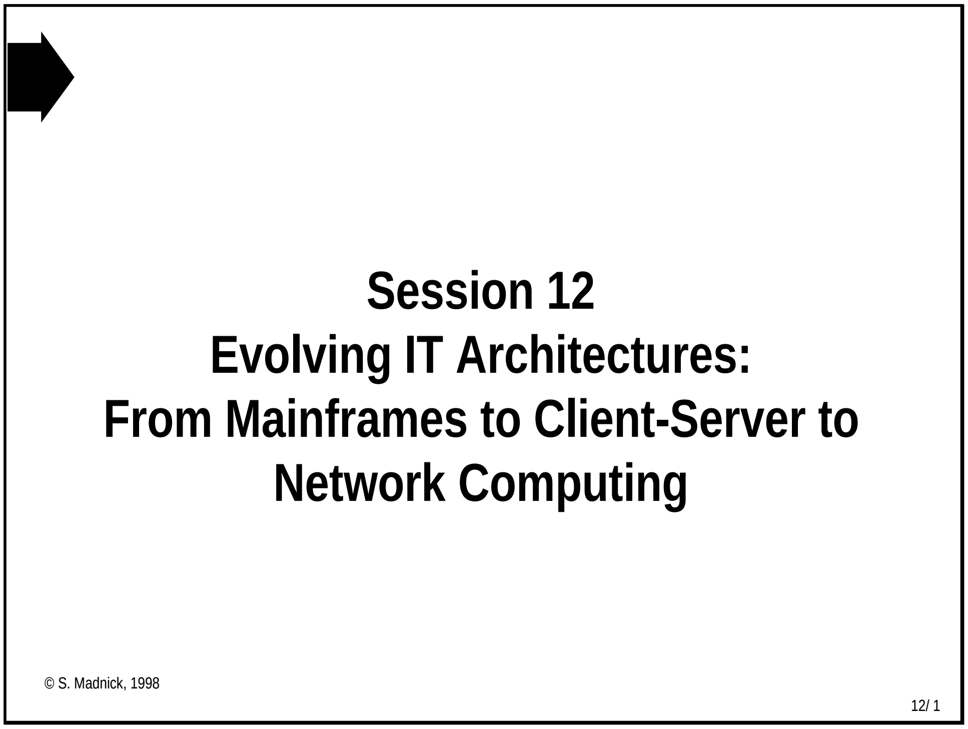
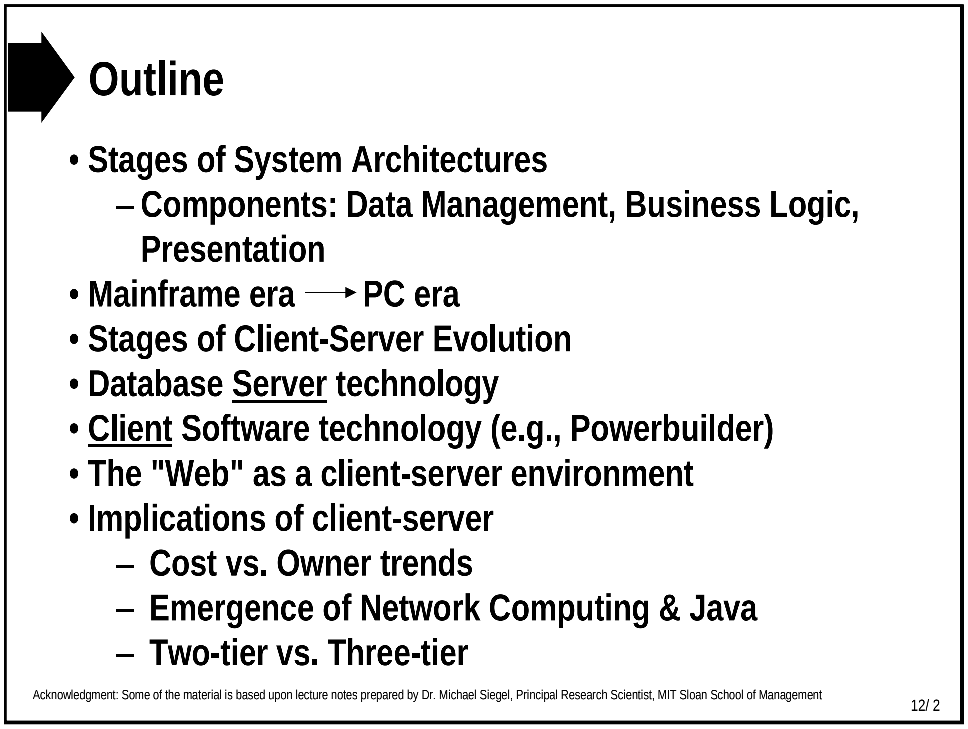


