Ch 2. Basic Computer System Terms

2020-09-29

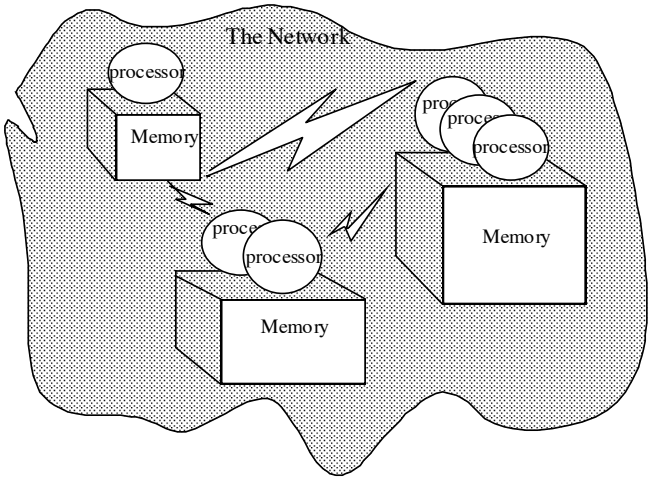
이종백

Basic Computer Terms

- There is widespread use of different terms to mean the same thing
  - field = column = attribute = ...
  - record = tuple = object = entity = ...
  - block = page = frame = slot = ...
  - file =data set = table = collection = relation = relationship =.. .
  - process = task = thread = actor = ...
- This chapter reviews those terms and defines them
- The standard units

Magnitude	Name	Abbreviation	Unit	Abbreviation
$10^{18} \approx 2^{60}$	exa	e, E	bit	b
$10^{15} \approx 2^{50}$	peta	p, P	byte (8 bits)	B
$10^{12} \approx 2^{40}$	tera	t, T	bits per second	bps
$10^{9} \approx 2^{30}$	giga, billion	g, b, G, B	bytes per second	Bps
$10^{6} \approx 2^{20}$	mega	m, M	instructions per second	ips
$10^{3} \approx 2^{10}$	kilo	k, K	transactions per second	tps
$10^{0} \approx 2^{0}$				
10^{-3}	milli	m		
10^{-6}	micro	μ		
10^{-9}	nano	n		
10^{-12}	pico	p		
10^{-15}	femto	f		

