb-terminal approach. The GUI preserves the existing investment while providing the benefits of ease of use associated with a GUI. It is possible to provide dramatic and functionally rich changes to the user interface without host application change.

Figure 2.5 illustrates how multiple applications can be integrated. in this way.. The data is available to authorized users as soon as it is captured. There is no delay while the forms are passed around the organization. This is usually a better technique tlıan *forms imaging technology* in which the forms are created and distributed internally in an organization. The use of work flow management technology and techniques.

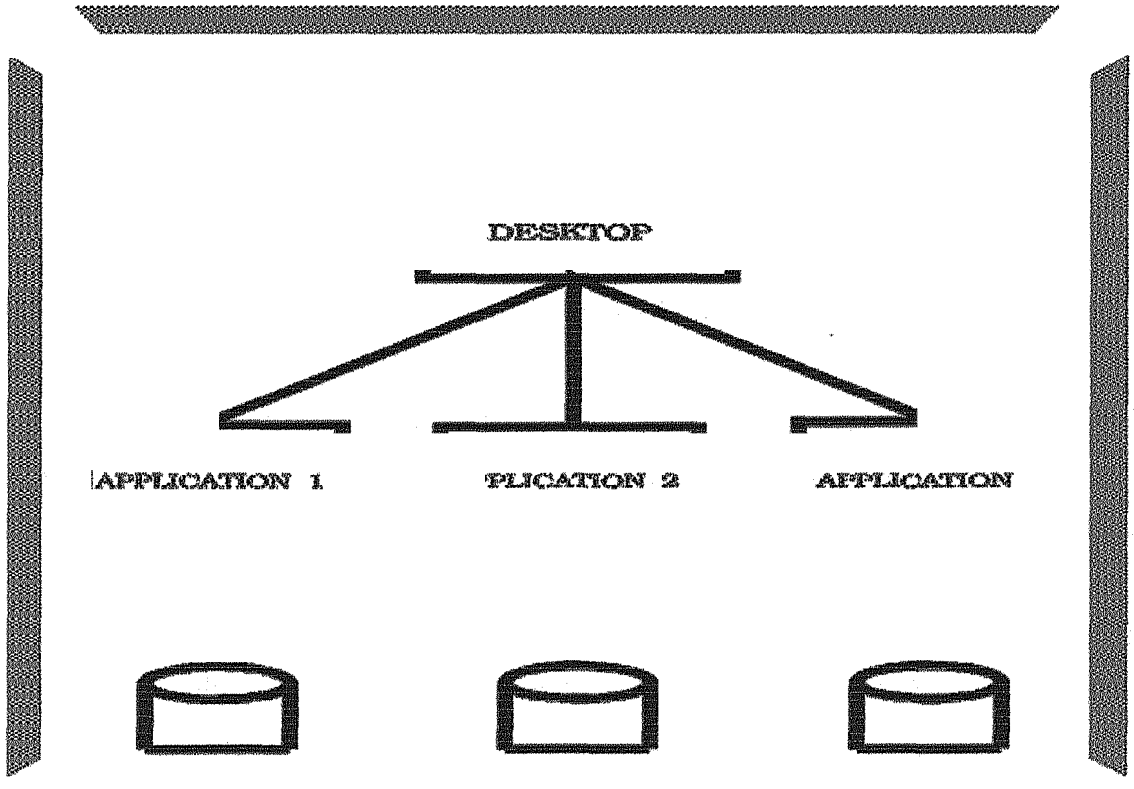


Figure2.5 Desktop application integration

Intelligent Character Recognition (ICR) technology can be an extremely effective ways to automate the capture of data from a form, without the need to key. Current experience with this technique shows accuracy rates greater than 99.5 percent for typed forms and greater than 98.5 percent for handwritten forms.

2.1.4 Downsizing and Client/Server Computing

Rightsizing and downsizing are strategies used with the client/server model to take advantage of the lower cost of workstation technology. Rightsizing and upsizing may involve the addition of more diverse or more powerful computing resources to an enterprise computing environment. The benefits of right sizing are reduction in cost and/or increased functionality, performance, and flexibility in the applications of the enterprise. Significant cost savings usually are obtained from a resulting reduction in employee, hardware, software, and maintenance expenses. Additional savings typically accrue from the improved effectiveness of the user community using client/server technology.

Downsizing is frequently implemented in concert with a flattening of the organizational hierarchy. Eliminating middle layers of management implies empowerment to the first level of management with the decision-making authority for the whole job. Information provided at the desktop by networked PCs and workstations integrated with existing host (such as mainframe and minicomputer) applications is necessary to facilitate this empowerment. These desktop-host integrated systems house the information required to make decisions quickly. To be effective, the desktop workstation must provide access to this information as part of the normal business practice. Architects and developers must work closely with business decision makers to ensure tlıat new applications and systems are designed to be integrated with effective business processes. Much of the cause of poor return on technology investment is attributable to a lack of understanding by the designers of the day-to-day business impact of their solutions.

Downsizing information systems is more than an attempt to use cheaper workstation technologies to replace existing mainframes and minicomputers in use. Although some benefit is obtained by this approach, greater benefit is obtained by reengineering the business processes to really use the capabilities of the desktop environment. Systems

solutions are effective only when they are seen by the actual user to add value to the business process.

Client/server technology implemented on low-cost standard hardware will drive downsizing. Client/server computing makes the desktop the users' enterprise. As we move from the machine-centered era of computing into the workgroup era, the desktop workstation is empowering the business user to regain ownership of his or her information resource. Client/server computing combines the best of the old with the new the reliable multi user access to shared data and resources with the intuitive, powerful desktop workstation.

Mainframe applications rarely can be downsized without modifications to a workstation environment. Modifications can be minor, wherein tools are used to *port (*or re host) existing mainframe source code or major, wherein the applications are *rewritten* using completely new tools. In porting, native COBOL compilers, functional file systems, and emulators for DB2, IMS DB/DC, and CICS are available for workstations. In rewriting, there is a broad array of tools ranging from PowerBuilder, Visual Basic, and Access, to larger scale tools such as Forte and Dynasty.

2.1.5 The Real World of Client/Server Development Tools

Many construction projects fail because their developers assume that a person with a toolbox full of carpenter's tools is a capable builder. To be a successful builder, a person must be trained to build according to standards. The creation of standards to define interfaces to the sewer, water, electrical utilities, road, school, and community systems is essential for successful, cost-effective building. We do not expect a carpenter to design such interfaces individually for every building. Rather, pragmatism discourages imagination in this regard. By reusing the models previously built to accomplish integration, we all benefit from cost and risk reduction.

Computer systems development using an SDE takes advantage of these same concepts: Let's build on what we've learned. Let's reuse as much as possible to save development costs, reduce risk, and provide the users with a common "look and feel."

Selecting a good set of tools affords an opportunity to be successful. Without the implementation of a comprehensive SDE, developers will not achieve such success.

The introduction of a whole new generation of Object Technology based tools for client/server development demands that proper standards be put in place to support shared development, reusable code, inter faces to existing systems, security, error handling, and an organizational standard "look and feel." As with any new teclmology, there will be changes. Developers can build application systems closely tied to today's technology or use an SDE and develop applications that can evolve along with the technology platform.

2.2 Terminology

It involves splitting an application into tasks and putting each task on the platform where it can be handled most effectively. Depending on the application and the software used, all data processing may occur on the client or be split between the client and the server. The server is connected to its clients via a network. Server software accepts requests for data from client software and returns the results to the client. The client manipulate; the data and present tlıe results to the user.

2.2.1 Client/server computer model

It implies a cooperative processing of requests submitted by a client to tlıe server which processes the requests and returns the results to the client. In this model, application processing is divided between client and server.

Any application in which the requester of action is on one system and the supplier of action is potentially on another.

Although the applications are dispersed, there is an emphasis on centralizing corporate databases and many network management and utility functions. This enables corporate management to maintain overall control of the total capital investment in computing and information systems, enabling corporate management to provide interoperability so that systems are tied together. At the same time it relieves individual departments and divisions of much of the overhead of maintaining sophisticated computer-based facilities but enables them to choose just about any type of machine and interlace

they need to access data and information. Network management and network security have a high priority in organizing information systems.

2.2.2 Disadvantages of client/server environment

- Maintenance difficulties and compatibility. Some parts may not work together.
- Lack of support tools. With the client-server architecture, support tools must be usually built within departments.
- Training. Different software and technology as well as development models require training and re-training

2.2.3 Features and Characteristics

- Networked web of small, powerful machines
- Open systems
- Scalability
- Individual client operating environment

The central feature of the client/ server architecture is the allocation of application level tasks between clients and servers. hı both client and server the basic software is an operating system running on the hardware platform, The platforms and the operating systems of client and server may differ.

2.3 Client-server applications architecture

The central feature of the client/server architecture is the allocation of application-level tasks between clients and servers. Communications software is the one that enables client and server to inter-operate. Communications software include; but not limited to TCP/IP, OSI and other proprietary architectures. Communications software and underlying operating system are to provide a base for distributed applications. The actual functions performed by the application can be split up between client and server in a way that optimizes platform and network resources and that optimizes the ability of users to

pe4ıform various tasks and to cooperate with one another in suing shared resources. In most cases, requirements are such that the bulk of the applications software executes at the server.

The essential factor in the success of a client-server environment is the way in which the user interacts with the system. Design of a user interface for clients is critical. Thus, there is a very heavy emphasis on providing a mature graphical user interface that is easy to learn and use. The presentation services in the client workstations are responsible for providing a user-friendly interface to the distributed applications available in the environment.

2.3.1 Client-server applications architectures types

The exact distribution of data and application processing depends on the nature of the database, the types of applications, the ability to achieve interoperability, and usage patterns of data.

Some of the possible types of client-server architectures are as follows:

- Server-based processing: The most basic type of client-server configuration is one in which the client is principally responsible for providing graphical user interface. All of the processing is done on tlıe server.

- Client-based processing: All application processing may be done at the client with the exception of data validation routines and other database logic functions that are best performed at the server. Some of the more sophisticated database logic functions may be based on the client side

- Cooperative processing: In a cooperative processing type, the application processing *is* performed in an optimized fashion, taking advantage of the strengths of both client. and server machines and of the distribution of data. Such types of configurations are more complex to set up and maintain. But this type of configuration may offer greater flexibility and user productivity and greater network efficiency.

2.4 Distributed approach to client/server computing

- One or more server-oriented software modules encapsulated into "business objects". Each business object will generally have a single, well-defined purpose, and make itself available to clients via a well-defined interface. For example, one object could be responsible for querying a database, another could be responsible for performing complex mathematical computations, and another could be responsible for logging messages to a management console.

- A web server that acts as a front door to the server objects. This does not mean that an communications between clients and server objects always communicate via the web server. It only means that clients initially establish connections or sessions to the server objects through the web server.

- One or more clients in the form of web browsers. In theory, this would include any browser in existence. However, for the purposes of this document, only the Netscape Navigator and Internet Explorer browsers will be considered.

There are a lot of different possibilities that share these common elements. This document will not try to cover all of them. Instead, it will cover the basic concepts of the most straightforward solutions. Each technology will have its own advantages and disadvantages. It is very likely that many of these solutions could be adapted or even combined to address individual requirements.

2.4.1 Advantages of Web-Based Distributed Objects

With traditional client/server approaches to software systems, one or more large server programs nm on a large server machine. Client machines connect to the server software and make requests of the server as necessary. With distributed object systems, on the other hand, the large server programs are broken down into a potentially large collection of small server objects. Each object can potentially reside on a different machine on the network Many server objects can even be nm on small desktop computers rather than on large server machines.

The growing popularity of distributed object technologies has been driven by several common problems with traditional client/server approaches. Some of tlıe areas that have been problems for traditional client/server systems include:

- Scalability - This term refers to how easily a particular solution could be extended from a small-scale application to a large-scale application. Many applications work well with just a few functions and users, but fall apart when trying to support hundreds of functions and many thousands of users. Also, many applications will perform well enough when running on a local-area network (LAN), but will not work well when running on a wide-area network (WAN).

- Maintainability - This term refers to lıow costly it is to administer a particular solution, including the costs associated with updating tlıe software and distributing updated versions to client machines.

- Reusability - This term refers to the ability to reuse software component from one system as part of a solution in another system. By reusing software components, software development and testing costs are significantly reduced.

In most cases, these problems were things that could be worked around. As proof of this, there are many traditional client/server systems in use today that are successfully maintained and that have scaled very well. However, most of these successes have occurred only after a rather costly design and implementation of the system has been completed.

By using web-based distributed object technologies, the above problems can be minimized and sometimes eliminated. There is usually no need to design complex infrastructure software that improves scalability and maintainability: the necessary infrastructure already exists, and web-based distributed object technologies take advantage of that fact. Rather than going into a detailed theoretical description of these issues, and how distributed object systems help address them, a sample software system will be described that demonstrates some of the issues.

2.5 Applications Frameworks

Consider an application where thousands of clients need access to a legacy database. In many cases, the only way to access a legacy database is by using proprietary database client software. For the purposes of this example, assume that the client will need to browse and update the database based on user inputs.. Also assume that the best way to access the legacy database is a through a proprietary client library that is provided by the database vendor. Figure 2.6 accessing a database without using distributed technologies

2.5.1 The Traditional Client/Server Approach

Without some form of distributed object technology, each client would need to lıave the database client software installed locally. Figure 2.6 shows the architecture for this approach. Each instance of the client application is linked into the database client library, which is marked as "DB" in Figure 2.6. This database client library is responsible for connectingto the database server, which will usually reside on a different machine.

If this design were used, the system wou1d have some scalability problems. Consider what would happen as the numbers of clients grow. Each client requires a dedicated connection to the database server. Therefore, there is a requirement of one DB connection per client. Many database servers will be limited in the number of database connections that can be supported. With this design, we can only have as many clients.as database connections.

Maintaining software of this design would also be difficult, particularly when it came to updating and distributing the updated software. Consider what would happen if there was a small change to the database schema. It is possible that a change in the client software, which directly accesses the database, would be required. The client application would need to be modified, and the modified version would need to be distributed and installed to every client machine. This may not be a problem when there is just a few client machines, but what if there are hundreds or even thousands of machines? The cost of making updates and getting those updates installed would be rather high.

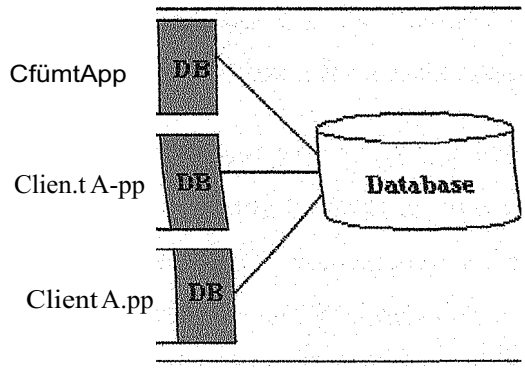


Figure 2.6 Accessing database without distributed technologies

Finally, there is very little of this design that could be reused in other systems. The client application, which we can assume is specific to this system, is coded to directly connect to and query the database, using the database client library. Any new systems to be developed would have to be implemented in a similar way. While it might be possible for the developer to design for reusability and place reusable code into libraries, this takes more time and effort in planning and design. In reality, very few traditional client/server systems have made code reuse a high priority.

2.5.2 The Distributed Object Approach

Using a distributed object design would allow some of the traditional client/server design's problems to be addressed. The basic idea is to move as much code as possible into server objects. These server objects would be responsible for accessing the database. Clients would make requests to the server objects, which would access the database and return results to the requesting client. Figure 2.7 shows this architecture. In Figure 2.7, the label "S" represent'! the server objects. There can be one or more instances of these server objects. It is also possible that a pool of active database comıectioııs is maintained. In Figure 2.7, multiple lines between the database library and the database server represent this pool of connections.

This design offers an improvement in the scalability of the system. There is no longer a direct tie between the number of database connections supported and the number of clients. Tlıe use of a pool of database connections makes it possible to support many more clients than can be supported with the traditional client/server design. Many of the

distributed object environments that are discussed in this document support one or more form of object pooling, which is a critical feature for scalable distributed object systems.

The maintainability of the system is also improved with this design, but still has some difficulties. Because the code to access the database now resides in the server objects, database schema and other code relating to the database access can be changed without requiring updates to the client software. Most changes will be localized to the server objects, bringing down the maintenance cost of the system. However, it is still possible that the client application will require changes. It may be necessary to add new features to the client application, or to modify the behavior of the client's user interface: changes like this will still require updates to be distributed to and installed on every client machine.