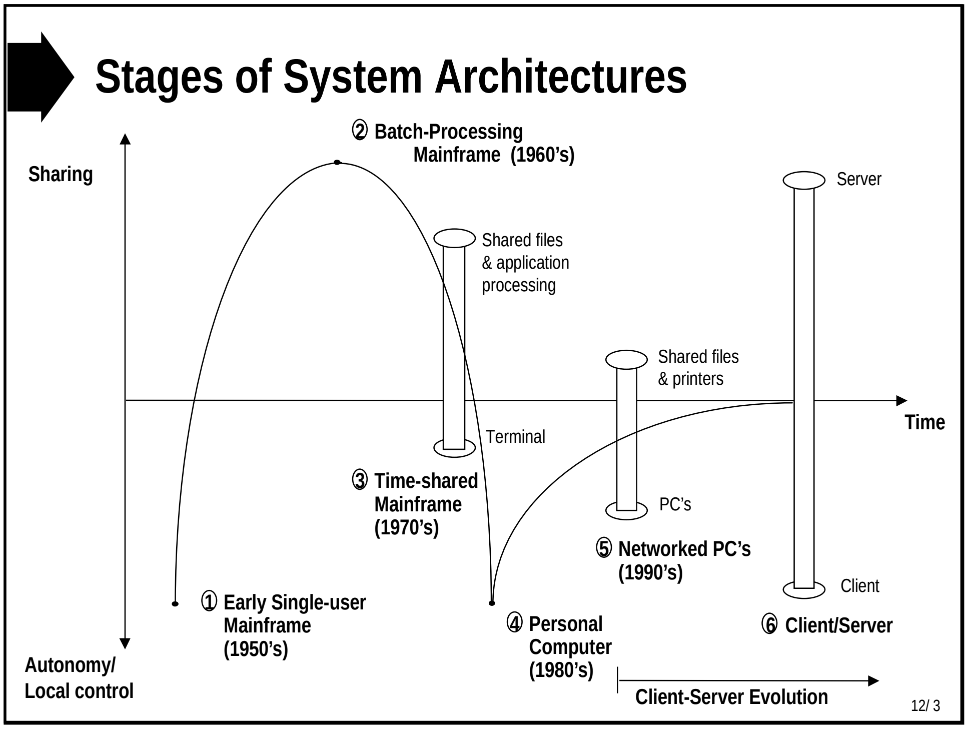
Session 12
Evolving IT Architectures:
From Mainframes to Client-Server to Network Computing



Outline

- Stages of System Architectures
  - Components: Data Management, Business Logic, Presentation
- Mainframe era → PC era
- Stages of Client-Server Evolution
- Database <u>Server</u> technology
- <u>Client</u> Software technology (e.g., Powerbuilder)
- The "Web" as a client-server environment
- Implications of client-server
  - Cost vs. Owner trends
  - Emergence of Network Computing & Java
  - Two-tier vs. Three-tier

Acknowledgment: Some of the material is based upon lecture notes prepared by Dr. Michael Siegel, Principal Research Scientist, MIT Sloan School of Management



Stages of System Architectures

Sharing

Autonomy/
Local control

Time

① Early Single-user
Mainframe
(1950's)

② Batch-Processing
Mainframe (1960's)

③ Time-shared
Mainframe
(1970's)

Shared files
& application
processing

Terminal

④ Personal
Computer
(1980's)

⑤ Networked PC's
(1990's)