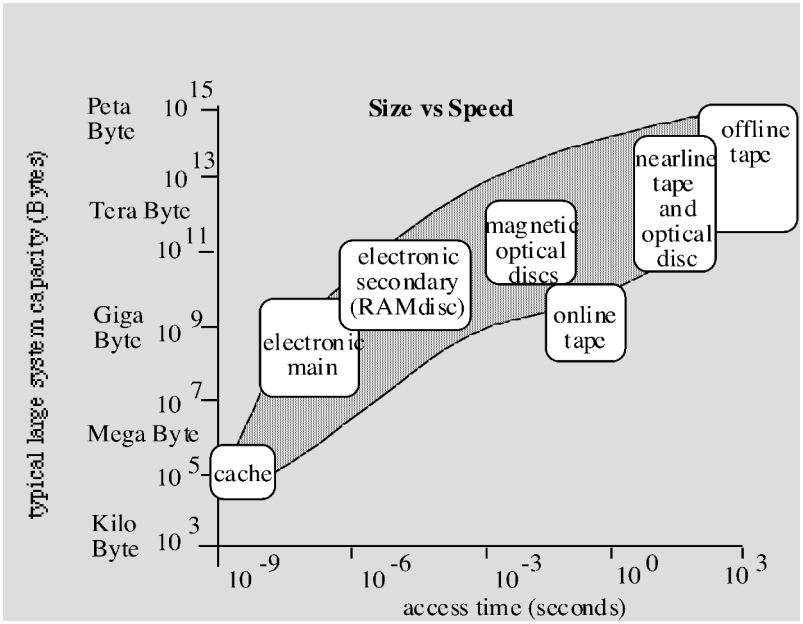
Basic Hardware

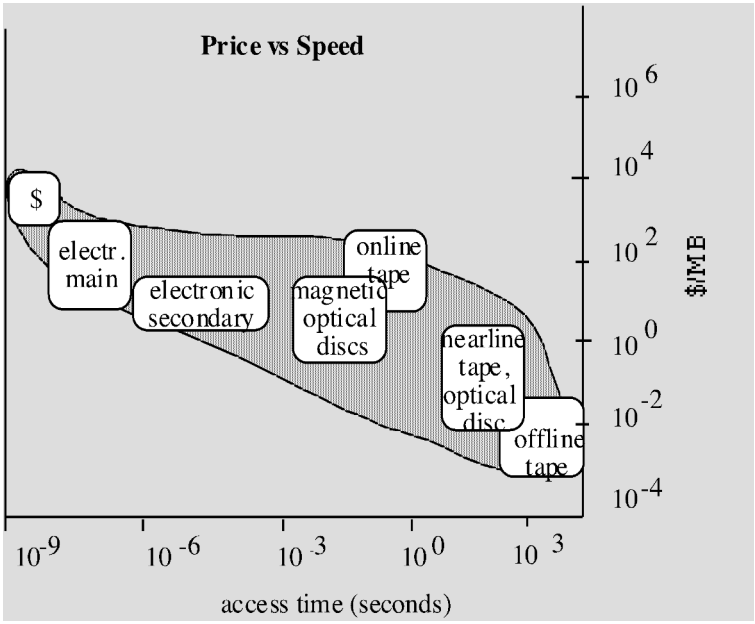


- Hardware consists of three types of modules
  - Processors
  - Memory
  - Communications (switches or wires)
- **Processors** execute instructions from a program, read and write **memory**, and send data via **communication** lines.
- Computation power, electronic memories sizes, and communication bandwidth are increasing
  - Distributed processing
  - Client-server
  - Clusters
- Transaction techniques are an enabling technology for distributed systems.

Memories

- Memories store data and allow processors to read and write the data
- Memory performance is measured by its access time:
  - memory access time = latency + (transfer size / transfer rate)
- Memory price-performance is measured in one of two ways:
  - **Cost/byte**: The cost of storing a byte of data in that media
  - **Cost/access**: The cost of reading a block of data from that media.





Electronic Memory

- Byte-addressable electronic memory : main memory
- Block-addressable bulk electronic memory : secondary storage
- The processor cannot directly manipulate secondary memory
  - A secondary memory block must be copied to main memory
  - The changed result is then copied back to secondary memory
- Moore's Law

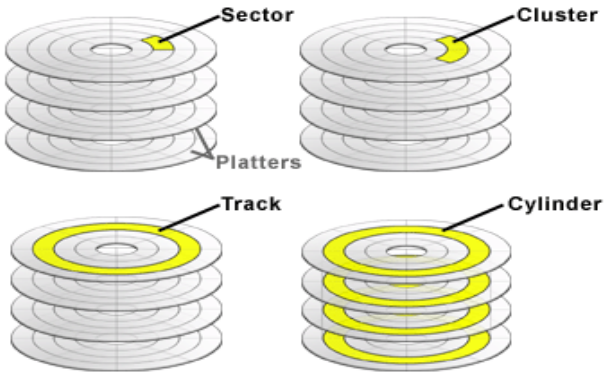
$$\text{MemoryChipCapacity(year)} = 4^{\frac{(\text{year} - 1970)}{3}} \text{ Kb/chip for year in [1970...2000]}$$

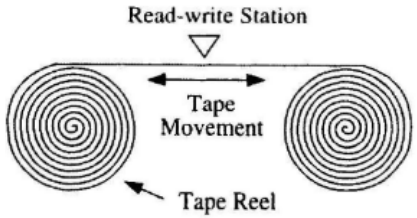
- Cache memory
  - Memories are getting bigger, processors are getting faster
  - Memories are not getting much faster
  - A processor spend most of its time waiting for instruction and data from memory

Magnetic Memory

- Two types of magnetic storage media: **Disk** and **Tape**
- **Disk**
  - Three-dimensional address space: (cylinder number, track number, sector number)
  - **Disk Access Time = Seek_Time + Rotational_Latency + (Transfer_Size/ Transfer_Rate)**