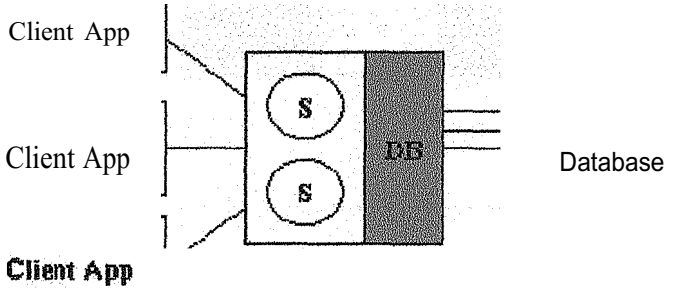


Figure 2.7 Accessing database using distributed object technology

This design makes code reuse a real possibility. For example, a server object could be designed so that it provides a method to clients that will retrieve a customer record based on a customer ID number. This server object might be used by my current system, which uses a "balance" field from the returned customer record as part of a billing system. The same server object could be used with a mailing list system, which might make use of the name and address fields of a customer record instead of the balance.

2.5.3 The Web-based Distributed Object Approach

The previous design works well in most respects, but still has some problems with maintainability. The need to install the client application on every client machine can be

costly, depending on how many clients there are and how often updates will be made. The web-based distributed object approach attempts to address tlıis issue. Figure 2.8 shows the architecture. This architecture is similar to the distributed object architecture, but it uses a web browser on the client and a web server on the server. A client accesses the desired application by navigating to the application's URL using the web browser. The application's source code, or a compiled version of it, is downloaded from the web server and nm within the client browser.

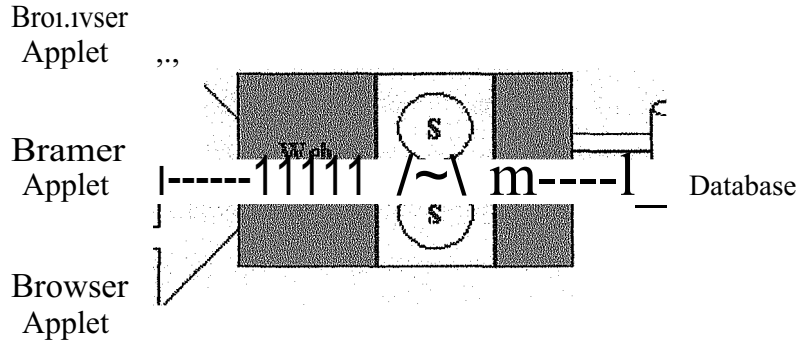


Figure 2.8 Accessing database using web-based technologies

The prime advantage to web-based distributed objects is that ALL code associated with the application is downloaded dynamically from the server. There is usually no need to install any application-specific software on the client machine. This greatly reduces long-term maintenance costs for an application.

2.5.4 Disadvantages.of Web-Based Distributed Objects

As was described in the previous sections, there are advantages to using a web based distributed object solution. There are also disadvantages that need to be considered before committing to one solution or another. The disadvantages when comparing to a traditional client/server solution include:

- Loss of simplicity - The traditional client/server solution is the most direct approach: client applications talk directly to servers such as database servers. The

web-based distributed object approach requires the use of web servers, web browsers, and intermediate server objects. The extra layers of software improve the scalability, maintainability, and reusability of the code, but at the cost of simplicity.

- Dependence on multiple vendors' software - The traditional client/server approach is generally dependent on fewer vendors' software. In the case of our database example, tlıe client application is dependent on the database client and server software, which is usually provided by a single vendor. The web-based distributed object approach will also depend on web browser and server software, as well as any other infrastructure code needed by the chosen technology.

There are also disadvantages when comparing to a simple (non web-based) distributed object solution. The primary means of dynamically downloading code into a browser that will be described in this document is the Java applet Java applets are downloaded in the form of Java "byte code", a standard format defined by the Java specifications. Using Java applets introduce the following disadvantages:

- Security - Running the client application from within a browser has security restrictions. These restrictions were created to make sure that clients didn't accidentally download and run programs tlıat did things like erase the local hard drive or steal private passwords. Due to security restrictions, most dynamically downloaded applet code can't do all of the things that a standalone application can do. This limitation can be addressed using "trusted" code, which allows downloaded code to be trusted by the user to do certain things.

- Performance during initialization - Because application code must be downloaded from the server when the application's URL is requested by a web browser, initialization time is much slower than it *is* with a standalone application. With a standalone application, the application's code is already installed locally, so there is no need to download the code.

- Run-time performance -Java byte code was designed to work in any browser on any platform - It must be interpreted by each machine's Java interpreter (also known as the Java Virtual Machine or JVM), and converted to native machine instructions.

This conversion process could potentially decrease performance of applications. Standalone applications are usually in the form of compiled code, which requires no interpreter.

Each of these disadvantages should be considered before committing to a particular design. In most cases, these disadvantages are acceptable given the great number of advantages. However, there will definitely be some systems where web-based distributed objects are not the best solution.

2.6 General Approach

Each technology and architecture approach should evaluated based on the following key areas and requirements:

- Portability - This area applies to both portability between different server platforms (Windows, UNIX, etc.) and portability between different client browsers.
- Scalability - The ideal technology will allow a small-scale solution to be implemented at a very low cost, but allow that solution to be built into a large-scale solution without an exponential rise in cost.
- Technology Lock-in - Due to tlıe uncertain future of Java and/or internet technologies, it is desirable to avoid a complete lock-in to one technology. Also, this area includes descriptions of lock-in to individual vendors. The ideal solution will allow a transition to alternate technologies and/or vendors if and wlıen it becomes necessary to do so.
- Performance - This are applies to both download (initialization) performance, as well as transactional performance.
- Ease of Development - The ideal solution will allow projects to be developed without requiring the work of one or more Java experts. While familiarity with Java is considered a prerequisite in all cases, it is desirable to require as little Java and/or distributed object expertise as possible.

- Cost - This area does not necessarily apply to a fixed dollar cost. It is instead intended to reflect the amount of software that needs to be purchased, installed, and configured. The ideal solution will work with little or no need for additional software.

For each of the above areas, there will be many cases where variants or workarounds can be used to improve the results of the evaluation.

2.7 Technology Example

Microsoft believes that Windows DNA is the premier application architecture for creating distributed computing solutions because it effectively addresses the key concerns of enterprise customers. Windows DNA offers what IT professionals want and delivers what application developers need.

Windows DNA is a proven platform that uniquely builds on your company's existing investments in Windows and PC platform technologies, applications, tools, and hardware. With Windows DNA, you can build distributed, multi-tier computing solutions that take full advantage of your existing client/server expertise, while enabling you to embrace emerging technologies such as the Internet. Windows DNA:

- Integrates and extends existing systems.

Windows DNA is designed to provide a high level of interoperability with existing enterprise applications and legacy systems, making it easy to protect and extend your current investments. With Windows DNA, you can build interoperability into all tiers of your applications, which makes it easy to add functionality to existing systems. Because Windows DNA is based on open protocols, industry standards, and published interfaces. solutions from other vendors integrate easily into the environment. This helps ensure interoperability with other mission-critical business applications such as corporate databases and enterprise resource planning systems. An open approach also facilitates compatibility with existing computing systems, which means that companies can continue to take advantage of legacy systems, as opposed to replacing them.

Microsoft is also. exporting key elements of Windows DNA to non-Microsoft platforms to provide a framework for interoperability and integration throughout an organization's computing environment

- Facilitates rapid application development (RAD)

Windows DNA provides the most comprehensive and integrated platform for building distributed applications because it relies on a rich set of services supplied by the Windows platform. These services are key infrastructure technologies that are required for any scalable, distributed application and include Web services, security, directory, transaction processing, messaging, and systems management. Other platforms provide these critical services as piecemeal, non-integrated offerings, which forces developers to act as system integrators. By providing a stable base of common, integrated services, Windows DNA relieves developers of the burden of having to either create their own infrastructure or integrate disparate pieces, so they can focus on delivering business solutions. The bottom line is that developers save time, reduce costs, and get applications to market more quickly.

Unlike traditional software development, which required each application to be built from scratch, the Component Object Model (COM) enables developers to create complex applications using a series of small software objects (COM components). A component could be a tax calculation engine or the business rules for a price list. The COM programming model greatly speeds up the development process because several development teams can work on different parts of the application simultaneously.

- Provides a high degree of flexibility

The use of COM components combined with the multi-tier programming model makes Windows DNA flexible enough to effectively address the ever-changing business requirements of today's competitive companies. Using COM means that code is written in small, manageable chunks, so it's much easier to make changes to an application or accommodate new functions. Since each component's functionality is isolated, less testing is necessary, and it's easier to update components on the server. COM also enables developers to reuse components from one project to the next. They can easily swap out or update a particular component without affecting other parts of the application.

Dividing applications into distinct tiers makes it easier for developers to satisfy the needs of a wide range of users. Employees who have mission-critical needs can use powerful Windows client front ends that provide the highest degree of performance and robust feature sets. And customers who want simple access from a variety of platforms can use browser-based front ends. Since the user interface is confined to the presentation tier, there's no need to change the business mies or data access methods.

2.8HTML

To maintain a broad reach to a wide range of client environments while achieving the greatest compatibility with all browsers, application developers will generally use standard HTML. Microsoft tools and application services support the current generation of standard HTML.

2.8.1 Scripting and DHTML

Many times corporations standardize on a single browser. Application developers who want to provide more functionality in their applications, than they can achieve with standard HTML, write code to determine the browser being used. Applications can be written to take advantage of the technologies inherent in the browser in order to gain maximum functionality and richness. With technologies like scripting and DHTML, application developers can create actions with functional Web-based interfaces for data entry or reporting without using custom controls or applets.

DHTML is based on the World Wide Web Consortium (W3C)-staııdard Document Object Model, which makes all Web page elements programmable objects. DHTML can be thought of as "programmable" HTML. Contents of the HTML document, including style and positioning informatioıı, can be modified dynamically by script code embedded in the page. In this way, scripts can change the style, content, and structure of a Web page without having to refresh it from the Web server.

2.8.2 Components

There are times when DIITML plus scripting isn't enough. Segments of applications may need to use the operating system and underlying machine on which they're hosted

while still maintaining an active connection to the Internet for data or additional services. In those instances, application developers can take advantage of the components and Internet services provided by Windows to build more robust applications.

2.8.3 Win32 API

Applications written using tlıe Win32 API offer the most functionality, but their reach is limited to the application platforms that support this API. Through the use of cooperating components, developers today have access to Internet technologies in the Windows application platform from within a Win32-based application. Some common examples are Microsoft Office and the Microsoft® Visual Studio® development system. These applications support unified browsing by embedding hyperlinks from within the application. They also host the browser for display of documentation written in DHTML and provide the capability to download updates to the products over the Internet seamlessly.

CHAPI'ER3

CLIENT/SERVER APPLICATION THE SERVER

3.1 Introduction

The server is a multi user computer. There is no special hardware requirement that turns a computer into a server. The hardware platform should be selected based on application demands and economics. Servers for client/server applications work best when they are configured with an operating system that supports shared memory, application isolation, and preemptive multitasking. An operating system with preemptive multitasking enables a higher priority task to preempt or take control of the processor from a currently executing, lower priority task

The server provides and controls shared access to server resources. Applications on a server must be isolated from each other so that an error in one cannot damage another. Preemptive multitasking ensures that no single task can take over all the resources of the server and prevent other tasks from providing service. There must be a means of defining the relative priority of the tasks on the server. These requirements are specific to the client/server implementation and not to the file server implementation. Because file servers execute only the single task of file service, they can operate in a more limited operating environment without the need for application isolation and preemptive multitasking.

Many organizations download data from legacy enterprise servers for local manipulation at workstations. In the client/server model, the definition of server will continue to include these fiinctions, perhaps still implemented on the same or similar plat forms. Moreover, the advent of open systems based servers is facilitating the placement of services on many different platforms.

Client/server computing is a phenomenon that developed from the ground up. Remote workgroups have needed to share expensive resources and have connected their desktop workstations into local area networks.

3.2 The Role of the Server

Servers provide application, file, database, print, fax, image, communications, security, systems, and network management services. These are each described in some detail in the following sections.

It is important to understand that a server is an architectural concept, not a physical implementation description. Client and server functions can be provided by the same physical device. With the movement toward peer computing, every device will potentially operate as a client and server in response to requests for service.

Application servers provide business functionality to support the operation of the client workstation. In the client/server model these services can be provided for an entire or. partial business function invoked through an Inter Process Communication (IPC) request for service. A collection of application servers may work iii concert to provide an entire business function. For example, in a pay roll system the employee information may be managed by one application server, earnings calculated by another application server, and deductions calculated by a third application server. These servers may run different operating systems on various hardware platforms and may use different database servers. The client application invokes these services without consideration of the technology or geographic location of the various servers.

Fax servers provide support similar to that provided by print servers. In addition, fax servers queue up outgoing faxes for later distribution when communications charges are lower, the fax server must be capable of dynamically compressing and decompressing documents for distribution, printing, and display. This operation is usually done through the addition of a fax card to the server.

Communications servers provide support for wide area network (WAN) communications. This support typically includes support for a subset of IBM System Network Architecture (SNA), asynchronous protocols, and LAN-to.:LAN NetBIOS communication protocols.

Security at the server restricts access to software and data accessed from the server. Communications access is controlled from the communications server. In most implementations, the use of a user login ID is the primary means of security

J.3 Server Functionality in Detail

The discussion in the following sections more specifically describes the functions provided by the server in a NOS environment.

3.3.1 Request Processing

Requests are issued by a client to the NOS services software resident on the client machine. These services format the request into an appropriate RPC and issue the request to the application layer of the client protocol stack. This request is received by the application layer of the protocol stack on the server.

3.3.2 File Services

File services handle access to the virtual directories and files located on the client workstation and to tlıe server's permanent storage. These services are provided through the redirection software implemented as part of the client workstation operating environment. The file services provide this support at the remote server processor. In the typical implementation, software, shared data, databases, and backups are stored on disk, tape, and optical storage devices that are managed by the file server.

To minimize the effort and effect of installation and maintenance of software, software should be loaded from the server for execution on the client. New versions can be updated on the server and made immediately available to all users. In addition, installation in a central location reduces the effort required for each workstation user to handle the installation process. Because each client workstation user uses the same installation of the software, optional parameters are consistent, and remote help desk operators are aware of them. This simplifies the analysis that must occur to provide support. Sharing information, such as word processing documents, is easier when everyone is at the same release level and uses the same default setup within the software. Central productivity services such as style sheets and macros can be set up for general use.

Backups of the server can be scheduled and monitored by a trained support person. Backups of client workstations can be scheduled from the server, and data can be stored at the server to facilitate recovery. Tape or optical backup units are typically used for backup; these devices can readily provide support for many users. Placing the server and its backups in a secure location helps prevent theft or accidental destruction of backups. A central location is readily monitored by a support person who ensures that the backup functions are completed. With more organizations looking at multimedia and image technology, large optical storage devices are most appropriately implemented as shared servers.

3.3.3 Fax/Print/Image Services

High-quality printers, workstation-generated faxes, and plotters are natural candidates for support from a shared server. The server can accept input from many clients, queue it according to the priority of the request and handle it when the device is available. Many organizations realize substantial savings by enabling users to generate fax output from their workstations and queue it at a fax server for transmission when the communication costs are lower. Incoming faxes can be queued at the server and transmitted to the appropriate client either on receipt or on request. In concert with workflow management techniques, images can be captured and distributed to the appropriate client workstation from the image server. In the client/server model, work queues are maintained at the server by a supervisor in concert with default algorithms that determine how to distribute the queued work.

Incoming paper mail can be converted to image form in the mail room and sent to the appropriate client through the LAN rather than through interoffice mail. Centralized capture and distribution enable images to be centrally indexed. This index can be maintained by the database services for all authorized users to query. In this way, images are captured once and are available for distribution immediately to all authorized users. Well-defined standards for electronic document management will allow this technology to become fully integrated into the desktop work environment. There are dramatic opportunities for cost savings and improvements in efficiency if this technology is properly implemented and used. Chapter 10 discusses in more detail the issues of electronic document management.

3.3.4 Database Services

Early database servers were actually file servers with a different interface. Products, Clipper, FoxPro, and Paradox execute the database engine primarily on the client machine and use the file services provided by the file server for record access and free space management. These are new and more powerful implementations of the original flat-file models with extracted indexes for direct record access. Currency control is managed by the application program, which issues lock requests and lock checks, and by the database server, which creates a lock table that is interrogated whenever a record access lock check is generated. The lack of server execution logic prevents these products from providing automatic partial update back out and recovery after an application, system, or hardware failure.