Shared files
& printers

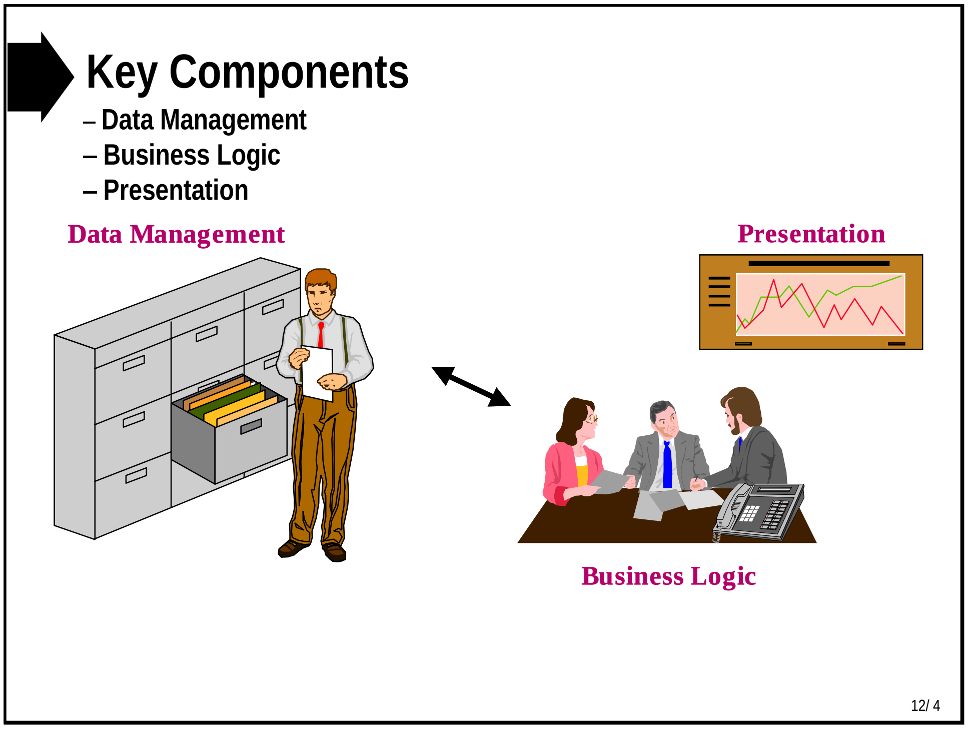
PC's

⑥ Client/Server

Server

Client

Client-Server Evolution



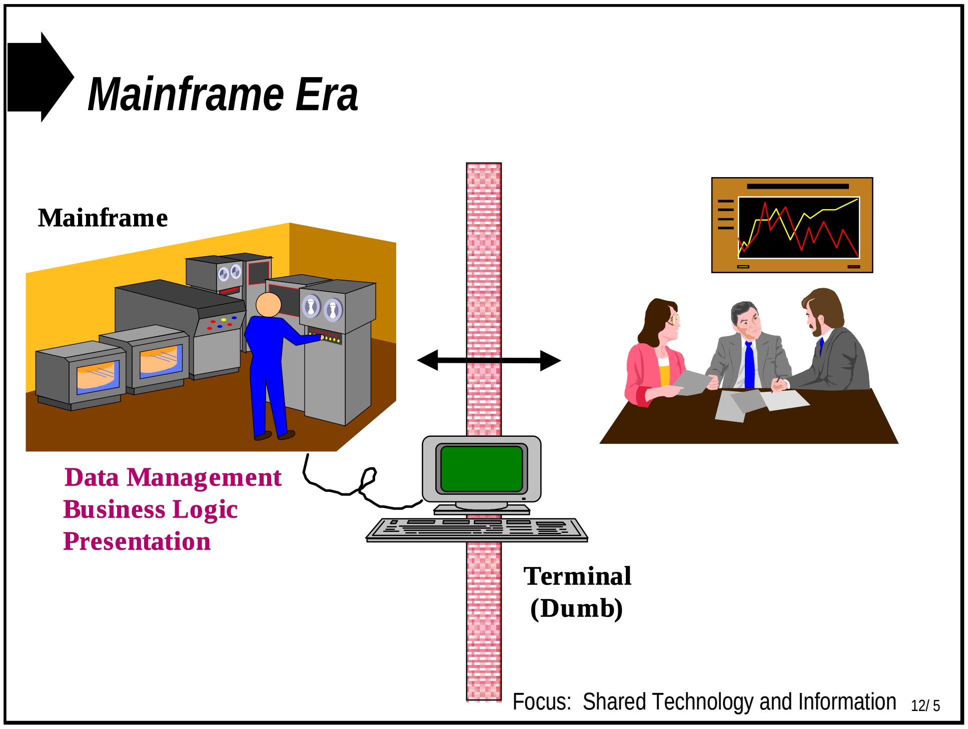
Key Components

– Data Management
– Business Logic
– Presentation

Data Management

Presentation

Business Logic



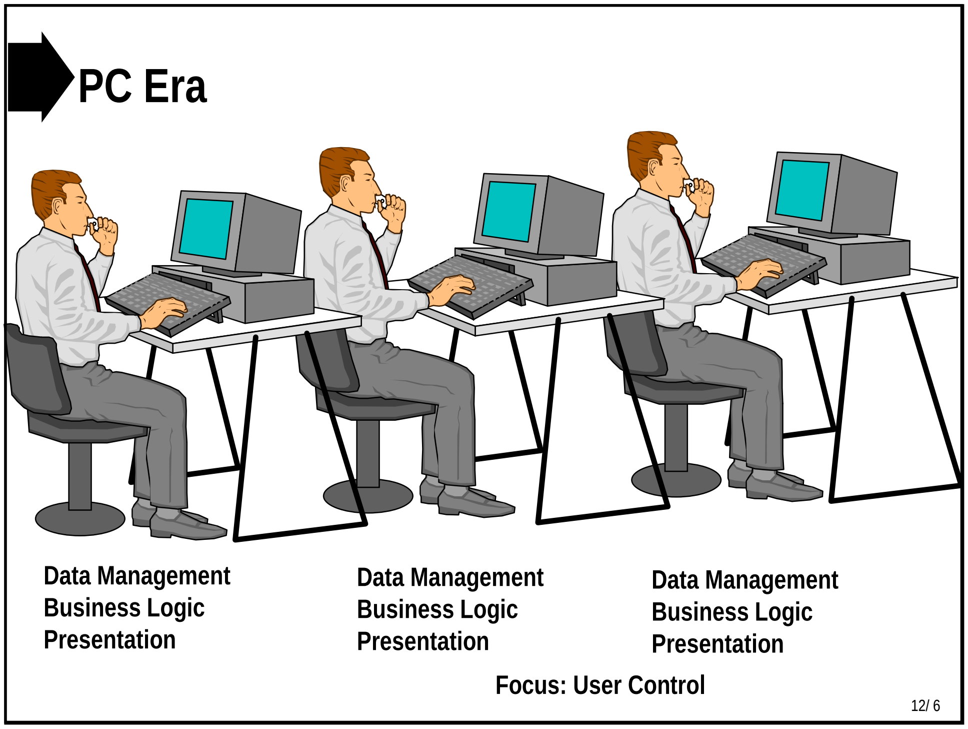
Mainframe Era

Mainframe

Data Management
Business Logic
Presentation

Terminal
(Dumb)