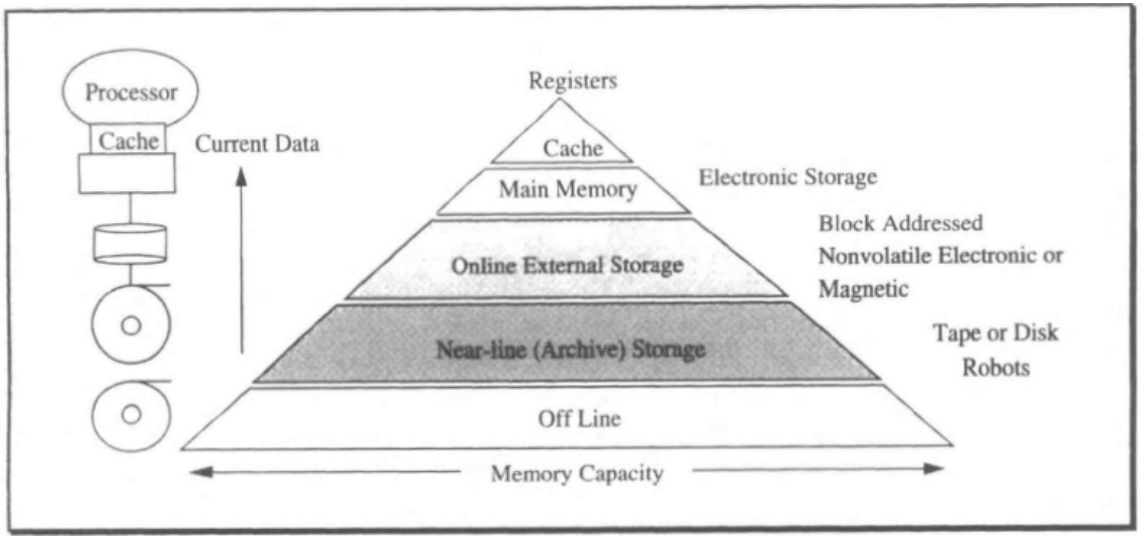
- Access patterns
  - Compare the times required for two access patterns to 1MB stored in 1000 blocks on disk (seek : 10ms, rotation : 5ms, 1KB transfer : 0.1ms)
  - **Sequential access**: 115 ms
  - **Random access** : 15.1 seconds





Memory Hierarchies

- The hierarchy uses small, fast, expensive cache memories to cache some data present in larger, slower, cheaper memories
- If hit ratios are good, the overall memory speed approximates the speed of the cache
- At any level of the memory hierarchy, the hit ratio is defined as:
  - **hit ratio = references satisfied by cache / all references to cache**
- Unless cache hit rates are very high (say, .99), the cache memory has approximately the same access time as the secondary memory.



The Five-Minute Rule

- Frequently accessed data should be in main memory. while it is cheaper to store infrequently accessed data on disk
- **The Five-Minute Rule**
  - Keep a data item in electronic memory if its access frequency is five minutes or higher; otherwise keep it in magnetic memory
- Example
  - A disk deliver 15 random accesses per second and is priced at 15K$ (1K$/a/s)
  - The extra CPU and channel cost for supporting disk is 1K$/a/s
  - So, one disk access per second costs about 2K$/a/s
  - 1MB memory costs 5K$ -> 1KB costs 5$
  - 1KB data record main-memory resident saves 1a/s, so 2K$ worth of disk access at a cost of 5$
  - If it saves 0.1a/s then it saves about 200$
  - Break-even point : 2000/5 ~ 400 second ~ 5miniutes

Communication Hardware

Table 2.6: The definition of the four kinds of networks by their diameters. These diameters imply certain latencies (based on the speed of light). In 1990, Ethernet (at 10 Mbps) was the dominant LAN.[3] Metropolitan networks typically are based on 1 Mbps public lines. Such lines are too expensive for transcontinental links at present; most long-distance lines are therefore 50 Kbps or less. As the text notes, things are changing (see Table 2.7).

Type of Network	Diameter	Latency	Bandwidth	Send 1 KB
Cluster	100 m	.5 µs	1 Gbps	10 µs
LAN (local area network)	1 km	5. µs	10 Mbps	1 ms
MAN (metro area network)	100 km	.5 ms	1 Mbps	10 ms
WAN (wide area network)	10,000 km	50. ms	50 Kbps	210 ms

Table 2.7: Point-to-point bandwidth likely to be common among computers by the year 2000.

Type of Network	Diameter	Latency	Bandwidth	Send 1 KB
Cluster	100 m	.5 µs	1 Gbps	5 µs
LAN (local area network)	1 km	5. µs	1 Gbps	10 µs
MAN (metro area network)	100 km	.5 ms	100 Mbps	.6 ms
WAN (wide area network)	10,000 km	50. ms	100 Mbps	50 ms