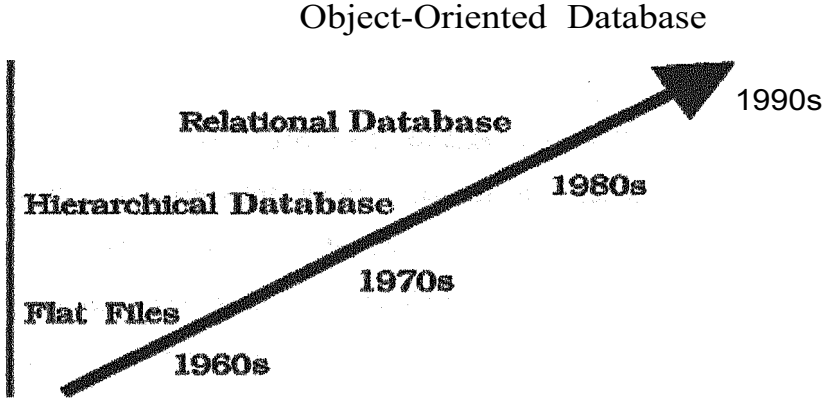


Figure 3.1 Bata base trends

Client/server database engines such as Sybase, IBM's Database Manager, Ingress, Oracle, and Informix provide support at the server to execute SQL requests issued from the client workstation. The :file services are still used for space allocation and basic directory services, but all other services are provided directly by the database server. Relational database management systems are the current technology for data management. Figure before charts the evolution of database technology from the first computers in the late

1950s to the object-oriented database technologies that are becoming prevalent in the mid-1990s.

- Flat Files: Sorting Physical Records: Database technology has evolved from the early 1960s' flat-file view when data was provided through punch cards or disk files simulating punch cards. These original implementations physically stored data columns and records acôordingtc the user view.

- Hierarchical Databases: Stôrage of Related Record Types: The second generation of database tecluıology, the hierarchical database, could store related record types physically or logically next to each other. In.the füerarchicalmodel implementation, when a user accesses a "physicalrecord type, other application-rela.teddata is usually stored physically close and will be moved from disk to DRAM all together.Internally stored pointers ate used to navigate from one record to the next if there.is.insufficientspace close by at·data creation time to insert the related data.

- Relational Databases: Extracted Indexes and SQL: As hardware technology evolves, it is important for the data management capabilities to evolve to use the new capabilities summarizes the .cuırentessential -characteristics ofthe database world. The relational database is the de facto standard today; therefore, investment by vendors wilI be in products that target and support fully compliant SQL databases.

- Object-Oriented a Bright Future: With the increasing maturity and popularity for development, there has been a significant increase in maturity and acceptance of object-oriented database management systems (OODBMS). Object-oriented database management systems provide support for complex data structures: such as compound documents, CASE entity relationship models, financial models, and CAD/CAM drawings. OODBMS proponents claim that relational database management systems (RDBMS) can handle only simple data structures (such as tables) and simple transaction-processingapplications that only need to create views combining a small number of tables. OODBMS proponents argue that there is a

large class of problems that need to be and will be more simply implemented if more complex data structures· can be viewed directly. RDBMS vendors agree with the need to support these data structures but argue that the issue is one of implementation, not architecture.

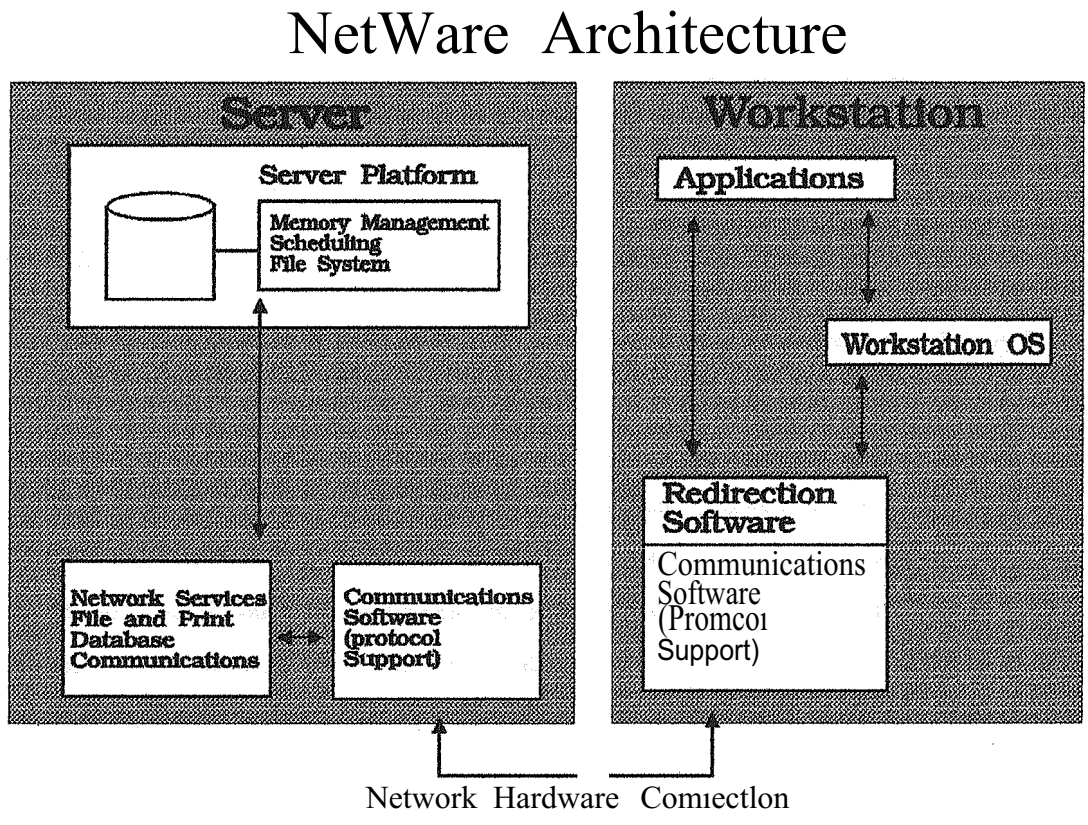


Figure 3.2 Object oriented database

The application characteristics that lead to an OODBMS choice are shown in Figure 3.2 OODBMS will become production capable for these types of applications with the introduction of 16Mbps D-RAM and the creation of persistent (permanent) databases in D-RAM. Only the logging functions will use real I/0. Periodically, D-RAM databases will be backed up to real magnetic or optical disk storage. During 1993, a significant number of production OODBMS applications were implemented. With the confidence and experience gained from these applications, the momentum is building, and 1994 and 1995 will see a significant increase in the use of OODBMS for business critical applications. OODBMS

have reached a maturity level coincident with the demand for multimedia enabled applications. The complexities of dealing with multimedia; demands the features of OODBMS for effective storage and manipulation.

3.3.5 Cemmunieatiens Services

Client/server applications require LAN and WAN communication services. Basic LAN services are integral to the NOS. WAN services are provided by various communications server products. Chapter 5 provides a complete discussion of connectivity issues in the client/server model.

3.3.6 Security Services

Client/server applications require similar security services to those provided by host environments. Every user should be required to log in with a user ID and password. If passwords might become visible to unauthorized users, the security server should insist that passwords be changed regularly. The enterprise on the desk implies that a single logon ID and logon sequence is used to gain the authority once to access all information and process for the user has a need and right of access. Because data may be stored in a less physically secure area, the option should exist to store data *in* an encrypted form. A combination of the user ID and password should be required to decrypt the data.

3.4 The Network Operating System

The network operating system (NOS) provides the services not available from the client OS.

1. NovellNetWare:

NetWare is a family of LAN products with support for IBM PC-compatible and Apple Macintosh clients.

NetWare has benefited from its high performance and low resource requirements as much.as it has from its relative ease of use. This performance has been provided through the use of a proprietary operating system and network protocols. Even thoiigh this has given Novell an advantage in performance, it has caused difficulties in the implementation

of application and database servers in the Novell LAN. Standard applications cannot run on the server processor, because NetWare does not provide compatible APis. Instead, NetWare provides a high performance capability called a NetWare Loadable Module (NLM) that enables database servers such as Sybase and Oracle, and communications servers such as Gateway Communications provides, to be linked into the NetWare NOS.

2. LAN Manager:

LAN Server, are the standard products for use in client/server implementations using OS/2 as the server operating system. LAN Manager/X is the standard product for client/server implementations using UNIX System V as the server operating system,

LAN Manager and Advanced Server provide client support for DOS, Windows, Windows NT, OS/2, and Mac System 7. Server support extends to NetWare, Apple'Ialk, UNIX, Windows NT, and OS/2. Client workstations can access data from both NetWare and LAN Manager servers at the same time. LAN Manager supports Net BIOS and Named Pipes LAN communications between clients and OS/2 servers. Redirection services are provided to map files and printers from remote workstations for client use.

3. IBM LAN Server:

IBM has entered into an agreement to resell and integrate the Novell NetWare product into environments where both IBM LAN Server and Novell NetWare are required. NetWare provides more functional, easier-to-use, and higher-performance file and print services. In environments where these are the only LAN functions, NetWare is preferable to LAN Manager derivative. The capability to interconnect to the SNA world makes the IBM product LAN Server attractive to organizations that prefer to run both products. Most large organizations have department workgroups that require only the services that Novell provides well but may use LAN Server for client/server applications using SNA services such as APPN.

4. PC Network File Services (NFS):

NFS is the standard file system support for UNIX. PC NFS is available from Sun Select and FTP to provide file services support :from a UNIX server to Windows, OS/2,

Mac, and UNIX clients. NFS lets a client mount an NFS host's filing system (or a part of it) as an extension of its own resources. NFS resource-sharing mechanisms encompass inter host printing. The transactions among NFS systems traditionally ride across TCP/JP and Ethernet, but NFS works with any network that supports 802.3 frames.

Sun Select includes instructions for adding PC-NFS to an existing LAN Manager or Windows for Workgroups network using Network Driver Interface Specification (NDIS) drivers.

3.5 The Available Platforms

Client/server computing requires that LAN and WAN topologies be in place to provide the necessary internet working for shared applications and data. The results of special interest are the projection that most workstations will be within LANs by 1996, but only 14 percent will be involved in an enterprise LAN by that date

1. Workstations in LAN Configuration

This model is the most basic implementation providing the standard LAN services for files and printer sharing.

2. LAN-to-LAN/WAN Configuration

Routers and communication servers will be used to provide communication services between LANs and into the WAN. In the client/server model, these connections will be provided transparently by the SDE tools. There are significant performance implications if the traffic volumes are large. IBM's LU6.2 implementation in APPC and TCP/JP provides the best support for high-volume, LAN-to-LAN/WAN communications, DEC's implementation of DEC net always has provided excellent LAN-to-WAN connectivity.

3. LAN-to-Host Configuration

The lack of real estate on the desktop encouraged most organizations to move to a single device using terminal emulation from the workstation to access existing mainframe applications. It will take considerable time and effort before all existing host-based applications in an organization are replaced by client/server applications. In the long term,

the host will continue to be the location of choice for enterprise database storage and for the provision of security and network management services.

Their roles will change, but they will be around as part of the enterprise infrastructure for many more years. Only organizations who create an enterprise architecture strategy and transformational plans will accomplish the migration to client/server in less than a few years. Without a well-architected strategy, gradual evolution will produce failure.

4. Enterprise-Wide

Information that is of value or interest to the entire business must be managed by a central data administration function and appear to be stored on each user's desk. These applications are traditionally implemented as Online Transaction Processing (OLTP) to the mainframe or minicomputer. With the client/server model, it is feasible to use database technology to replicate or migrate data to distributed servers. Wherever data resides or is used, the location must be transparent to the user and the developer. Data should be stored where itbest meets the business need.

3.6 The Server Operating System

Servers provide the platform for application, database, and communication services. There are six operating system platforms that have the greatest positional and/or are prevalent today: NetWare, OS/2, Windows NT, MVS, VMS, and UNIX.

3.6.1 NetWare

NetWare is used by many organizations, large and small, for the provision of file, printer, and network services. NetWare is a self-contained operating system. It does not require a separate OS (as do Windows NT, OS/2, and UNIX) to nm. Novell is taking steps to allow NetWare to run on servers with UNIX. Novell purchased USL and will develop shrink-wrappedproducts to run under both Netware and UNIX System V, Release 4.2. The products will enable UNIX to simultaneouslyaccess information from both a NetWare and a UNIX server.

3.6.2 OS/2

OS/2 is the server platform for Intel products provided by IBM in the System Application Architecture (SAA) model. OS/2 provides the storage protection and preemptive multitasking services needed for the server platform. Several database and many application products lıave been ported to OS/2. The only network operating systems directly supported with OS/2 are LAN Manager and LAN Server. Novell supports the use of OS/2 servers running on separate processors from the NetWare server. The combination of Novell with an OS/2 database and application servers can provide the necessary environment for a production-quality client/server implementation. Appendix A describes such an implementation.

3.6.3 Windows NT

With the release of Windows NT (New Technology) in September of 1993, Microsoft staked its unique position with a server operating system. Microsoft's previous development of OS/2 with IBM did not create tlıe single standard UNIX alternative that was hoped for. NT provides the preemptive multitasking services required for a functional server. It provides excellent support for Windows clients and incorporates the necessary storage protection services required for a reliable server operating system. Its implementation of C2 level security goes well beyond that provided by OS/2 and most UNIX implementations. It will take most of 1994 to get the applications necessary to provide an industrial strength platform for business critical applications. With Microsoft's prestige and marketing muscle, NT will be installed by many organizations as their server of choice.

3.6.4 MVS

IBM provides MVS as a platform for large applications. Many of the existing application services tlıat organizations have purchased operate on System 370.compatible hardware running MVS. The standard networking environment for many large organizations SNA is a component of MVS. IBM prefers to label proprietary systems today under the umbrella of SAA. The objective of SAA is to provide all services on all IBM platforms in a compatible way the IBM version of the single-system image.

MVS provides a powerful database server using DB2 and LU6.2. With broad industry support for LU6.2, requests that include DB2 databases as part of their view can be issued from a client/server application. Products such as Sybase provide high-performance static SQL support, making this implementation viable for high-performance production applications.

3.6.5 OPENVMS

Digital Equipment Corporation provides OPENVMS as its server platform of choice. VMS has a long history in the distributed computing arena and includes many of the features necessary to act as a server in the client/server model. DEC was slow to realize the importance of this technology, and only recently did the company enter the arena as a serious vendor. NetWare supports the use of OPENVMS servers for file services. DEC provides its owıı server interface using a LAN Manager derivative product called Path works.

Path works runs native on the VAX. and RISC Alpha RXP. This is a particularly attractive configuration because it provides access on the same processor to the application, database, and fi.le services provided by a combination of OPENVMS, NetWare, and LAN Manager. Digital and Microsoft have announced joint agreements to work together to provide a smooth integration of Windows, Windows NT, Path works, and OPENVMS. This will greatly facilitate the migration by OPENVMS customers to the client/server model.

3.6.6 UNIX

UNIX is a primary player as a server system in the client/server model. Certainly, the history of UNIX in the distributed computing arena and its open interfaces provide an excellent opportunity for it to be a server of choice.

To understand what makes it an open operating system, look at the system's components. UNIX was conceived in the early 1970s by AT&T employees as an operating environment to provide services to software developers who were discouraged by the incompatibility of new computers and the lack of development tools for application

development. The original intention of the UNIX architecture was to defme a standard set of services to be provided by the UNIX kernel. These services are used by a shell that provides the command-line interface. Functionality is enhanced throughı the provision of a library of programs. Applications are built up from the program library and custom code. Thepower and appeal of UNIX lie in the common definition of the kernel and shell and in the large amount Of software tlıat lıas been built and is· available. Applications built around these standards catibe pörled to many diffefeıt hardware platforms.

To oveicôfue this, and in aıı attempt tô achieve an inıplementatfön ôfUNIX better suited to the needs of developers, the University of California at Berkeley and other institutions developed better varieties.of UNIX. As a result, the original objective of a portable platform was cofuprôıfüsed. The new prôdu.cts Were surely better, but tlıey were not compatible with each other or the original implementation. Through the mid- 1980s, many versions of IJNIX that had increasing functionality were released. IBM, of course, entered the fray in 1986 with its own UNIX derivative, AIX. Finally, in 1989, an agreement was reached on the basic UNIX kernel, shell functions, and APis.

In addition to the complexity this entails, a more serious problem exists with software versioning. Software vendors update their software on a regular basis, adding functionality and fixing problems. Because the UNIX kernel is implemented on each platform and the software must be compiled for the target platform, there are differences in the low-level operation of each platform.