Focus: Shared Technology and Information



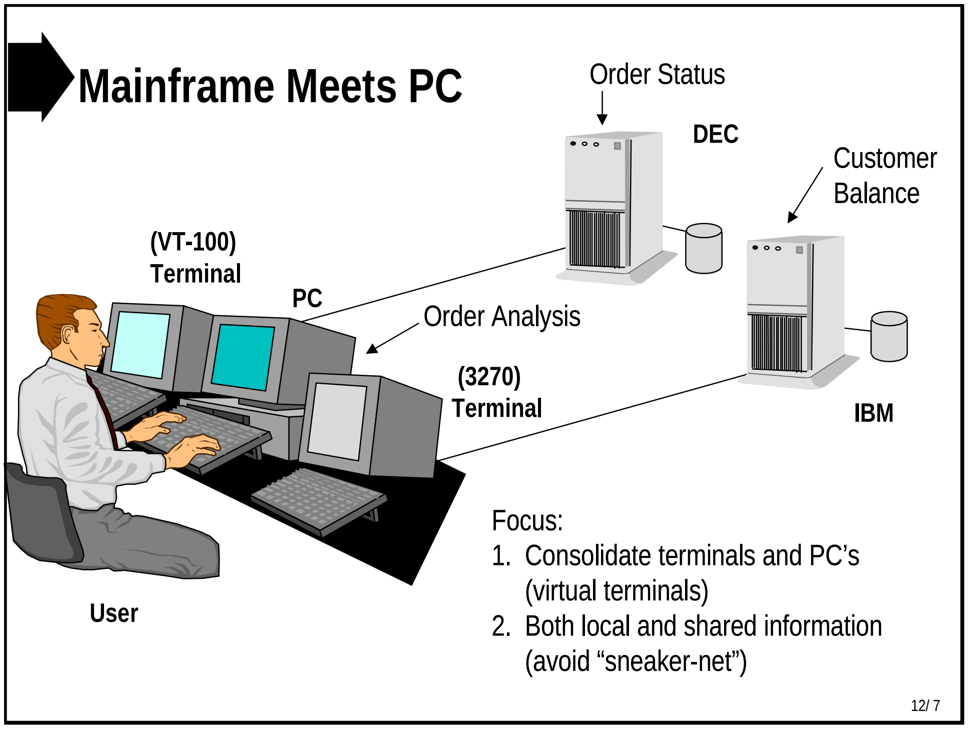
PC Era

Data Management
Business Logic
Presentation

Data Management
Business Logic
Presentation

Data Management
Business Logic
Presentation

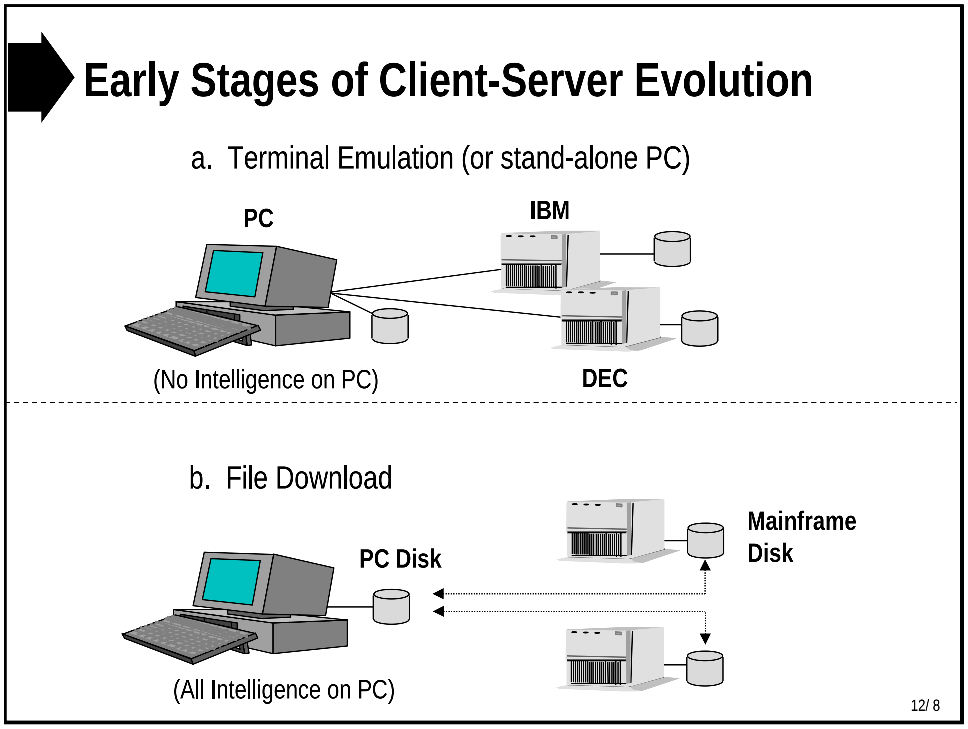
Focus: User Control



Mainframe Meets PC

Focus:

1. Consolidate terminals and PC’s (virtual terminals)
2. Both local and shared information (avoid “sneaker-net”)



Early Stages of Client-Server Evolution

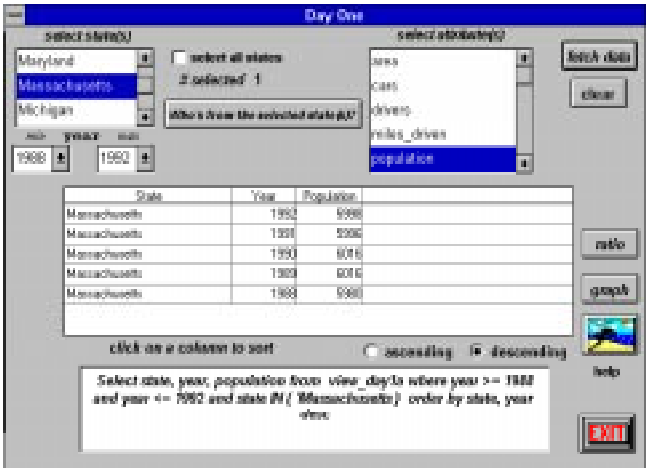
a. Terminal Emulation (or stand-alone PC)

(No Intelligence on PC)

b. File Download

(All Intelligence on PC)

Example: GUI Interface



Day One

select state(s)

Maryland
Massachusetts
Michigan

□ select all states

selected: 1

Who's from the selected state(s)?

min year max

1988 ± 1992 ±

select attribute(s)