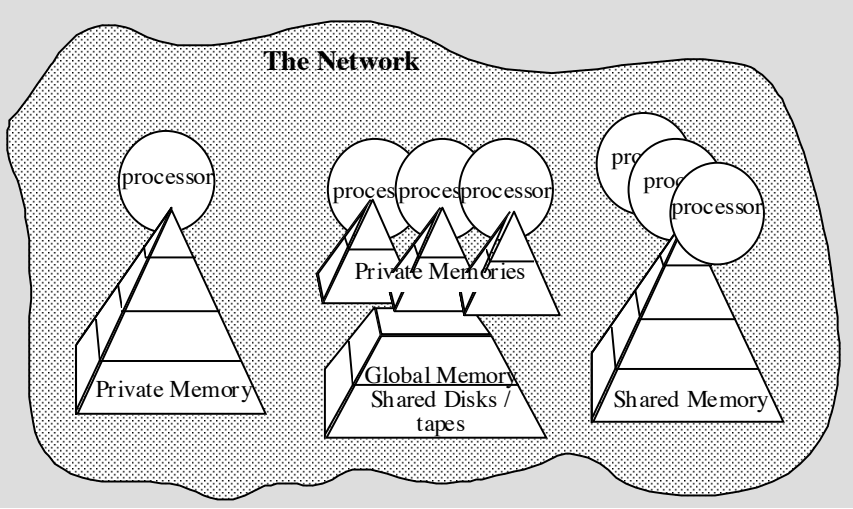
The Rules of Exponential Growth

- **Electronic memory(Moore's Law):**
- MemoryChipCapacity(year) = $4^{((year-1970)/3)}$ Kb/chip for year in [1970...2000]

- **Magnetic memory(Hoagland's Law):**
- MagneticAreaDensity(year) = $10^{((year-1970)/10)}$ Mb/inch^2 for year [1970...2000]
- **Processors(Joy's Law):**
- SunMips(year) = $2^{(year-1984)}$ MIPS for year in [1984...2000]

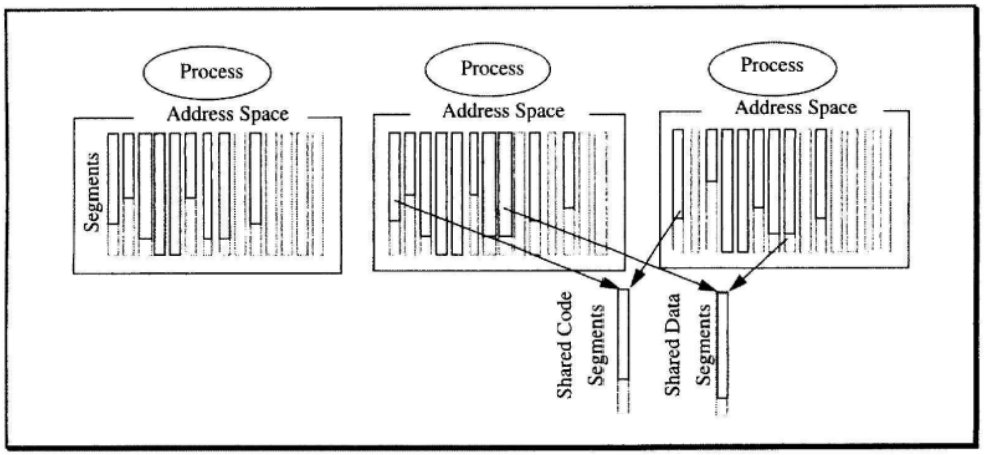
Hardware Architectures (Processor-Memory Architecture)

- **Shared nothing**
  - Each memory is dedicated to a single processor
- **Shared global**
  - Each processor has some private memory not accessible to other processors. There is, however, a pool of global memory
- **Shared memory**
  - Each processor has transparent access to all memory.



Basic Software – Address Spaces

- A *process address space* is an array of bytes
- Each process executes instructions from its address space and can read and write bytes of the address space using processor instructions
- A process executes in an address space—a paged, segmented array of bytes. Some segments may be shared with other address spaces
- Address spaces are partitioned into segments, and segments are partitioned into pages.
  - Virtual memory addresses are actually (segment, page, byte)

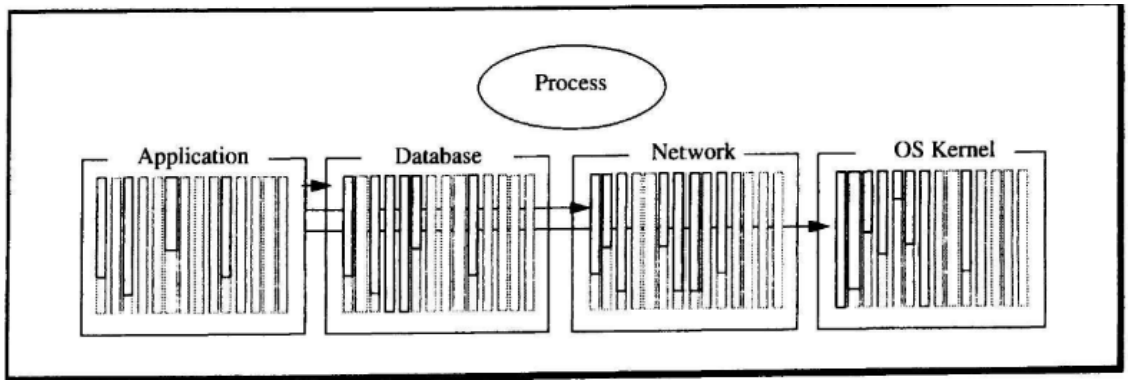


Basic Software – Processes

- A *process* is a virtual processor
- It has an address space that contains the program the process is executing and the memory the process reads and writes
- Processes provides
  - An ability to execute programs in parallel
  - A protection entity
  - A way of structuring computations into independent execution streams
- Each process executes on behalf of some user, or *authority,* and with some *priority*