This requires that software vendors port their applications to each platform they support. This porting function can take from several days to several months .. In fact, if the platform is no longer popular, the port may never occur. Thus, users who acquire a UNIX processor may find that their software vendor is no longer committed to upgrading their software for this platform.

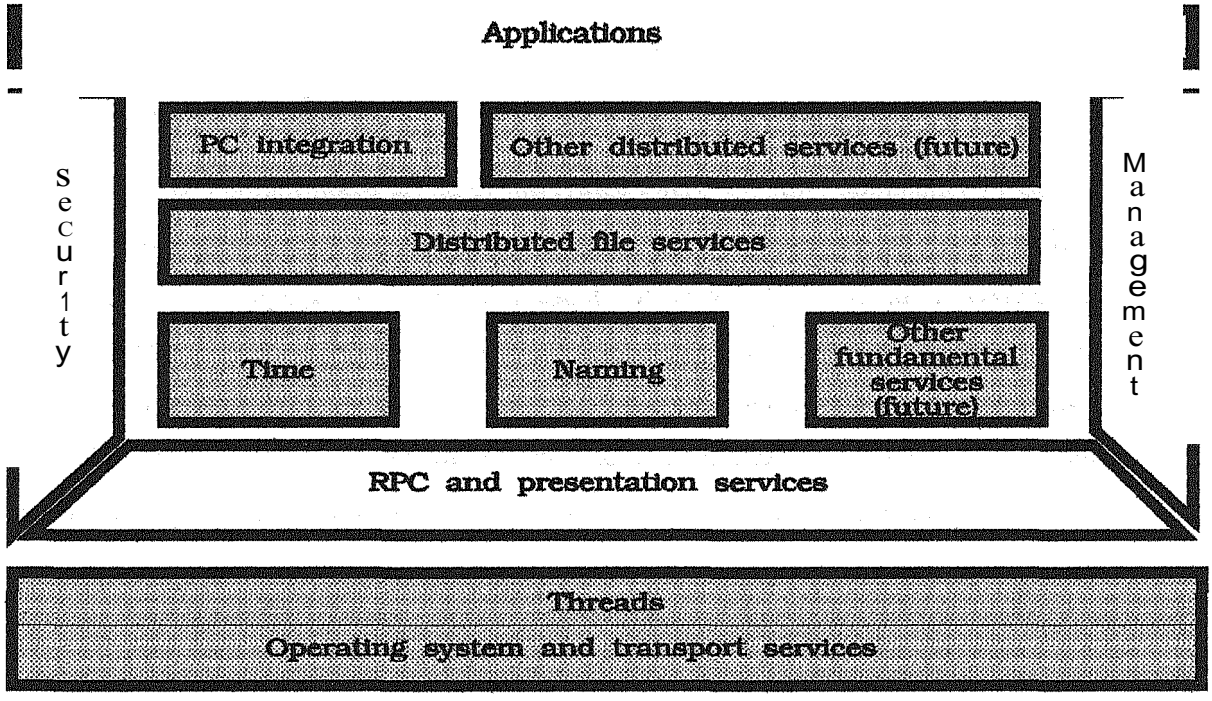


Figure 4.3 UNIX history

UNIX is particularly desirable as a server platform for client/server computing, because of the large range of platforni sizes available and the huge base of application and development software available. Universities are contributing to the UNIX momentum by graduating students who see only UNIX during their student years. Government agencies are insisting on UNIX as the platform for all government projects. The combination of these pressures and technology changes should ensure that UNIX compatibility will be mandatory for server platforms in the last half of this decade.

3.6.7 Distributed Computing Environment (DCE)

Spin this fantasy around in your mind. All the major hardware and software vendors get together and agree to install a black box in their systems that will, in effect, wipe away their technological barriers. This black box will connect a variety of small operating systems, dissimilar lıardware platforms, incompatible communications protocols, all sorts of applications and database systems, and even unlike security systems. And the black box

will do all this transparently, not only for end users but also for systems managers and applications developers.2 OSF proposes the distributed computing environment (DCB) as this black box. DCB is the most important architecture defined for the client/server model. It provides the bridge between existing investments in applications and new applications based on current technology.

DCB is a prepackaged group of integrated interoperability applications that connect diverse hardware and software systems, applications, and databases. To provide these services, DCB components must be present on every platform in a system. These components become active whenever a local application requests data, services, or processes from somewhere. The OSF says that DCB will make a network of systems from multiple vendors appear as a single stand-alone computer to applications developers, systems administrators, and end users. Thus, the single-system image is attained.

The various elements of DCB are as follows:

- Remote Procedure Call (RPC) and Presentation Services: Interface Definition Languages across a network in a transparent manner that helps to mask the network's complexity.

- Naming: User-oriented names, specifying computers, files, and people should be easily accessible in a distributed environment

- Security: Distributed applications and services must identify users, control access to resources, and guard the integrity of all applications

- Threads: This terminology represents a method of supporting parallel execution by managing multiple threads of control within a process operating in a distributed environment.

- Time Service: A time service synchronizes all system clocks of a distributed environment so that executing applications can depend on equivalent clocking among processes.

- Distributed File Services: By extending the local file system throughout the network, users gain full access to files on remote configurations

- PC Integration: Integration enables PCs using MS-DOS, Windows NT, and OS/2 to access file and print services outside the MS-DOS environment. DCE uses Microsoft's LAN Manager/X.

- Management: Although partly addressed by the previous components, management is so complex in a distributed, heterogeneous configuration that OSF has defined a new architecture: distributed management environment (DME). DME provides a common framework for the management of stand-alone and distributed systems. This framework provides consistent tools and techniques for managing different types of systems and enables vendors to build system management applications that work on a variety of platforms

- Communications: DCE is committed to support the OSI protocol stack.

3.7 System Application Architecture (SAA)

SAA is IBM's distributed environment. SAA was defined by IBM in 1986 as architecture to integrate all IBM computers and operating systems, including MVS, VM/CMS, OS/400, and OS/2-EE. SAA defines standards for a common user access (CUA) method, common programming interfaces (CPI), and a common coımnunication link (APPC).

To support the development of SAA-compliant applications, IBM described SAA frameworks (that somewhat resemble APis). The first SAA framework is AD/Cycle, the SAA strategy for CASE application development. AD/Cycle is designed to use third-party tools within the IBM SAA hardware and mainframe Repository Manager/MYS data storage facility. Several vendors have been selected by IBM as AD/Cycle partners, namely.

Unfortunately, the most important component, the Repository Manager, has not yet reached production quality in its MVS implementation and as yet there are no plans for a

client/server implementation. Many original IBM customers involved in evaluating the Repository Manager have returned the product in frustration. Recently, there has been much discussion about the need for a production-quality, object-oriented database management system (OODBMS) to support the entity relationship (ER) model underlying the repository. Only this, say some sources, will make implementation and performance practical. A further failing in the SAA strategy is the lack of open systems support. Although certain standards, such as Motif, SQL, and LU6.2, are identified as part of SAA; the lack of support for AIX has prevented many organizations from adopting SAA. IBM has published all the SAA standards and has licensed various protocols.

The failure of SAA is attributable to the complexity of IBM's heterogeneous product lines and the desire of many organizations to move away from proprietary to open systems solutions. This recognition led IBM to announce its new Open Enterprise plan to replace the old System Application Architecture (SAA)plan with an open network strategy. System View is a key IBM network product linking OS/2, UNIX, and AS/400 operating systen;ıs. Traditional Systems Network Architecture (SNA) networking will be replaced by new technologies, such as Advanced Peer-to-Peer Communications (APPC) and Advanced Peer-to-Peer Networking (APPN).

CHAPTER4

CLIENT/SERVER APPLICATION THE CLIENT

4.1 Introduction

The client in the client/server model is the desktop workstation. Any workstation that is used by a single user is a client. The same workstation, when shared simultaneously by multiple users, is a server. Aıı Apple Macintosh SE, an IBM PS/2 Model 30, an ALR 386/220, a Compaq System Pro., an NCD X-Terminal, a DEC station 5000-all are used somewhere as a client workstation. There is no specific technological characteristic of a client.

During the past 10 years, workstation performance improved dramatically. For the same cost, workstation CPU performance increased by 50 times, main memory has increased by 25 times, and permanent disk storage has increased by 30 times. This growth in power allows much more sophisticated applications to be run from the desktop.

Communications and network speeds have improved equally in the last 10 years. In 1984, the performance and reliability of remote file, database, and print services were inadequate to support business applications. With the advent of high-speed local and wide area networks (LANs and WANs), networking protocols, digital switches, and fiber-optic cabling, both performance and reliability improved substantially. It is now practical to use these remote services as part of a critical business application.

The client workstation may use the DOS, Windows, Windows NT, OS/2, Mac OS (also referred to as System 7), or UNIX operating system. The client workstation frequently provides personal productivity functions, such as word processing, which use only the hardware and software resident right on the workstation. When the client workstation is connected to a LAN, it has access to the services provided by the network operating system (NOS) in addition to those provided by the client workstation. The workstation may load software and save word-processed documents from a server and therefore use the file server functions provided through the NOS. It also can print to a remote printer through the NOS. The client workstation may be used as a terminal to access applications resident on a host

minicomputer or mainframe processor. This enables the single workstation to replace the terminal, as well as provide client workstation functionality.

In a client/server application, functions are provided by a combination of resources using both the client workstation processor and the server processor. For example, a database server provides data in response to an SQL request issued by the client application. Local processing by the client might calculate the invoice amount and format the response to the workstation screen.

4.2 The Role of the Client

In the cljent/server model, the client is primarily a consumer of services provided by one or more server processors. The model provides a clear separation of functions based on the idea of servers acting as service providers responding to requests from clients. It is important to understand that a workstation can operate as a client in some instances while acting as a server in other instances. For example, in a LAN Manager environment, a workstation might act as a client for one user while simultaneously acting as a print server for many users. This chapter discusses the client functions.

The client almost always provides presentation services. User input and final output, if any, are presented at the client workstation. Current technology provides cost effective support for a graphical user interface (GUI). This book recommends that all new applications, with direct interaction by a human, be developed using a GUI. The windowing environment enables the client user to be involved in several simultaneous sessions. Such functions as word processing, spreadsheet, e-mail, and presentation graphics-in addition to the custom applications built by the organization=can be active simultaneously. Windows 3.x and Mac System 7 do not support true multitasking; thus, only one task at a time can safely be engaged in a communications session. Windows NT, OS/2, and UNlX are preemptive multitasking operating.systems and thus will support any number of active communications sessions.

Facilities such as Dynamic Data Exchange (DDE), Object Level Embedding (OLE), and Communicating Object Request Broker Architecture (CORBA), which are discussed later in this chapter, provide support for cut-and-paste operations between word processors,

databases, spreadsheets, and graphics in a windowing environment. Beyond this, a selectable set of tasks may be performed at the client. In fact, the client workstation can be both client and server when all information and logic pertinent to a request is resident and operates within the client workstation.

Software to support specific ftmctions-for example, field edits, con-text-sensitive help, navigation, training, personal data storage, andmanipulation=-frequently executes on the client workstation. All these functions use the GUI and windowing fünctionality. Additional business logic for calculations, selection, and analysis can reside on the client workstation.

A client workstation uses a local operating system to host both basic services and the network operating system interfaces. This operating system may be the same or different :from that of the server. Most personal computer users today use DOS or Windows 3.x as their client operating system, because current uses are primarily personal productivity applications-not ones requiring a client/server.

The Common Open Software Environment (COSE) group of UNIX kernel vendors has agreed on a common set of API's for most UNIX services. This allows application developers to build one application for all platforms. This will serve to expand the number of applications that will run across the various UNIX platforms. In tum, this will increase the use of UNIX on the desktop and subsequently reduce the per-seat cost.

With tlıe uncertainty surrounding tlıe operating system alternatives, it is important that all development be done with an SDE that isolates the operating system from the application. Then, if operating system changes are warranted the applications should be able to port without any impact beyond recompilation.

4.3Client Services

The ideal client/server platform operates in an open systems environment using a requester-server discipline that is based on well-defined standards. This enables multiple hardware and software platforms to interact. When the standard requester-server discipline is adhered to, servers may grow and change their operating system and hardware platforms

without changing the client applications, and run the same application issuing the same requests for service as long as the standard requester-server discipline is adhered to. Traditional host applications that use the client for presentation services operate only by sending and receiving a character data stream to and from a server. All application logic resides on the server. This is the manner in which many organizations use workstation technology today. The expensive mainframe CPU is being used to handle functions that are much more economically provided by the workstation.

First-generation client/server applications using software such as Easel enable the input and output data streams to be reformatted at the client without changes to the host applications. They use an API that defines the data stream format. Easel uses the IBM-defined Extended High Level Language Application Program Interface (EHLLAPI). GUI front ends may add additional functionality, such as the capability to select items for input from a list, selectively use color, or merge other data into the presentation without changing the host application.

In the client/server implementation of this system, the workstation user deals only with a GUI. The workstation plots this address onto a map that in turn displays the location of the fire. In addition, the locations of all fire stations and vehicles are plotted on the map. The dispatch operator can see at a glance the entire status of fire support close to the fire. Previous implementations of this application displayed lists of optional fire vehicles. From this list the operator keyed in a selected vehicle. The GUI front end, however, enables the vehicles to be shown in a window and selected by using a mouse pointer. This not only reduces the cost of execution but can significantly reduce errors, increase productivity, and reduce stress experienced by the dispatch operator.

The functionality of the client process can be further extended at the client by adding logic that is not implemented in the host server application. Local editing, automatic data entry, help capabilities, and other logic processes can be added in front of the existing host server application. If many errors are detected at the client, or functions such as online help are completely off loaded, the workload of the host server decreases. There is an opportunity to provide extensive interactive help and training integrated into a client/server application using only the services of the client workstation and NOS.

Completion of multipart forms often involves redundant data entry into multiple computer systems or applications. Collecting this data at the source or into a common data entry function and distributing it to the other data entry functions can reduce costs and errors. Ideally, the information is entered by the individual or process responsible for the data creation. This enables the individual with the knowledge to make corrections and to do so immediately. The workgroup LAN server captures the data and stores it. When a business process defmed to capture data from one copy of the form is invoked, the stored data is automatically merged into the form. This is updated, by the user, with additional data that is now available. In this manner, data is keyed only once and every business process uses the same data. Information is made available immediately after capture and can be distributed electronically to all authorized users.

Another commonly used technique to leverage the power and ease of use of the workstation is provided by tools. These tools provide easy-to-use facilities to manipulate data either stored on the existing host databases or downloaded to local servers. This technique of "data mining" through the use of powerful developer tools to provide rapid development of new management decision support functions, portends the future for systems development. Future developers will be *knowledge workers* =-technclogists with an equally strong business understanding using tools that are intuitive and powerful. Data will be provided to the workstation user in a form consistent with his or her. business understanding.

Why is workstation teclmology so effective? It supports the new business paradigm of employee empowerment. It provides the windowing capabilities to simultaneously access and display all information necessary to complete the business process. The capability of powerful workstation technology to recommend and make decisions based on historical precedent can dramatically reduce cost and improve service by shortening the decision-makingcycle.

4.4 Request for Service

Client workstations request services from the attached server. Whether this server is in fact the same processor or a network processor, the application format of the request is

the same. NOS software translates or adds the specifics required by the targeted requester to the application request.

Inter process communication (IPC) is the generic term used to describe communication between rnnning processes. In the client/server model, these processes might be on the same computer, across the LAN, or across the WAN.

The most basic service provided by the NOS is *redirection.* This service intercepts client workstation operating system calls and redirects them to the server operating system. In this way, requests for disk directories, disk files, printers, printer queues, serial devices, application programs, and named pipes are trapped by the redirection software and redirected (over the LAN) to the correct server location. It is still possible for some of these services to be provided by the client workstation. The local disk drives may be labeled A: and C: and the remote drives labeled D:, E:, and F:.

How does redirection work?

I. Any request for drive A: or C: is passed through to the local file system by the redirection software. Requests for other drives are passed to the server operating system. Printers are accessed through virtual serial and parallel ports defined by the NOS redirector software.

2. The NOS requester software constructs the remote procedure call (RPC).to include the API call to the NOS server.

3. The NOS server then processes the request as if it is executed locally and ships the response back to the application.

Novell commercialized this redirector concept for the Intel and MS-DOS platforms, and it has been adopted by all NOS and UNIX network file system (NFS) vendors. The simplicity of executing standard calls to a virtual network of services is its main advantage.