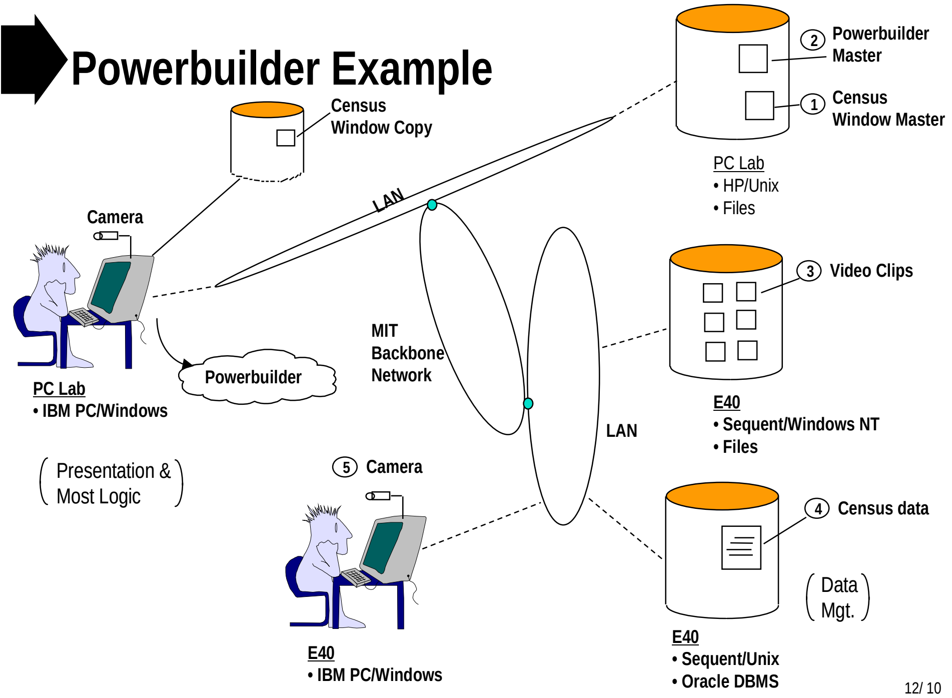
area
cars
drivers
miles_driven
population

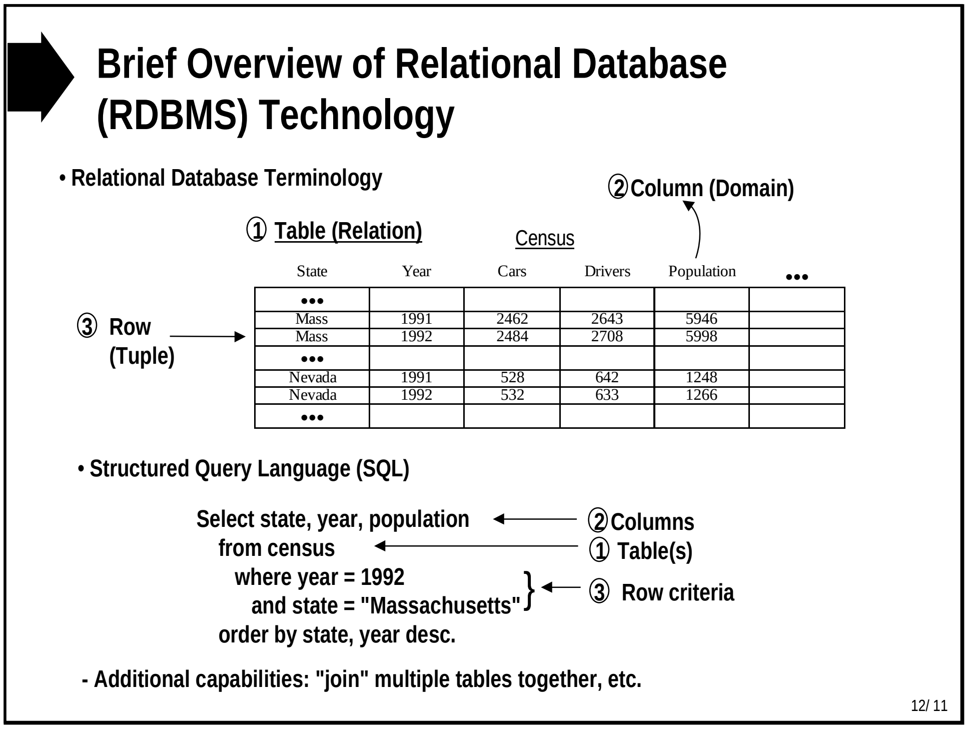
State	Year	Population
Massachusetts	1992	5998
Massachusetts	1991	5996
Massachusetts	1990	6016
Massachusetts	1989	6016
Massachusetts	1988	5980

click on a column to sort ○ ascending ● descending

Select state, year, population from view_day1a where year >= 1988 and year <= 1992 and state IN ('Massachusetts') order by state, year desc

Powerbuilder Example





Brief Overview of Relational Database (RDBMS) Technology

- Relational Database Terminology

① Table (Relation) — Census

② Column (Domain)

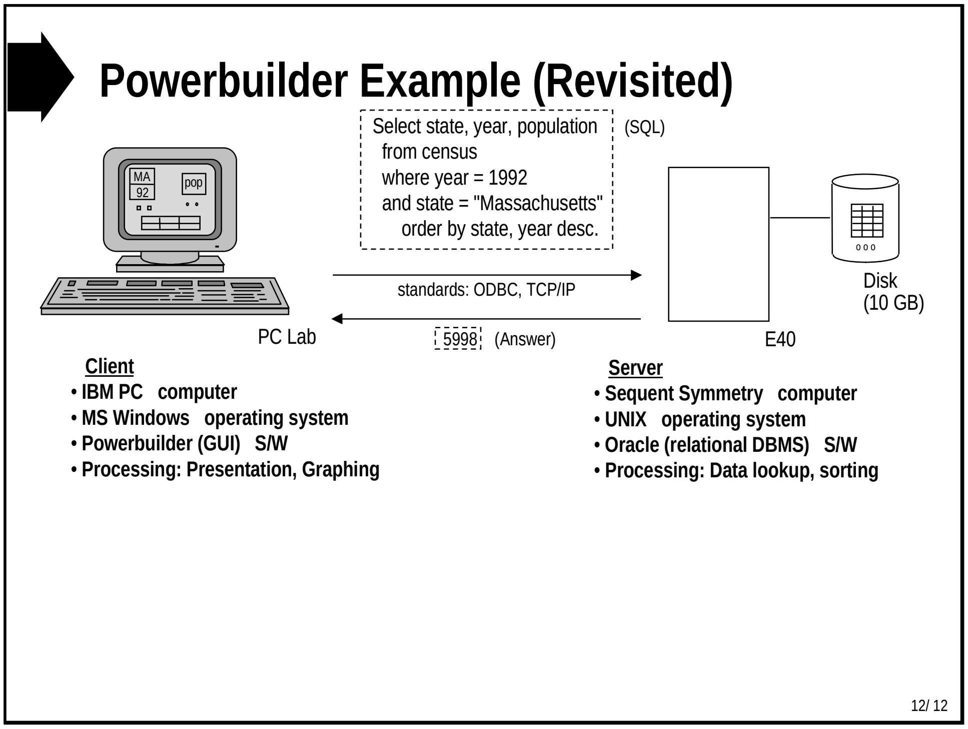
③ Row (Tuple)