Basic Software – Protection Domains

- **Protection domain:** encapsulated environment to protect subsystems from faults in application
- There are two ways to provide protection :
  - Process = protection domain
  - Address space = protection domain



Basic Software - Threads

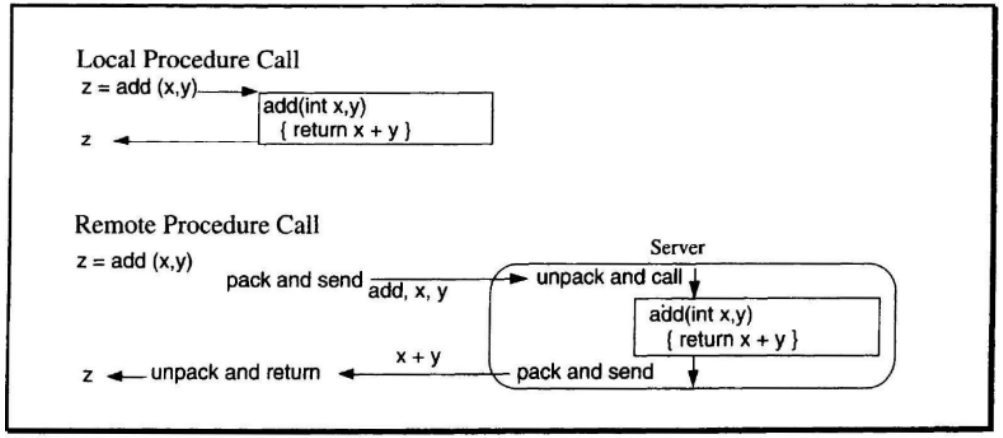
- There is a need for multiple processes per address space:
  - Producer-consumer while scan
  - Asynchronous buffer flushing
  - File read-ahead …
- Thread multiplex one operating system process among many virtual processes
- Thread is inexpensive to create and dispatch
- The term thread is used in this book to connote these light-weight processes

Basic Software – Messages and Sessions

- There are two styles of communication among processes:
- **Datagram**
  - Sender looks up the recipient‘s process name and its address
  - Sender constructs an envelope containing the name of sender and reciever, and the message
  - The resulting envelope is given to the network to deliever
- **Session**
  - Communication between two processes via a pre-established bidirectional message pipe
  - Both parties can send and receive messages via this session
  - Benefits
    - **Shared state**: A session represents shared state between the client and the server
    - **Authorization**: Once the session key is established, it becomes shared state
    - **Error correction**: Session can detect lost messages and endpoint fails
    - **Performance**: Client, server information is cached.

Generic System Issues (Client and Servers)

- Two structures for multiple interacting processes:
- **Peer-to-peer**
  - The two processes are independent peers, each executing its computation and occasionally exchanging data with the other.
- **Client-server**
  - The two processes interact via request-reply exchanges
  - ***Client*** makes a request to a server
  - ***Server*** performs the request and replies to the client
  - The transparent invocation of server procedures is called **remote procedure call (RPC)**



Generic System Issues (Naming)

- How a client denotes a server it wants to invoke
- Typical naming schemes distinguish between an object's name, its address, and its location
- An object can have several names. Some of these names may be synonyms, called aliases
- Name server
  - Names are grouped into a hierarchy called the name space
  - *Name servers* store parts of the name space local to their neighborhood; in addition, they store a directory of more global name servers.

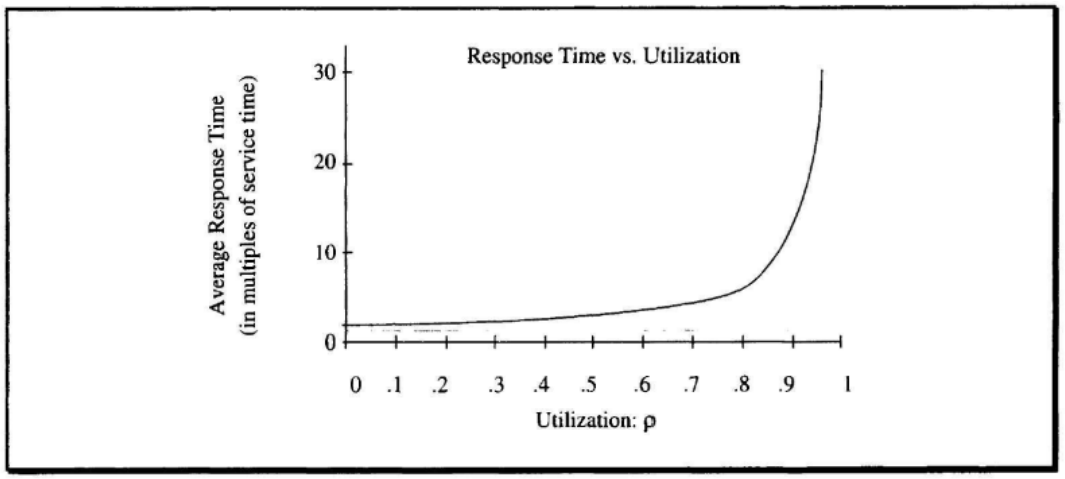
Generic System Issues (Authentication and authorization)