4.4.1 Remote Procedure Can (RPC)

Over the years, good programmers have developed modular code using structured techniques and subroutine logic. Today, developers want subroutines to be stored as a-named objects "some where" and made available to everyone with the right to use them. Remote procedure calls (RPC) provide this capability. RPC standardize the way programmers must write calls, so that remote procedures can recognize and respond correctly.

If an application issues a functional request and this request is embedded in an RPC, the requested function can be located anywhere in the enterprise that the caller is authorized to access. The RPC facility provides for the invocation and execution of requests from _processors running different operating systems and using hardware platforms different from that of the caller. Many RPC also provide data translation services. The call causes dynamic translation of data between processors with different physical data storage formats. These standards are evolving and being adopted by the industry.

4.4.2 Fax/Print Services

The NOS enables the client to generate print requests even when the printer is busy. These are redirected by the NOS redirector software and managed by the print server queue manager. The client workstation can view the status of the print queues at any time. Many print servers notify the client workstation when the print request is completed. Fax services are made available in exactly the same manner as print servers, with the same requester server interface and notification made available.

4.4.3 Windows Services

A client workstation may have several windows open on-screen at any time. The capability to activate, view, move, size, or hide a particular window is provided by the window services of the client operating system. These services are essential in a client/server implementation, because they interact with message services provided to notify the user of events that occur on a server. Application programs are written with no sensitivity to the windowing. Each application is written with the assumption that it has a

virtual screen. This virtual screen can be an arbitrary size and can even be larger than the physical screen.

The application, using GUI software, places data into the virtual screen, and the windowing services handle placement and manipulation of the application window. This greatly simplifies application development, because there is no need for the developer to build or manage the windowing services. The client user is totally in control of his or her desktop and can give priority to the most important tasks at hand simply by positioning the window of interest to the "front and center." The NOS provides software on the client workstation to manage the creation of pop-up windows that display alerts generated from remote servers. E-mail receipt, print complete, Fax available, and application termination are examples of alerts that might generate a pop-up window to notify the client user.

4.4.4 Remote Boot Services

Some applications operate well on workstations without any local disk storage; X-terminals and workstations used in secure locations are examples. The client workstation must provide sufficient software burned into erasable programmable read-only memory (E-PROM) to start the initial program load (IPL)-that is, boot-process. E-PROM is included in all workstations to hold the Basic Input/Output System (BIOS) services. This mini-operating system is powerful enough to load the remote software that provides the remaining services and applications functions to the client workstation or X-terminal.

4.4.5 Other Remote Services

Applications can be invoked from the client to execute remotely on a server. Backup services are an example of services that might be remotely invoked from a client workstation. Business functions such as downloading data from a host orchecking a list of stock prices might also be invoked locally to run remotely. Software is provided by the NOS to run on the client workstation toinitiate these remote applications.

Mobile computing is increasingly being used to remain functional while out of the office. With appropriate architectural forethought, applications can be built to operate effectively from the office LAN· or the remote laptop. Current technology supports full-

powered workstations with the capability for GUI applications consistent with the desktop implementation. The IPC protocol of choice for mobile access is TCP/IP based.

- Utility Services: The operating system provides local functions such as copy, move, edit, compare, and help that execute on the client workstation.
- Message Services: Messages can be sent and received synchronously to or from the network. The message services provide the buffering, scheduling, and arbitration services to support this function.
- Network Services: The client workstation communicates with the network through a set of services and APIs that create, send, receive, and format network messages. These services provide support for communications protocols, such as NetBIOS, IPX, TCP/IP, APPC, Ethernet, Token Ring, FDDI, X.25, and SNA These are more fully described in Chapter 5, "Components of Client/Server Applications-Connectivity."
- Application Services: In addition to the remote execution services that the NOS provides custom applications will use their own APIs embedded in an RPC to invoke specialized services from a remote server.

4.4.6 Database Services

Database requests are made using the SQL syntax. SQL is an industry standard language supported by many vendors. Because the language uses a standard form, the same application may be run on multiple platforms. There are syntactical differences and product extensions available from most vendors. These are provided to improve developer productivity and system performance and should be carefully evaluated to determine whether their uses are worth the incompatibility implied by using proprietary components. Using unique features may prevent the use of another vendor's products in a larger or smaller site. Certain extensions, such as stored procedures, are evolving into *de facto* standards.

The use of stored procedures is often a way of avoiding programmer use of proprietary extensions needed for performance. A clear understanding, by the technical

architects on the project, of where the standards are going is an important component of the SDE standards for the project.

4.4.7 Network Management Services-Alerts

Most network interface cards (NIC) can generate alerts to signify detected errors and perhaps to signify messages sent and received. These alerts are valuable in remote LAN management to enable early detection of failures. Because many errors are transient at first, simple remote detection may allow problems to be resolved before they become critical. Applications may also generate alerts to signify real or potential problems. Certain error conditions indicate that important procedures are not being followed. Application program failure may occur because current versions of software are not being used.

Support for a remote client workstation may be greatly simplified if alerts are generated by the applications. This should be part of every standard SDE. Many alert situations can be generated automatically from standard code without the involvement of the application developer. A more complete discussion of network management issues is included in the communications section of Chapter 5.

4.4.8 Dynamic Data Exchange (DDE)

DDE is a feature of Windows 3.x and OS/2 Presentation Manager that enables users to pass data between applications from different vendors through support for common APIs. For example, a charting package can be linked to a database to provide the latest chart data whenever the chart is referenced.

4.4.9 Object Linking and Embedding (OLE)

OLE is an extension to DDE that enables objects to be created with the object components software *aware.* Aware means that a reference to the object or one of its components automatically launches the appropriate software to manipulate the data. For example, a document created with a word processor may include an image created by a graphics package. The image can be converted to the internal graphics form of the word processor, such as WPG form for Word Perfect. With OLE, the image can be included in its original form within the document object; whenever the image is selected or highlighted,

the graphics package will take control to manipulate the image. Activation of the software is totally transparent to the users as they navigate through the document.

Currently with OLE, one software package accesses data created from another through the use of a *viewer* or *launcher.* These viewers and launchers must be custom built for every application. With the viewer, users can see data from one software package while they are running another package. Launchers invoke the software package that created the data and thus provide the full functionality of the launched software.

Both these techniques require the user to be aware of the difference between data sources. DDE and OLE provide a substantial advantage: any DDE- or OLE-enabled application can use any software that supports these data interchange APis. An e-mail application will be able to attach any number of components into the mail object without the need to provide custom viewers or launchers.

4.4.10 Common Object Request Broker Architecture (CORBA)

CORBA is a specification from the Object Management Group (OMG), a UNIX vendor consortium. OLE focuses on data sharing between applications on a single desktop, and CORBA addresses cross-platform data transfer and the process of moving objects over networks. CORBA support enables Windows and UNIX clients to share objects. A word processor operating on a Windows desktop can include graphics generated from a UNIX workstation.

CHAPTER5

THE FUTURE AND ADVANTAGES OF CLIENT/SERVER COMPUTING

5.1 The Future of Client/Server Computing

The single-system image is a reality. In the future, cheap and powerful workstation technology will be available to everyone with truly distributed applications using processing power wherever it is available and providing information wherever it is needed. In the future, information will be available for use by owners and authorized users, without the constant need for professional systems developers and their complex programming languages. The future will bring information captured at its source and available immediately to authorized users.

The future will provide information from data in its original form: image, video, audio, graphics, document, spreadsheet, or structured data, without the need to be aware of specific software for each form. Successful organizations of the future those that are market-driven and competitive will be ones using client/server as an enabling technology to add recognized value to their product or service. The future is now for early adopters of technology. By the turn of the century, the enterprise on the desk will be the norm for all successful organizations. Laggards will not be price competitive, will not provide competitive customer services, and soon will cease to exist.

5.1.1 Store for Netwo.rking Everyone's a Peer

Trends in computer hardware clearly indicate that D-RAM and processor MIPS are going to become very cheap. Object technologies based on the CORBA model and represented today by Sun's DOE project will enable the resources of a network of machines each processor available as client and server to participate in providing business solutions. Networked computing provides an opportunity for whole new classes of client/server computing. OS/2 the various versions of UNIX, and Windows NT provide the necessary components-shared memory, preemptive multitasking, database servers, communications servers, and GUI services. Suddenly, because of the conjunction oftl:ıese components, truly

distributed, peer-to-peer computing is a reality. Applications will find their servers without the need for application developer help

5.1.2 Store for Software Development Everything's an Object

Object-oriented development (OOD) can facilitate the system development environments (SDE) described throughout this book. The premise behind OOD is code reuse. The traditional concept of code reuse involves creating repositories of software that can be reused by developers. The object-oriented concept takes this traditional view and recycles it with greater formalism and improved repository management tools. The good news is that code reuse and OOD works; we have measured significant productivity improvements for development and maintenance, compared to standard development methodologies based on sound structured development practices. However, there is a steep learning curve that must be climbed before these gains are realized. OOD is not a new technology; it has been around for more than *15* years. A true OOD standard developing environment has yet to be established. Until this standardization occurs, the full potential of OOD described in this chapter will not be reached.

5.1.J Enabling Technologies

Client/server computing describes a model for building application systems, along with the core hardware and software technology that helps in building these systems. The material in the following paragraphs describes aggregations of these core technologies that have created enabling technologies. Enabling technologies are combinations of hardware and software that can be used to assist in creating a particular kind of application system.

1. Expert Systems

The main business advantage of using *expert systems* technology is tlıe opportunity to protect business know-how. Many organizations are severely short of experts. With the aging of the work force, and as a large number of experienced workers retire together, this shortage will become worse. Encapsulating the rules that describe an expert's response to a business situation into a stored electronic rules base, provides the substantive opportunity for higher productivity and reduced costs as these stored rules are consistently applied.

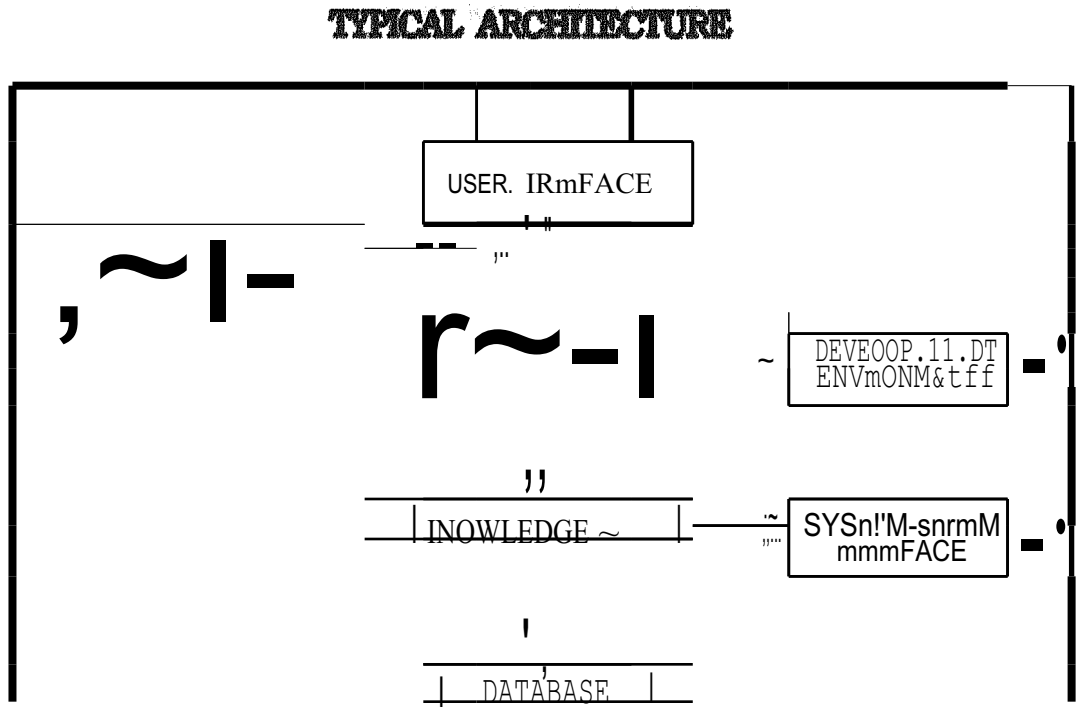


Figure 5.1 Typical expert systems application

Expert systems currently are used mainly in government and financial organizations. In the government sphere, knowledgeable personnel create rule bases to determine people's eligibility for programs such as welfare aid. Welfare programs/in particular, change these rules rapidly, and an expert system that manages and applies these rnles can improve the fairness and decrease the cost of adjudication for the program.

2. Geographic Information Systems

Geographic information systems (GIS) provide the capability to view the topology of a landscape, including features such as roads, sewers, electrical cables, and mineral and soil content. GIS is a technology that lıas promised much and finally is beginning to deliver. As with the expert systems technology, GIS are truly useful when they integrate with the business process. From a technological perspective, GIS must operate on standard technologies, integrate with the organization SDE, and directly access the organizational databases.

GIS applications are naturals for client/server technology. Powerful workstations manage the mapping. Connectivity enables shared access to the layers maintained by various departments. The GIS database is related to attributes stored in other databases that provide considerably more value in combination. For example, combining the voters list with the street maps allows polling booths to be located with easy access for all and ensures that no natural or artificial barriers are blocking the way.

3. Point-of-Service (POS)

Point-of-service (POS) technologies traditionally known as point-of-sale technologies are ubiquitous. Every restaurant, supermarket, most department stores, and even auto service stations use POS technology at tlıe site for pricing, staff management, accounting, product distribution, and inventory control. POS is one of the most widely installed examples of client/server technology. Implementations use an intelligent cash register, bar code scanner, scale, or gas pump as the client working with a UNIX or OS/2 server.

4. Imaging

Imaging is the conversion of documents from a physical medium (for example, paper) to a digital form where they can be manipulated by computers. Imaging should be viewed as an enabling technology. Information that is available in machine-readable form never should be conveıted to paper and scanned back into machine-readable fonu. The business process should strive to maintain and use information in machine-readable form from tlıe earliest moment.

Figure 5.2 shows a typical document imaging system. Information is entered into the system from a scanner. The scanner, similar to a fax machine, converts the paper image into digital form. This image is stored on a permanent medium, such as a magnetic or optical disk. Information must be indexed on entry so it can be located after it is stored

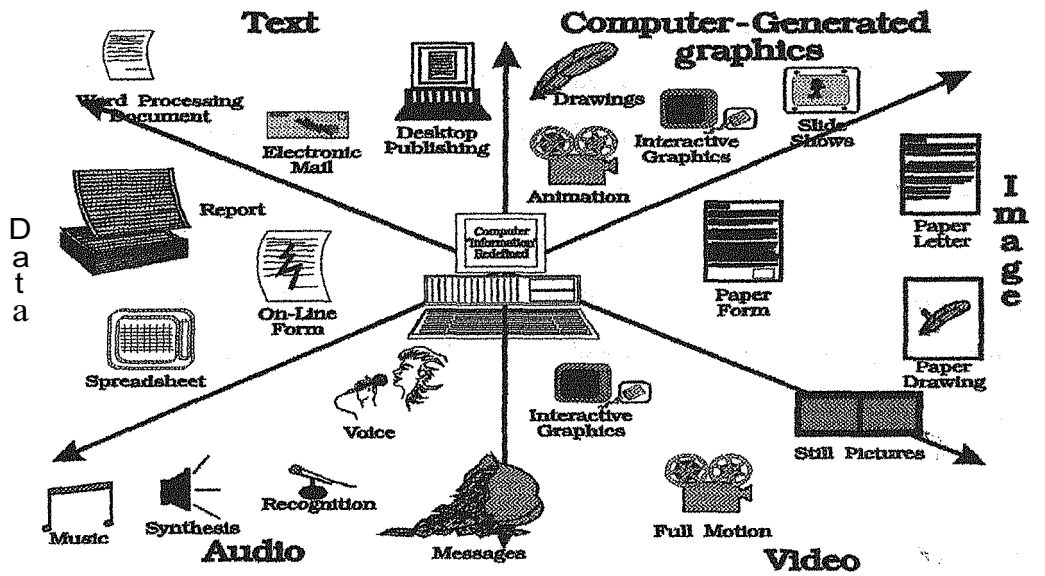


Figure 5.2 Typical imaging system

The index usually is stored in a relational database on a high-speed magnetic disk. Access to stored images is always initiated by an index search. High-resolution screens enable users to view the images after storage. Laser printers are used to fecre~t.ethe image on paper as required.