State	Year	Cars	Drivers	Population	•••
•••					
Mass	1991	2462	2643	5946	
Mass	1992	2484	2708	5998	
•••					
Nevada	1991	528	642	1248	
Nevada	1992	532	633	1266	
•••					

- Structured Query Language (SQL)

```
Select state, year, population       ← ②Columns
  from census                        ← ① Table(s)
    where year = 1992              }
      and state = "Massachusetts"  } ← ③ Row criteria
  order by state, year desc.
```

- Additional capabilities: "join" multiple tables together, etc.



Powerbuilder Example (Revisited)

```
Select state, year, population   (SQL)
  from census
  where year = 1992
  and state = "Massachusetts"
    order by state, year desc.
```

standards: ODBC, TCP/IP

5998 (Answer)

PC Lab

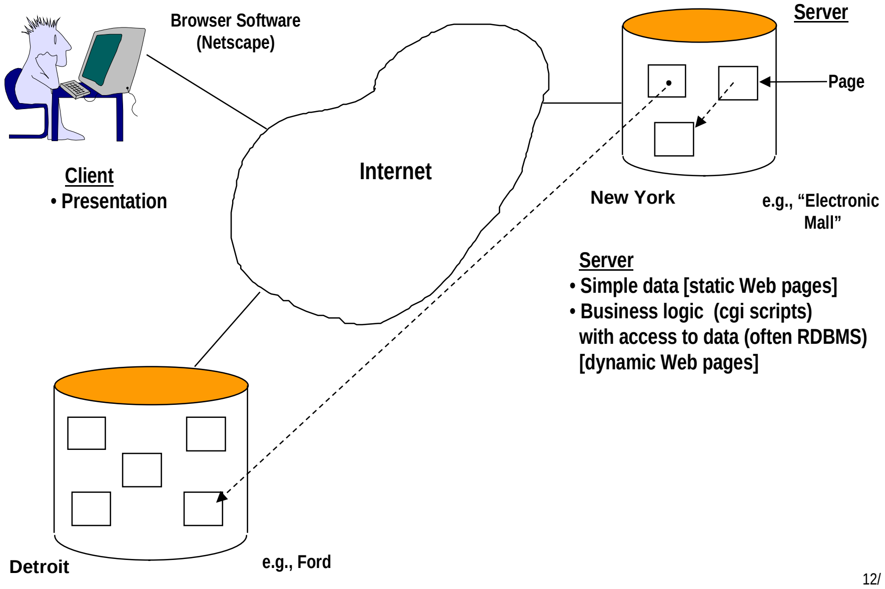
E40

Disk (10 GB)

Client
- IBM PC computer
- MS Windows operating system
- Powerbuilder (GUI) S/W
- Processing: Presentation, Graphing

Server
- Sequent Symmetry computer
- UNIX operating system
- Oracle (relational DBMS) S/W
- Processing: Data lookup, sorting

The “Web” as a "2-tier" Client-Server System



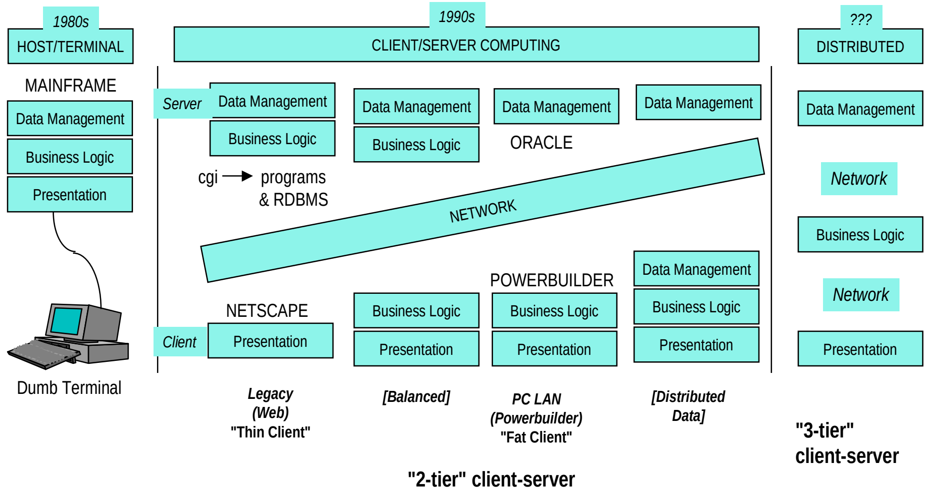
Client
- Presentation

Server
- Simple data [static Web pages]
- Business logic (cgi scripts) with access to data (often RDBMS) [dynamic Web pages]

Definition

Client-Server forms a subset of distributed computing systems where data management, application (e.g., business) logic, and presentation functions are separated by predefined interfaces that enable them to be distributed and operate, in real-time, as if they were a single program on a single computer...