- Each process wants to authenticate the other
- Authentication techniques
  - Passwords
  - Challenge-response
  - Public key system

- Client allowed to perform a operation by authorization

Generic System Issues (Scheduling and Performance)

- Server's response time : Wait time + Service time
- As the utilization of a server or resource increases, the wait time rises dramatically



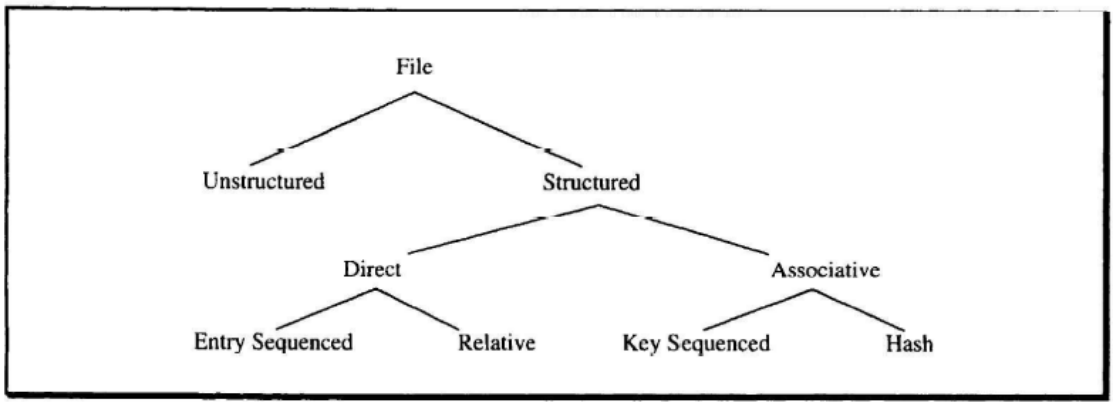
Files

- File Operations
  - Create()
  - Delete()
  - Memory-mapped files
    - Map_file()
    - unmap_file()
  - Explicit file actions
    - Open()
    - Close()
    - Read()
    - Write()

File Organizations

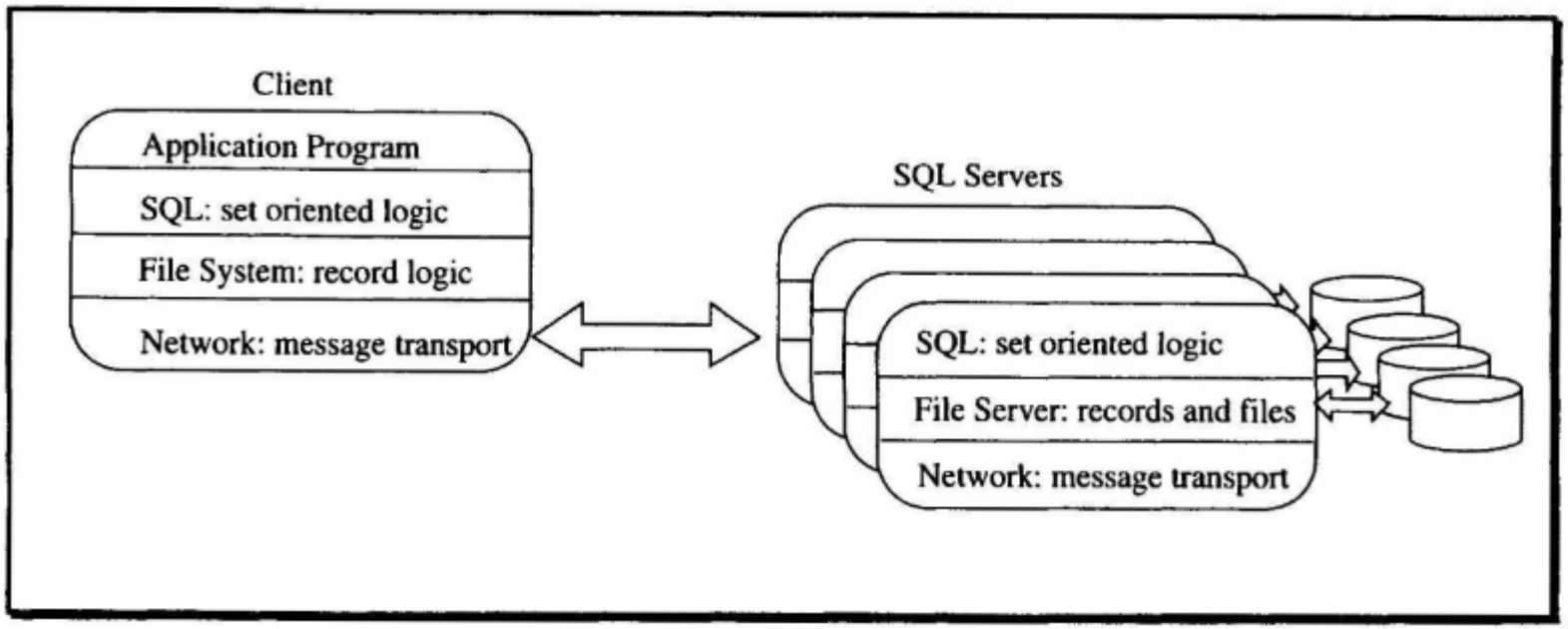
- The file system keeps a descriptor for each file
  - File name, the authid of the file creator, and an access control list …
- **Unstructured file**: binary image, simple byte stream…
- **Structured file**: a collection of records with similar structure
  - Chapter 13~15 describes structured file details