As Figure 5.2 illustrates, images can be accessed by any workstation with.access to the image server. Note that the image server replaces the filing cabinet but provides the additional advantage of allowing multiple access to the same documents or folders. The movement toward standards for the creation, distribution, indexing, printing, display, and revision of images has enabled a large number of vendors to enter the market with products. This has led to a dramatic reduction in the price of these components.

5. Electronic Document Management Multimedia

The concepts of electronic image management relate to the manipulation of information contained in forms, blueprints, x-rays, microfilm, fingerprints, photographs, and typewritten or handwritten notes. Electronic document management adds the capability

to manipulate information from other media, such as audio and video. In addition, the "folder" device gives the user an intuitive interface to information equivalent to, but more flexible than, a filing cabinet. The information is always available on the desktop. Several users can use the folder simultaneously. Folders are always re filed in the correct location. Billions of these documents exist and are used daily in the process of providing government services. Consider that the Los Angeles County municipal hospitals alone have 5 billion pieces of paper, x-rays, scans, photographs, audio reports, and videos (and so on) filed to maintain patient records. Currently, the cost of this technology is prohibitively high for most organizations, but these systems will come down in price as all computer components do.

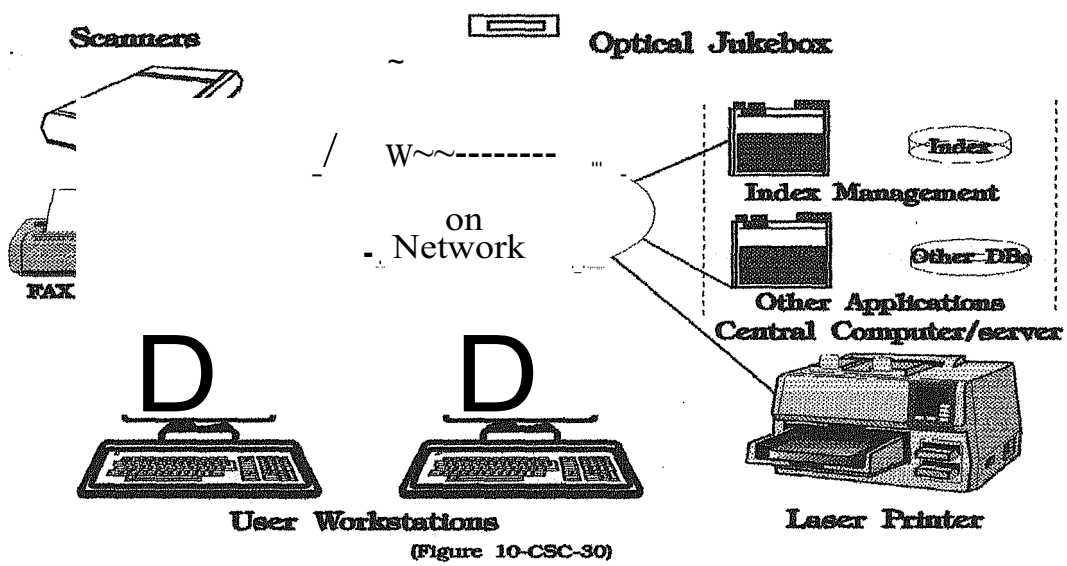


Figure 5.3 Multimedia technologies

High-speed communications networks can provide the capability to distribute information other than voice conversations throughout a county, state, or country. With the advent of fiber-optic cabling, the capacity for information distribution to a location, office, library, or home is essentially infinite. As this technology become readily available, we will be able to consider where best to store and use information without concern for transmission time or quality. This is particularly true within a small geographical area, such

as a county where the "right of way" is owned and private fiber-optic networks can be installed. High-speed networks iii conjunction with new standards for data integrity ensure that information can be stored throughout the network and properly accessed from any point iii the network

6. Full-Text Retrieval

An area of explosive growth, coincident with the availability of high-powered workstations and RISC servers, is full-text retrieval. Originally a technology used by the military to scan covert transmissions and media communications, full-text retrieval is now a mainstream technology. Vendors such as Fulcrum and PLS have packaged their technology to be used iii more traditional business applications. Northern Telecom bundles a product, Helmsman, with all its switch documentation to facilitate document access. All major news services provide an electronic feed of their information. This information is continuously scanned by reporters, businesses, and government offices to identify significant events or locate trends. Dow Jones provides their news retrieval system with access to 10**12bytes (that's three billion pages) of textual data. Many criteria searches can be run against all this text in a few seconds. Figure 10.6 illustrates the flow of information in a full-text retrieval application.

5.2 Tran sformational Systems

In the more than 40 years since the introduction of the stored program computer in 1951, we have seen tremendous advances in the capabilities of this technology. Computers have proven over and over that they can add numbers at mind-numbing rates. We have extrapolated from this capability the functionality to maintain accounts, calculate bills, print checks, and create memos. All this functionality has enabled organizations to grow and do more work with fewer clerical and administrative staff

As the world economy becomes more integrated, goods and services are provided by companies arıd individuals from all parts of the world. Consumers can and will buy the most cost-effective quality product and service available. This substantially increases the necessity for organizations to demonstrate their value. Western economies, with their higher salaries and cost of plant, are particularly threatened by this trend. However,

Western economies have the advantage of a highly educated population. Educated staff are willing to accept decision-making responsibility and are better able to adapt to change. The challenge is to find ways in which technology can enable the West to capitalize on these advantages.

5.2.1 Emergency Public Safety

Emergency (E911) dispatch operators are responsible for sending the right emergency response vehicles to an incident as quickly as possible and at the same time dealing with the crisis being reported over the telephone. This functionality must be provided 24-hours-per-day, 365-days-per-year, with the maximum possible performance.

Through the use of client/server computing, it is now possible to duplicate all of the functionality of such an existing traditional design with the additional advantages of better performance, a graphical user interface (GUI), a single point of contact, higher reliability, and lower costs. With a client/server-based system, the dispatch operator is empowered to oversee how staff and equipment are allocated to each incident.

The implementation of such an E911 service can dramatically improve the rate at which emergency calls can be answered and reduce the incidence of unnecessary dispatches. Workstation technology provides the dispatcher with a less stressful and more functional user interface. The dispatcher can respond quickly to changes in the environment and communicate this information immediately to the vehicle operator. The system is remarkably fault-tolerant If a single workstation is operating, the dispatcher can continue to send emergency vehicles to the incident. This architecture is general-enough to apply to any application that has reasonable quantities of transient data.

5.2.2 Electronic Data Interchange

Electronic data interchange (EDI) technology enables unrelated organizations to conduct their business computer to computer without the need to use the same computer applications or technology in all locations. Combining *just in time* (JIT) manufacturing with EDI truly transforms the process:

1. A salesperson accepts an order through a laptop computer system.

2. Using the electronic mail facilities of the organization, the order is shipped to the order entry system.

3. The component parts are determined, and electronic purchase orders are generated to each supplier.

4. The EDI link routes the order, which is processed by the supplier's order system.

5. An EDI link is used to validate the purchaser's credit worthiness.

6. The paperless invoice is sent by way of EDI back to the purchaser.

7. When it is time for payment, an EDI link is used to generate the electronic funds transfer (EFT) to pay.

With EDI, a single entry by the person closest to the customer causes the facilities of the manufacturer and its suppliers to schedule appropriate production, shipping, and billing. The maximum possible time is allowed for all parties to process the order, thus reducing their need to carry inventory. A further advantage comes when production is driven by orders, because only those products that will actually be sold are manufactured. Manufacturers are able to offer more flexibility in product configuration, because they are manufacturing to order. The use of EDI standards allows organizations to participate in this electronic dialog regardless of differences among their individual technologies or application systems.

S.2.3 Financial Analysis

Financial analysts are overloaded with data. It is impossible for them to process all the data received. They must read it, looking for gems of information. Powerful workstation technology enables these analysts to specify personal filters to be applied against the data in order to present only information of likely interest and to present it in order of most likely interest. These filters provide search criteria specific to each analyst and provide only information satisfying the filter criteria to the analyst

Improvements in technology enable the data to be scanned in real time. Alerts can be generated to the analyst whenever a significant event is detected. In this way, the

analyst's job is transformed. He or she is now concerned with developing the rules to drive the filters and with understanding how to react to the significant events that are detected. Meaningful and useful data is available to support the analyst's decision making. He or she has more time to make informed decisions

5.3 Advantages of Client/Server Computing

Organizations want to take advantage of the low cost and user-friendly environment that existing desktop workstations provide. There is also a strong need and desire to capitalize on existing investment at the desktop and in the portfolio of business applications currently running in the host. Thus, corporate networks are typically put in place to connect user workstations to the host. Immediate benefits are possible by integrating these three technologies: workstations, connectivity, and hosts. Retraining and redevelopment costs are avoided by using the existing applications from an integrated desktop.

Client/server computing provides the capability to use the most cost-effective user interface, data storage, connectivity, and application services. Frequently, client/server products are deployed within the present organization but are not used effectively. The client/server model provides the technological means to use previous investments in concert with current technology options. There has been a dramatic decline in the cost of the technology components of client/server computin.g.Organizations see opportunities to use technology to provide business solutions. Service and quality competition in the marketplace further increase the need to take advantage of the benefits available from applications built on the client/server model.

Client/server computing in its best implementations.moves the data-capture and information-processing functions directly to the knowledgeable worker that is, the worker with the ability to respond to errors in the data, and the worker with the ability to use the information made available. Systems used in the front office, directly involved *in* the process of doing the business, are forced to show value. If they don't, they are discarded under the cost pressures of doing business. Systems that operate in the back room after the business process is complete are frequently designed and implemented to satisfy an administrative need, without regard to their impact on business operations. Client/server applications integrate the front and back office processes because data capture and usage

become an integral part of the business rather than an after-the-fact administrative process. In this mode of operation, the processes are continuously evaluated for effectiveness. Client/server computing provides the technology platform to support the vital business practice of continuous improvement.

2.3.1 The Advantages of Client/Server Computing

The client/server computing model provides the means to integrate personal productivity applications for an individual employee or manager with specific business data processing needs to satisfy total information processing requirements for the entire enterprise.

- Enhanced Data Sharing

Data that is collected as part of tlıe normal business process and maintained on a server is immediately available to all authorized users. The use of Structured Query Language (SQL) to define and manipulatethe data provides support for open access from all client processors and software. SQL grants all authorized users accesstotheinformation through a view that is consistent with their business need. Transparent network services ensure that the same data is available with the same currency to all designated users.

- Integrated Services

In the client/server model, all information that the client (user) is entitled to use is available at the desktop. There is no need to change into terminal mode orlog intoanöther processor to access information. All authorized information and processes areJ.lir¢ctly available from the desktop interface. The desktop tools e-mail, spreadsheet, presentation graphics, and word processing are available and can be used to deal with iıifonnation provided by application and database setvets resident on the network. Desktop users can use their desktop tools in conjunction with information made available from the corporate systems to produce new and useful information.

Another excellent and easily visualized example of the integration possible in the client/server model is implemented in the retail automobile service station. Figure 5.4 illustrates the comprehensive business functionality required in a retail gas service station.

The service station automation (SSA) project integrates the services of gasoline flow measurement, gas pumps billing, credit card validation, cash registers management, point-of-sale, inventory control, attendance recording, electronic piice signs, tank monitors, accounting, marketing, truck dispatch, and a myriad of other business functions. The system uses all of the familiar client/server components, including local and wide-area network services. Most of the system users are transitory employees with minimal training in computer technology. The service station automation system is a classic example of the capabilities of an integrated client/server application implemented and working today.

IN1EGRA1ED REr AIL OUI'LE:r SYSTEM **ARCHITECTURE**

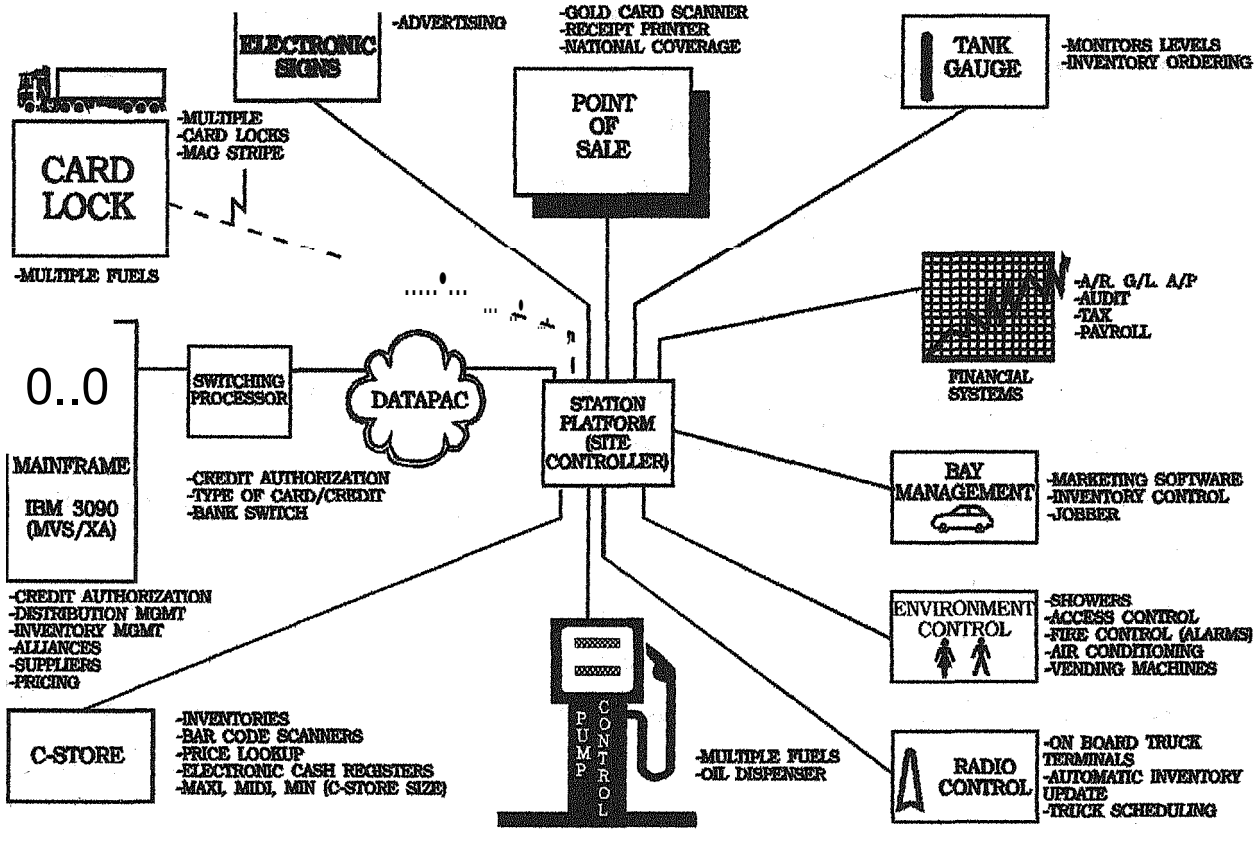


Figure 5.4 Integrated retail outlet system architecture.

- Sharing Resources Among Diverse Platforms

The client/server computing model provides opportunities to achieve true open system computing. Applications may be created and implemented without regard to the hardware platforms or the technical characteristics of the software. Thus, users may obtain client services and transparent access to the services provided by database, communications, and applications servers. Operating systems software and platform hardware are independent of the application and masked by the development tools used to build the application.

Client/server applications operate in one of two ways. They can function as the front end to an existing application the more limited mainframe-centric model discussed in Chapter 1 or they can provide data entry, storage, and reporting by using a distributed set of clients and servers. In either case, the use or even the existence of a mainframe host-is totally masked from the workstation developer by the use of standard interfaces such as SQL.

- Data Interchangeability and Interoperability

SQL is an industry-standard data definition and access language. This standard definition has enabled many vendors to develop production-class database engines to manage data as SQL tables. Almost all the development tools used for client/server development expect to reference a back-end database server accessed through SQL. Network services provide transparent connectivity between the client and local or remote servers. With some database products, such as Ingress Star, a user or application can define a consolidated view of data that is actually distributed between heterogeneous, multiple platforms.

The client/server model provides the capability to make ad hoc requests for information. As a result, optimization of dynamic SQL and support for distributed databases are crucial for the success of the second generation of a client/server application. The first generation implements the operational aspects of the business process. The second generation is the introduction of ad hoc requests generated by the knowledgeable user looking to gain additional insight from the information available.