Client-Server Computing As a Technology Shift


1980s

HOST/TERMINAL

MAINFRAME

Data Management

Business Logic

Presentation

Dumb Terminal

1990s

CLIENT/SERVER COMPUTING

Server

Client

Data Management

Business Logic

cgi → programs & RDBMS

NETSCAPE

Presentation

Legacy (Web) **"Thin Client"**

Data Management

Business Logic

Business Logic

Presentation

[Balanced]

Data Management

ORACLE

POWERBUILDER

Business Logic

Presentation

PC LAN (Powerbuilder) **"Fat Client"**

Data Management

Data Management

Business Logic

Presentation

[Distributed Data]

NETWORK

???

DISTRIBUTED

Data Management

Network

Business Logic

Network

Presentation

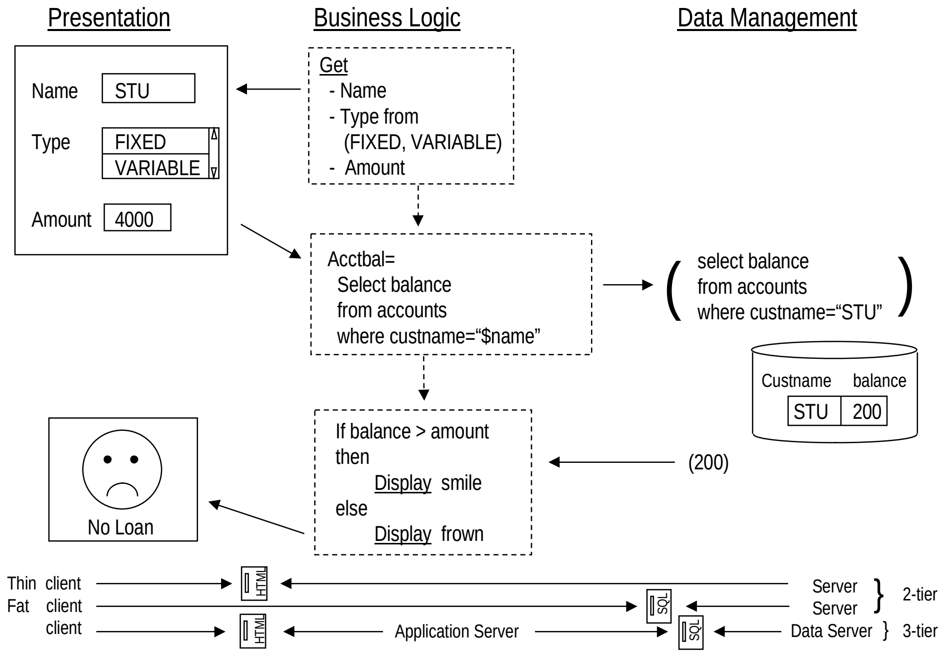
"3-tier" client-server

"2-tier" client-server

Source: Gartner Group

Question: Where does Java fit into this?

SAMPLE APPLICATION ILLUSTRATING SEPARATION OF COMPONENTS


Presentation

Business Logic

Data Management

Name STU

Type FIXED VARIABLE

Amount 4000

Get
- Name
- Type from (FIXED, VARIABLE)
- Amount

Acctbal=
Select balance
from accounts
where custname="$name"

(select balance
from accounts
where custname="STU")

Custname	balance
STU	200

If balance > amount
then
Display smile
else
Display frown

(200)