- Distributed Files
  - File may be distributed among servers
  - The files can be *partitioned* or *replicated*



SQL

- SQL offers set-oriented, non-procedural data definition, manipulation, and control



Software Performance

- Typical software costs for storing and fetching records from a database

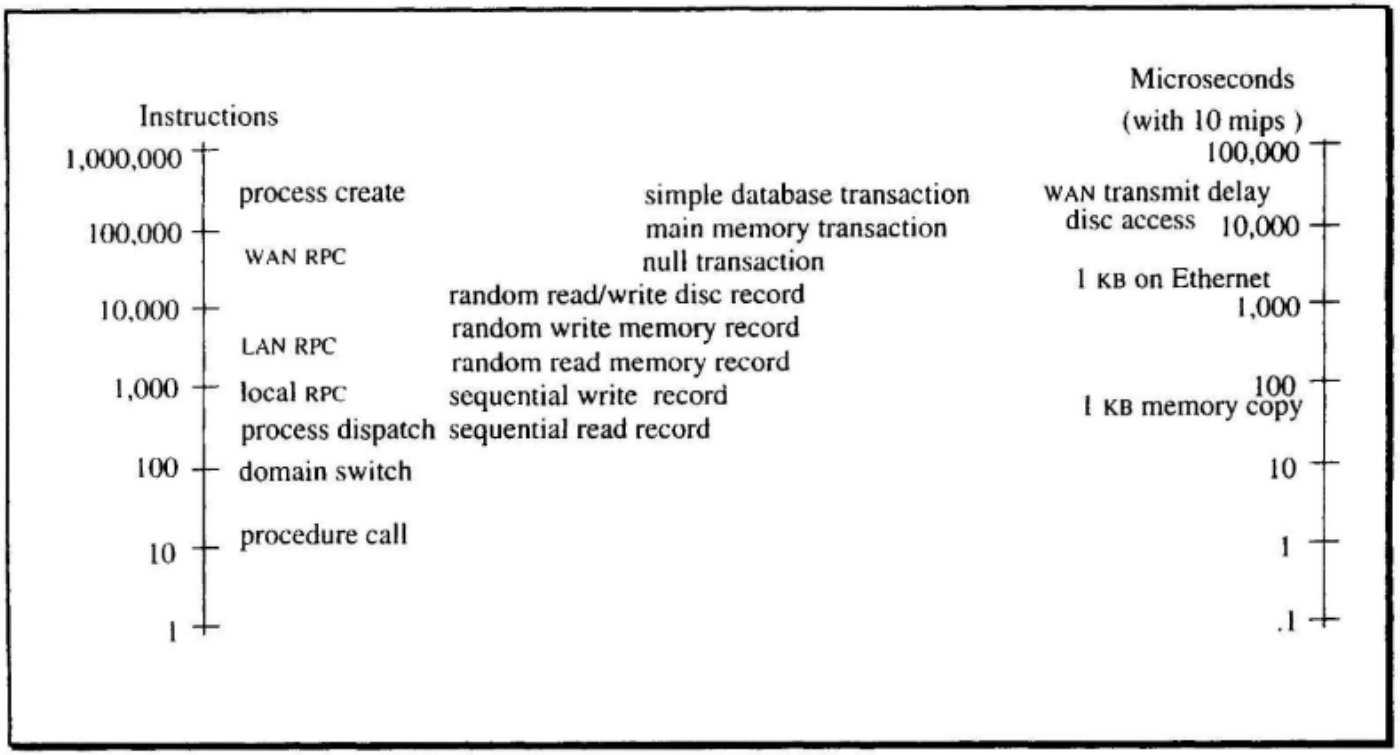


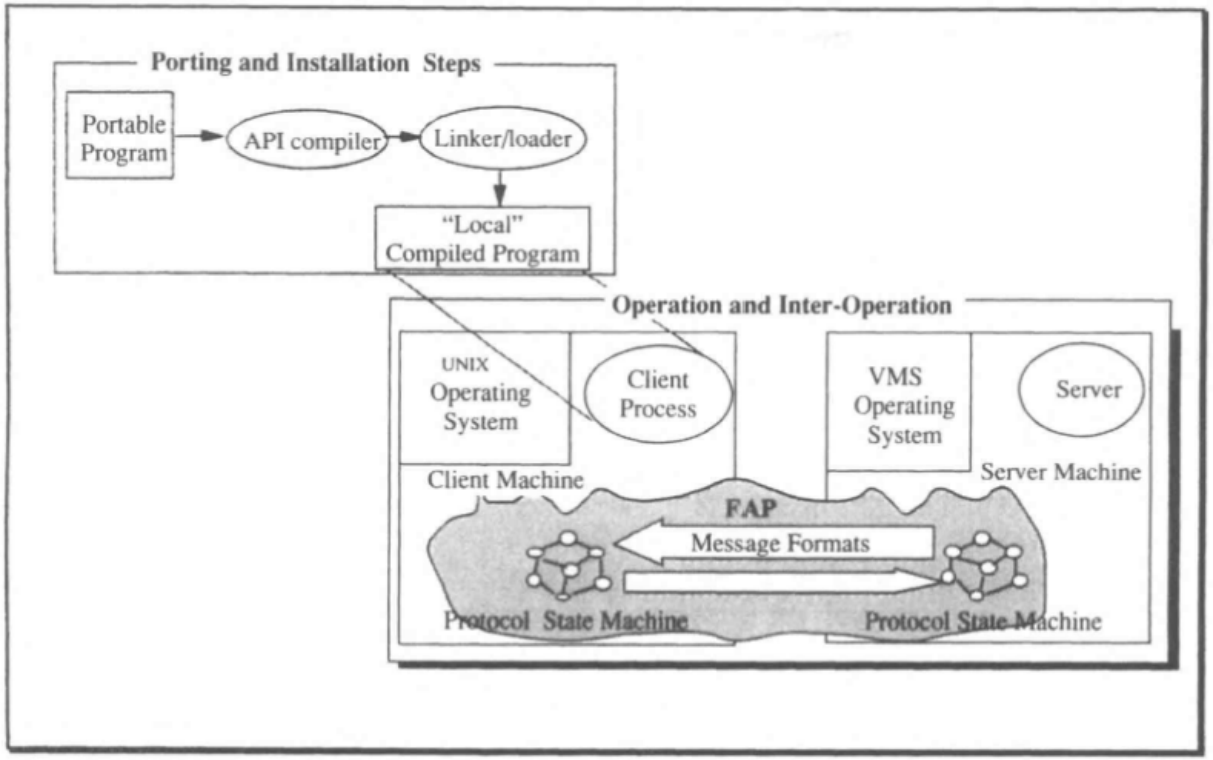
Figure 2.16: Typical performance of commercial systems. The chart at the left gives the approximate instruction cost of major operating system, transaction processing system, and database system operations. The chart at the right is in terms of microseconds. It maps the instructions on the left to a 10-mips machine and shows the latencies of disks, LANs, and WANs.

Transaction Processing Standards

- **Portability**
  - By writing programs in standard languages, the programs can be run on (ported to) many different computer systems
- **Interoperability**
  - By defining and implementing standard ways to exchange data, computers from different manufacturers can exchange data without any special conversion or interfacing code

- **API**
  - Application Programming Interface
  - API is programming interface
- **FAP**
  - Formats and Protocols
  - FAP is Communication protocol

APIs and FAPs



Summary

- Basic hardware and software concepts from the perspective of client-server distributed computing
- Hardware trends
  - Processors
  - Memories
  - Communications
- Software notions
  - Address space
  - Process
  - Message and Session
- Files
- TP standards