- Masked Physical Data Access

When SQL is used for data access, users can access information from databases anywhere in the network. From the local PC, local server, or wide area network (WAN) server, data access is supported with the developer and user using the same data request. The only noticeable difference may be peıformance degradation if the network bandwidth is inadequate. Data may be accessed from dynamic random-access memory (D-RAM), from magnetic disk, or from optical disk, with the same SQL statements. Logical tables can be accessed without any knowledge of the ordering of columns or awareness of extraneous columns by selecting a subset of the columns in a table. Several tables may be joined into a view that creates a new logical table for application program manipulation, without regard to its physical storage format

The use of new data types, such as binary large objects (BLOB), enables other types of information such as images, video, and audio to be stored and accessed using the same SQL statements for data access. RPC frequently include data conversion facilities to translate the stored data of one processor into an acceptable format for another.

- Location Independence of Data and Processing

We are moving from the machine-centered computing era of the 1970s and 1980s to a new era in which PC-familiar users demand systems that are user-centered. Previously, a user logged into a mainframe, mini-, or micro-application. The syntax of access was unique in each platform. Function keys, error messages, navigation methods, security, performance, and editing were all very visible. Today's users expect a standard "look and feel." Users log into an application from the desktop with no concern for the location or technology of the processors involved.

In the 1970s, users logged into the IBM mainframe, the VAX minicomputer, or one of the early microcomputer applications. It was evident which platform was being used. Each platform required a unique login sequence, security parameters, keyboard options, and custom help, navigation, and error recovery. In the current user-centered world, the desktop provides the point of access to the workgroup and enterprise services without regard to the

platform of application execution. Standard services such as login, security, navigation, help, and error recovery are provided consistently among all applications.

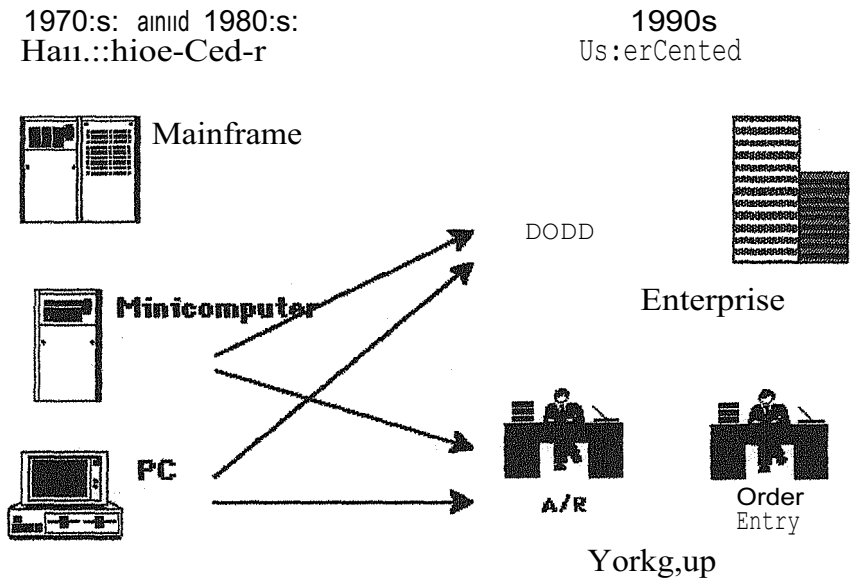


Figure 5.5 The computing transformation.

Developers today are provided with considerable independence. Data is accessed through SQL without regard to the hardware, operating system, or location providing the data. Consistent network access methods envelop the application and SQL requests within an RPC. The network may be based in Open Systems Interconnect {OSI), Transmission Control Protocol/Internet Protocol (TCP/IP), or Systems Network Architecture (SNA), but no changes are required in the business logic coding. The developer of businesslogic deals with a standard process logic syntax without considering the physical platform.

The application developer deals with the development language and uses a version of SDE customized for the organization to provide standard services. The specific platform

characteristics are transparent and subject to change without affecting the application syntax.

- Centralized Management

As processing steers away from the central data center to the remote office and plant, workstation server, and local area network (LAN) reliability must approach that provided today by the centrally located mini- and mainframe computers. The most effective way to ensure this is through the provision of monitoring and support from these same central locations. A combination of technologies that can "see" the operation of hardware and software on the LAN monitored by experienced support persomıel provides tlıe best opportunity to achieve the level of reliability required.

5.3.2 Technelegy Revolution

The changes in computer technology that have taken place during the past five years are significantly greater than those of the preceding 35 years of computer history. There .is no doubt tlıat we will continue to experience an even greater rate of change during the coming five-year period.

- Future Teclıınôlôgies

Consulting a crystafball, projecting the future, and making decisions based on the projections a common failure of the computer industry. Predicting the future is a risky business. Industry leaders, technicians, and investors have been equally unsuccessful on occasion.

It is important, however, to achieve an educated view of where technology is headed during the life of a new system. The architecture on which a new system is built must be capable of supporting all users throughout its life. Large organizations traditionally have assumed that their applications will provide useful service for 5 to 10 years. Many systems are built with a view of only what is available and provable today, and they are ready to fall apart like a deck of cards when the operatiıng environment changes and the architecture cannot adapt to the new realities

- Computing Power Compared

A 1990 survey of U.S. Fortune 1000 companies, completed by a well-known computer industry research firın, found that on an MIPS (millions of instructions per second) basis, more than 90 percent of the processing power available to organizations exists at the desktop. This clıeap computing power is typically underused today. It is a sunk cost available to be used as clients in the implementation of client/server applications.

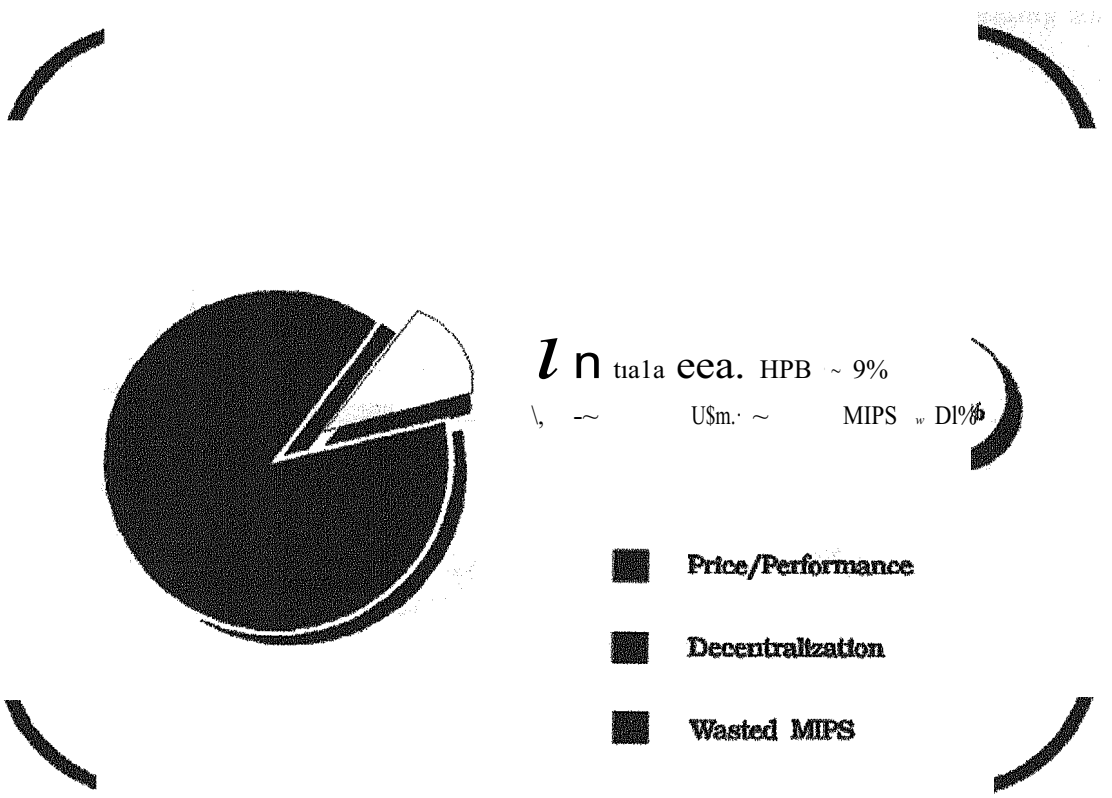


Figure 5.6 Managing the shift to distributed processing.

Figure *5.6* illustrates the portion of processor capacity allocated to the central site and the desktop. In most organizations, the 9 percent of processor capacity residing in the "glass house" central computer center provides 90 percent or more of enterprise computing. The 90 percent of processor capacity on the desktop and installed throughout the organization provides less than 10 percent of the processing power to run the business.

Most workstation systems are used for personal productivity applications, such as word processing, presentation graphics, and spreadsheet work. The personal productivity functions performed on these machines typically occupy the processor for a maximum of two to three hours per day.

- Input/Output (I/0) Compared

Most applications require information that is manipulated also to be read and saved. In the next example, added to the CPU processing is requirement to perform 1000 physical data read or write, operations per second. Figure 2.8 shows the costs of performing these operations.

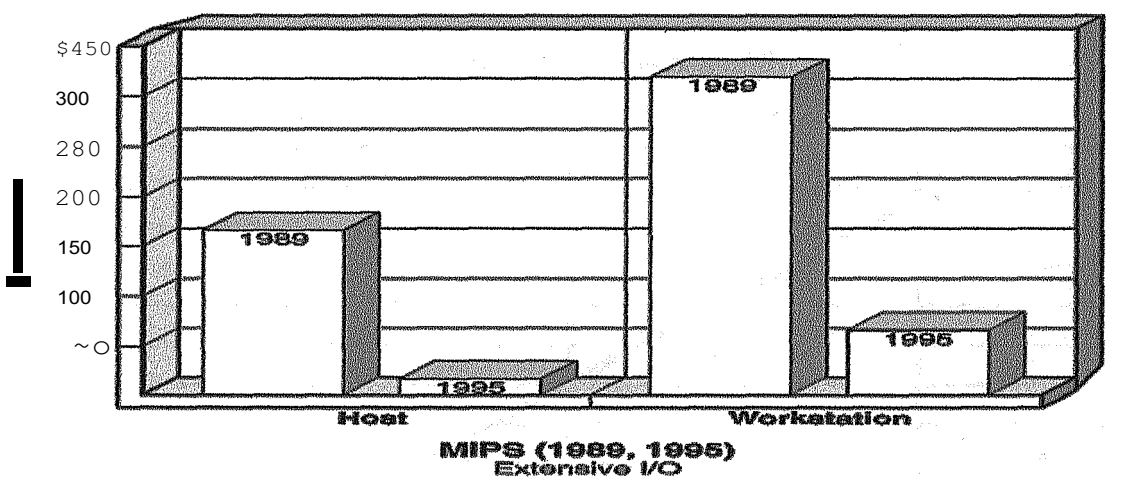


Figure 5.7 The I/O bottleneck.

The same portion of the mainframe configuration required to provide one MIPS execution capability can simultaneously handle this I/O requirement. The workstation configuration required to simultaneously handle these two tasks in 1989 cost at least twice that of the mainframe configuration. In addition, the configuration involved multiple processors without shared memory access. In order to preserve data integrity, the I/O ınust be read only. The dramatic reduction in workstation cost projected in 1995 is predicated on the use of symmetric multiprocessors to provide CPUs with shared memory and 011 the use of coprocessors providing the cached controllers necessary to support parallel I/0. (Parallel

I/O enables multiple .I/O requests to several devices to be serviced concurrently with host CPU processing.) However, the costs are still projected to be 75 percent greater than costs on the mainframe for this high rate of I/0.

Only through the effective use of real storage (D-RAM) can we hope to use the available CPU power. Data can be accessed from D-RAM without the need to do physical I/O except to log the update. Database technology uses a sequential log to record changes. These sequential writes can be buffered and done very rapidly. The random updates to the database are done when the system has nothing better to do or when the shared D-RAM containing the updated data is required for other data. The log is used to recover the database after any failure that terminates the application.

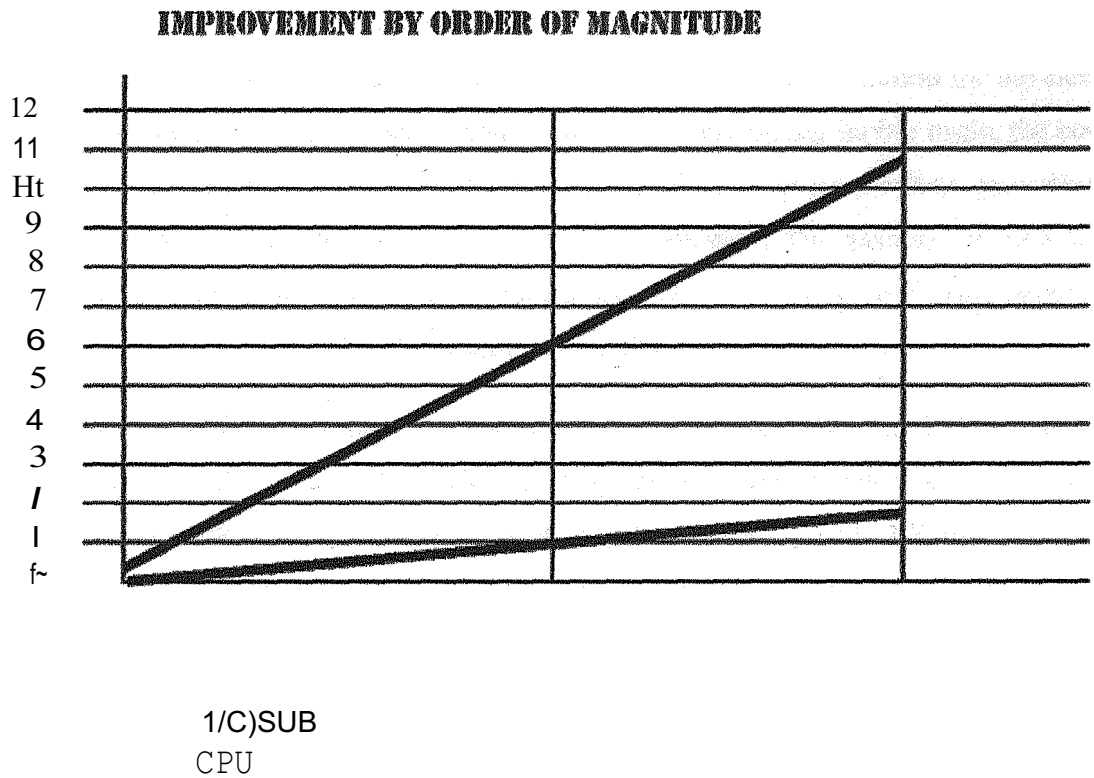


Figure 5.8 Processor power versus I/O capacity.

Workstation technologies can deal with personal data. Data extracted from central systems for analysis by the end user, data from integrated external sources for comparison,

and integrated new types of data such as voice annotation to documents. All these data forms provide additional uses for lower-cost, permanent data storage. Decision-support systems can use workstation technologies and massive amounts of additional data to provide useful, market-driven recommendations.

- Main Storage

Arguably, the most dramatic technological revolution affecting the computer industry today is caused by the increase in the amount of main storage (D-RAM) available to an application. D-RAM is used for the execution of programs and the temporary storage of permanent data.

Computer users have entered the era of very large and inexpensive D-RAM. Figure 5.9 represents the manner in which this technology has evolved and continues to evolve. Every three years, a new generation of D-RAM technology is released. Each new generation is released with four times the capacity of the previous generation for the same chip price. At the point of .introduotion and at any given time during its life cycle, the cost of these chips is reduced to a price equal to the price of chips from the previous generation. As the capacity of individual D-RAM cliips has increased, the quantity of D-RAM available to the client (and server) has increased massively. Laboratory and manufacturing evidence reveals that this trend will continue at least through 1996.

Desktop workstations purchased in 1988 with 1 megabit (M-bit) D-RAM chips were available in 1992 with 4Mbit DRAM chips for the same or lower cost In 1988, typical desktop workstations contained 1 to 4 megabytes (Mbytes) of D-RAM. In 1992, these same configurations contain from 4 to 16Mbytes. In 1995, these configurations will use 16Mbitchips and be available with 16 to 64Mbytes for the same price. By 1998 within the life span of many applications being developed today-these configurations will use 64Mbit chips and contain from 64 to 256Mbytes ofD-RAM for the same price.

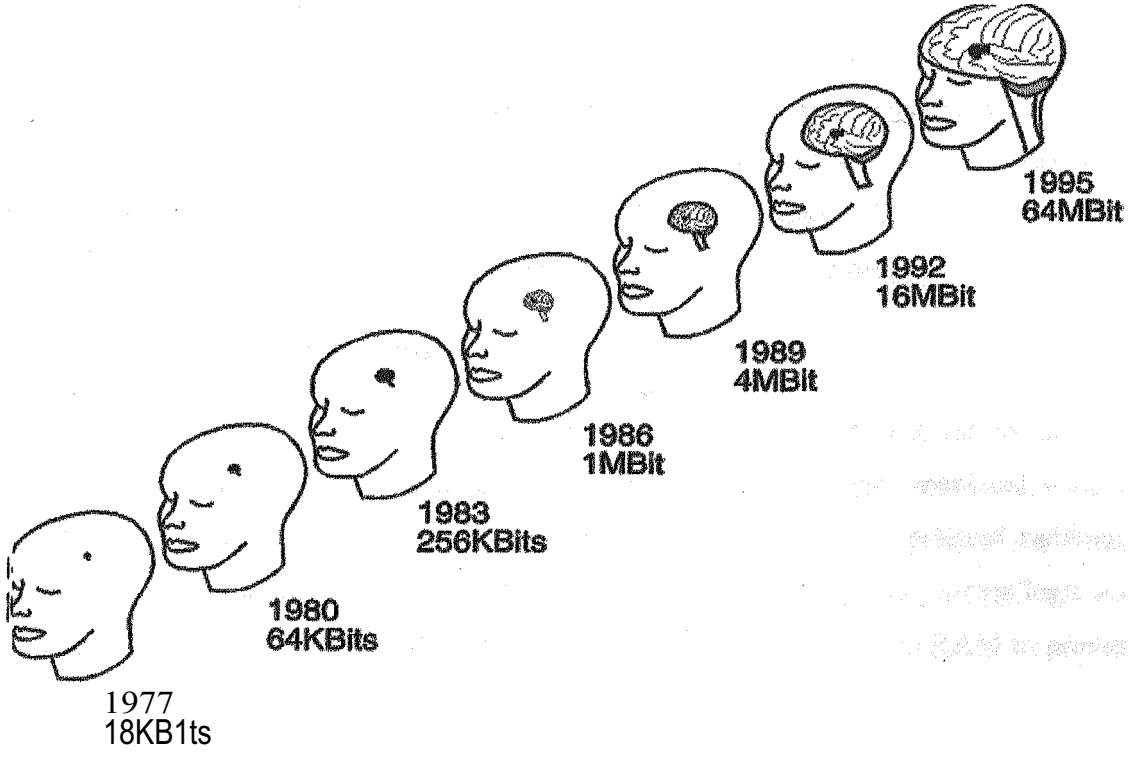


Figure 5.9 D-RAM chip evolution.

A revolutionary change is occurring in our capabilityto provide functionality at the desktop. Most developers cannot generate any where near the amount of code necessary to fill a 64Mbyte processor on the desk. Yet applications being built today will be used on desktop processors with this amount of D-RAM. The client workstation can now contain in D-RAM all the software that the user will want to use. This eliminates the delay that was previously inherent in program switching that is. program loading and startup. It is now practical to use a multitasking client workstation with several active tasks and to switch regularly among them. Virtual storage is a reality. Workstation D-RAM costs were less than $50 per megabyte in 1992. The cost difference for an additional 4 megabytes is only $200. Only one year earlier, short-sighted application designers may have made system design decisions based on a cost of $1000 for 4Mbytes.

- Software Trends

To achieve the benefit of this advance in technology, organizations must choose software that can use it. Traditional development tools, operating systems, character mode user interfaces, and non-SQL-based database teclmology cannot take advantage of this quantity ofD-RAM and the power available from workstation technology.

- Graphical Screen Designs

Graphical user interfaces (GUis) require large amounts of D-RAM to hold the screen image, pull-down lists, help text, navigation paths, and logic associated with all possible selectable events. Because a GUI enables processingto.be selected?faıtdo.ıılly rather than in the traditional sequential, top-to-bottom order, all possible prqçess'.Jögic.and GUI management code associated with the image must be available in O„R..A.Mto.Provide appropriateresponses.

CICS developers do not good GUI developers make.3 GUI application development requires a special mindset Education, experience, and imagination are prerequisites for moving from the character mode world to the GUI world. Laying out a character mode screen requires that fields are lined up row to row and the screen is not cluttered with too many fields. GUI layout is more difficult, because there are so many options. Colors, pull-down lists, option buttons, text boxes, scrollbars, check boxes, and multiple windows are all layout capabilities. The skills that a layout artist commonly possesses are more appropriateto the task than tlıose which a programmer usually demonstrates.

- Relational Databases

Another dramatic change in software is in the area of database management, Traditional file system and database technologies rely on locality of reference for good performance in accessing data. Locality of reference implies that all data needed to satisfy a request is stored physically close together. However, today's business environmentrequires multi keyed access to rows of information derived from multiple tables. Performance is only possible in these environments when database searches are performed in main storage

using extracted keys organized into searchable lists. Physical access to the database is restricted to the selection of rows that satisfy all search criteria.

Relational systems can and do perform, but poor standards of use can defeat them. An example of successful performance, this book has implemented an application, described in appendix (A). That processes more than 400npdate transactions per second into a five-table relational database view. This specific example is implemented under DB2 on a midsize ES9000 processor.

2.3.3 Connectivity

The era of desktop workstations began in 1981 with the introduction oftheIBM personal computer (PC). The PC provided early users with. •·tlıe}par,~pilityJç, do spreadsheets, word processing, and basic database services for perso:naldata. With.in three years, it became clear that high-quality printers, backup tapes, high-capacıty cliskdevices, and software products were too expensive to put on everyone's desktop.J.,AN techıı ology evolved to solve this problem. Novell is and lias been the most successful vendor in the LAN market.

1. Step 1 Workstations Emulate Corporate Systems

Figure 5.10 shows the trend in the introduction of PCs into organizations during the period from 1980 until 1995. In most large organizations, desktop workstations provide personal productivity and some workgroup functions, but host services still provide most other business functions. The lack of desktop real estate encourages the addition of terminal emulation services to the workstation. This emulation capability connects the workstation directly to the corporate systems. The connectionwas and generally still is provided by .a direct connection from the workstation to the host server or its controller. It is possible to use a sub-5,000 workstation as a 500 dumb terminal.

Connectivity provides the opportunity to move beyond terminal emulation to use the foll potential of the workstation. Often the first client/server applications in a large organization use existing mainframe applications. These are usually presentation services-only applications.

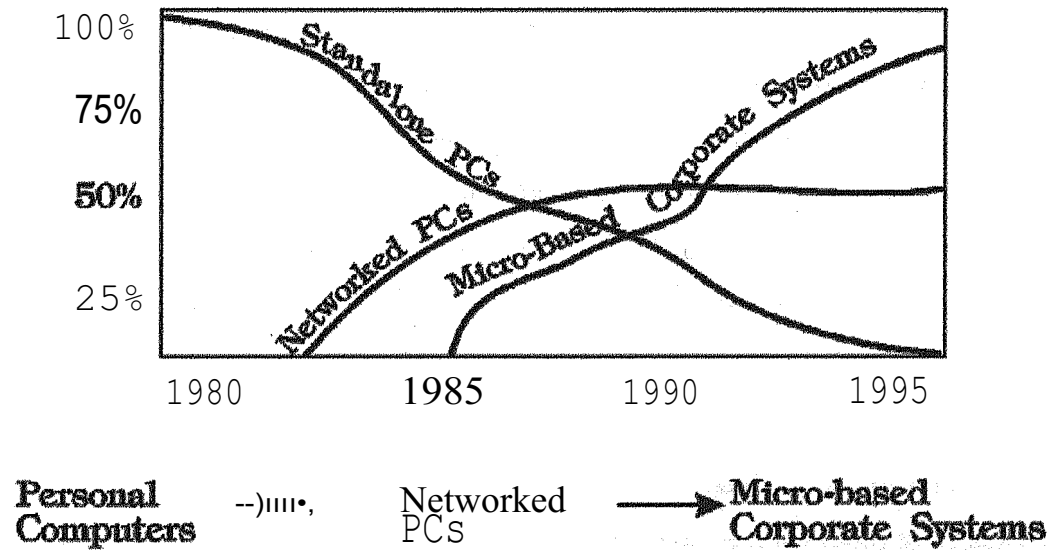


Figure 5.10 Trends in PC-micro expenditures.

2. Step 2 Adding Servers for Database and Communications

The next step in connectivity is the implementation of specialized servers to provide database and communications services. These servers provide LAN users with a common database for shared applications and with a shared node to connect into the corporate network. The communications servers eliminate the need for extra cabling and workstation hardware to enable terminal emulation. The LAN cabling provides the necessary physical connection, and the communications server provides the necessary controller services.

3. Step 3 Full-Fledged Client/Server Applications

With its implementation of communications and database servers in place, an organization is ready for the next step up from presentation services-only client/server applications to full-fledged client/server applications. These new applications are built on the architecture defined as part of the system development environment (SDE).

5.3.4 User Productivity

Personal computer users are accustomed to being in control Of their environment. Recently, users have been acclimated to the GUI provided by products. Productivity is enhanced by the standard look and feel that most applications funning in these environments provide. A user is trained both to get into applications and to nıove from function to function in a standard way. Users are accustomed tôthe/availabilityôfcontext-sensitive help, "friendly" error handling, rapid performance, and flex.ıbi.lityl

The personal computer user makes the change and sees the result. The mainframe programmer must make the change, compile the program, invoke the program, and run the test. If the user understands the request, the implications, and the syntactical requirements, he or she may get it right the first time. Usually, it takes several iterations to actually get it right, often in concert with. a frustrated user who tries to explain the real requirement.

We aren't suggesting that all applications can be developed by nonprogramm using desktop-only tools. However, now that it has become rather easy to build these types of applications on the desktop, it is important for professional IS people to understand the expectations raised in the minds oftlıe end-user community.

5.3.5 Ways to Improve Performance

Client/server-developedapplications may achieve substantially greater perförınance when compared with traditional workstations or host-only applications.

1. OffloadWork to Server

Database and communications processing are frequently offloaded to a faster server processor. Some applications processing also may be offloaded, particularly for a complex process, which is required by many users. The advantage of offloading is realized when the processing power of the server is significantly greater than that of the client workstation. Shared databases or specialized communications interfaces are best supported by separate processors. Thus> the client workstation is available to handle otlıer client tasks. These advantages are best realized when the client workstation supports multitasking or at least easy and rapid task switching.

2. Reduce Total Execution Time

Database searches, extensive calculations, and stored procedure execution can be performed in parallel by the server while the client workstation deals directly with the current user needs. Several servers can be used together, each performing a specific function. Servers may be multiprocessors with shared memory, which enables programs to overlap the LAN functions and database search functions.dn general, the increased power of the server enables it to perform its functions faster-thanthe client workstation. In order for this approach to reduce the total elapsed time, the additional time.required to transmit the request over the network to the server must be less than the saving: High-speed local area network topologies operating at 4, 10, 16, or 100Mbs(megabits per second) provide high-speed communications to manage the extra traffic in less time than the savings realized from the server. The time to transmit the request to the. Sef\let,>e,2~cuf&ftr~guest, and transmit the result to the requestor, must be less than the time to perform the entire transaction on the client workstation.

3. Use a Multitasking Client

As workstation users become more sophisticated, the capability to be simultaneously involved in multiple processes becomes attractive. Independent tasks can be activated to manage communications processes, such as electronic mail, electronic feeds from news media and the stock exchange, and remote data collection (downloading from remote servers). Personal productivity applications, such as word processors, spreadsheets, and presentation graphics, can be active. Several of these applications can be dynamically linked together to provide the desktop information processing environment. Functions such as Dynamic Data Exchange (DDE) and Object Linking and Embedding (OLE) permit including spreadsheets dynamically into word-processed documents. These links can be *hot* so that changes in the spreadsheet cause the word-processed document to be updated, or they can be *cut and paste* so that the current status of the spreadsheet is copied into the word-processed document.

5.3.6 Vendor Independence

If client and server functionality is clearly split and standards-based access is used, there can be considerable vendor independence among application components. Most organizations use more expensive and more reliable workstations from a mainstream vendor such as Compaq, IBM, Apple, Sun, or Hewlett-Packardfortlıeir servers .. Other organizations view client workstation technology as a commodity arıd S~lecf.low~r-priced and possibly less-reliable vendor equipment. The mainstream vendôrs?~aver2e~~0dJhis trend and are providing competitively priced client workstations. Eaclıof the.nınainstr:eam vendors reduced its prices by at least 65 percent betweens (1991..1993)} pfirtıarilyi in response to an erosion of market share for client workstations.

The resulting shakeout in the industry has significantly reduced the tıuttıber of vendors and makes the use of traditionally low priced clones very risky. Hardware can generally be supported by third-party engineers, but software compatibility is a serious concern as organizations find they are unable to install and run new products.

An excellent example of this independence is the movement of products such aS FoxPro and Paradox to use client services to invoke, through SQL, the serverfunctions provided by Sybase SQL Server. A recent survey of client development products that support the Sybase SQL Server product identified 129 products. This is a result of the openness of the API provided by Sybase. Oracle also has provided access to its API, and several vendors-notably Concentric Data Systems, SQL Solutions, and Database have developed front-end products for use witlı Oracle. ASK also has realized the importance of open access to buyers and is working with vendors such as Fox and PowerBuilder to port their front ends in support of the Ingress database engine.

5.J. 7 Faster Delivery of Systems

Some software development and systems integration vendors have had considerable success using client/server platforms for the development of systems targeted completely for mainframe execution. These developer workstations are often the first true client/server applications implemented by many organizations. The workstation environment, powerful multitasking CPU availability, single-user databases, and integrated testing tools all

combine to provide the developer with considerable productivity improvements in a lower-cost environment, Our analysis shows that organizations that measure the "real" cost of mainframe computing will cost justify workstation development environments in 3 to 12 months.

Client/server application development shows considerable productivity improvement wlıeıı the software is impleıneııted within an SDE. As previously noted, orgaıiizational standards-based development provides the basis for object-oriented development techniques and considerable code reuse. This is particularly relevant in the client/server model, because some natural structuring takes place with the division of functionality between the client and server environments. Reuse of the server application functionality, database, and network services is transparent and airtıostautortıattctBecause the applications are built with little regard to standard ïront-enn functionality, many features are part of the standard GUI and are automatically reused.

- Smaller and Simpler Problems

Client/server applications frequently are involved with data tion or data analysis. In such applications, tlıe functionality is personal to a single user or a few users. These applications frequently can be created using standard desktop products with minimal functionality. For example, data may be captured directly into a form built with a forms development tool, edited by a word processor, and sent on through the e-mail system to a records management application. In the back end, data may be downloaded to a workstation for spreadsheet analysis.

CONCLUSION

Client in the client/server model is the desktop workstation. Any workstation th.at is used by a single user is a client. The same workstation, when shared simultaneouslyby multiple users, is a server, The server is a multi user computer. There is no special hardware requirement that turns a computer into a/server. The hardware platform should be selected based on application demands and eco1lon1ics. Servers for client/server applications work best when they are configured with an Operating system that supports shared memory, application isolation, and preemptive nfüfütasking.

Client/server computing provides the capaon ity to use the most cost effective user interface, data storage, connectivity, and application services. Frequently, client/server products are deployed within the present ori:,ranization but are not used effectively. The client/server model provides the technological means to use previous investments in concert with current technology options. There has been a dramatic decline in the cost of the technology components of client/server computing. Organizations see opportunities to use technology to provide business solutions. Service and quality competition in the market place further increase the need to take advantage of the benetits \a.va.füiote from applications built on the client/server model.

Client: An entity that requests service, It can be an application.ruıırıırıgön a personal computer or a dumb terminal requesting data for its screen,

Server: An entity that provides a service to a client. A server can be au application or object within an application

Network clients request information or a service from a server, and that server responds to the client by acting on that request and returning results. This approach to networking has proven to be a cost-effective way to share data between tens or hundreds of clients. Usually the client and server are two separate devices on a LAN. but client/server systems work equally well on long-distance WANs (irıeluding the Internet).

REFERENCES

[l]Smith Patrick, Client/server computing, Carmel, Ind, SAMS, cl992 .

[2]0rfali Robert, Dan Harkey. Client/server prograllllI1.ing with OS/2 2.0, Van Nostrand Reinhold.New York, c 1992.

[3] Dewire Dawna Travis, Client/server computing, McGravvt.B:i ll, New York, c1993.

[4] Baker Richard H, how to build client/server systems, McGravv'tHill, New York, cl 994.

[5] Inmon William H, Developing client/server applications,Q.Eff Pub. Group, Boston, cl 993.

[6] Boar Bernard H, Implementing client/server computing : 'a strat~gic perspective, 1987.