No Loan

Thin client → HTML ← Server } 2-tier

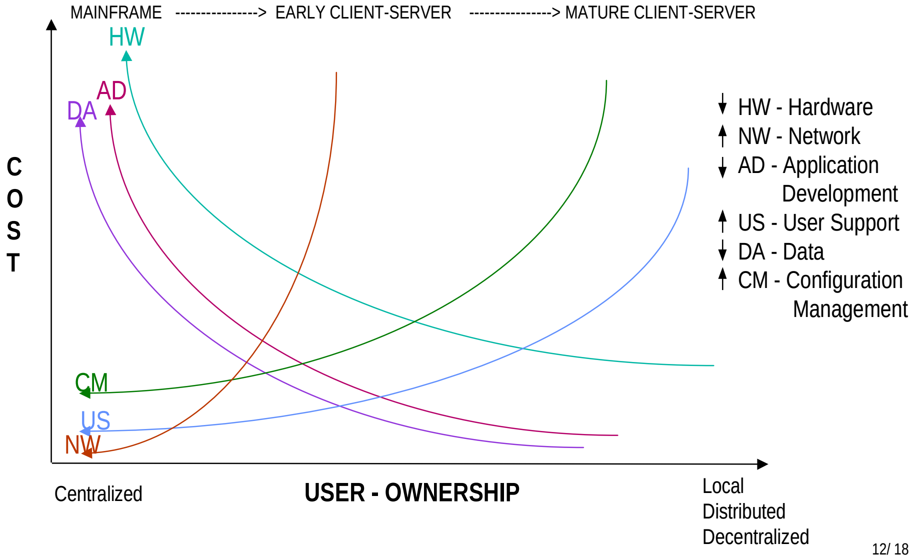
Fat client → SQL ← Server

client → HTML ← Application Server → SQL ← Data Server } 3-tier

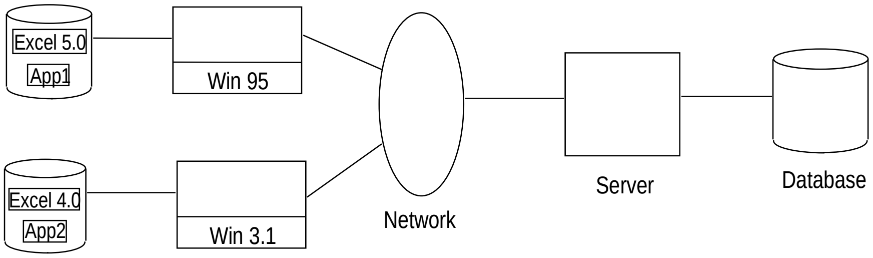
IMPLICATIONS OF CLIENT-SERVER

- **New Architectures**
  - Distributed Computing
  - Distributed Databases
- **Increased Heterogeneity**
- **Increased Network and Connectivity Requirements**
- **Challenges “managing the transition”**
- **User-ownership vs. Central-ownership**
  - Hardware
  - Networks
  - Configuration Management
  - User-Support
  - Data
  - Application Development

COST VS OWNERSHIP TRENDS

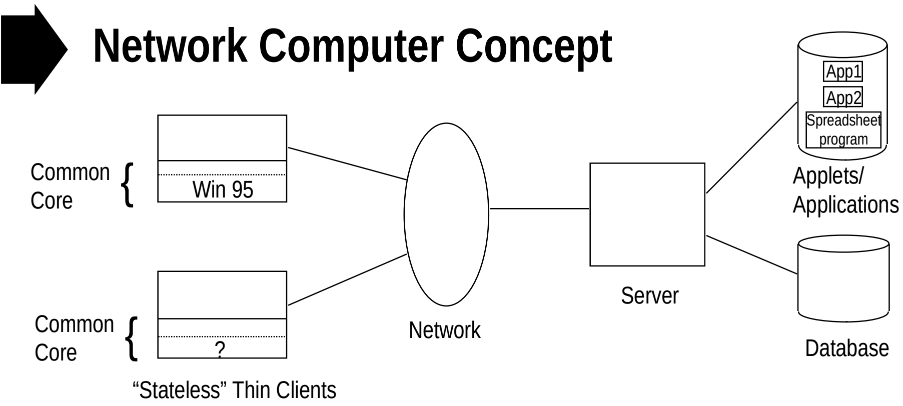


Traditional "Fat Client" Situation



- **Problems** ("total cost of ownership high")
  - – "Expensive" hardware (hard disk)
  - – Software investment (esp. if infrequent use)
  - – Updating software
  - – System management (software conflicts/incompatibilities)
  - – Proprietary software

Network Computer Concept

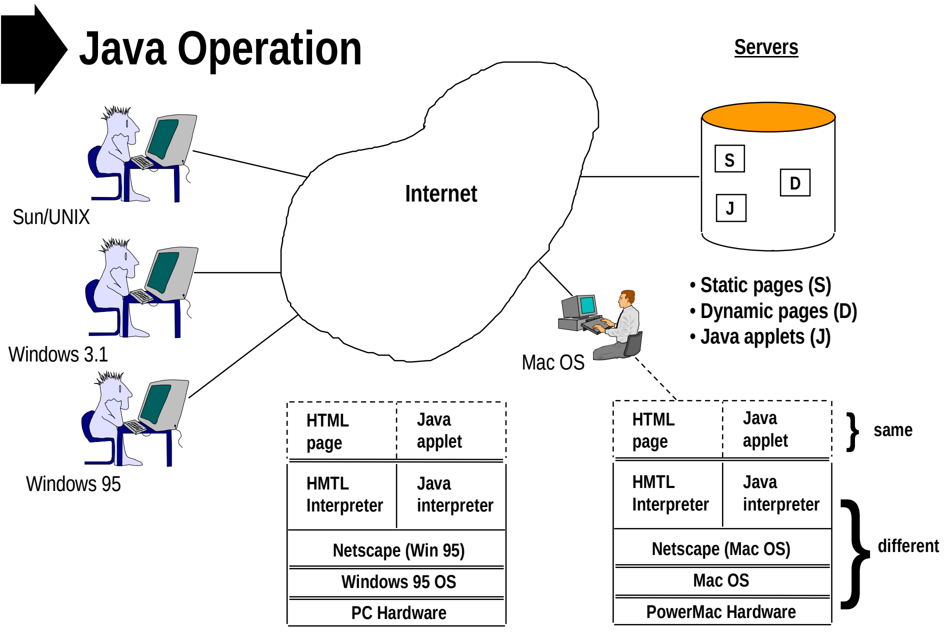


- **Solutions**
  - – Minimal hardware (usually no hard disk)
  - – Software downloaded as needed (possibly only portions)
  - – No persistent "state" / freshest software loaded each time
  - – Minimal client-side system management (all identical)
  - – Movement toward "open" standards

Java

- **Highly interactive**
  - – Traditional Web - application software runs on server
  - – Java applets dynamically downloaded and run on client (e.g., input data validation)
- **“Nice” programming language**
  - – Simpler than C/C++
  - – Object-oriented
- **Secure programming environment**
  - – Strong typing (array checking)
  - – Interpreter
- **Portable (“write once, run anywhere”)**
  - – Based on Java byte-code interpreter

Java Operation



- **Static pages (S)**
- **Dynamic pages (D)**
- **Java applets (J)**

HTML page	**Java applet**
HMTL Interpreter	**Java interpreter**
Netscape (Win 95)	
Windows 95 OS	
PC Hardware	

HTML page	**Java applet**
HMTL Interpreter	**Java interpreter**
Netscape (Mac OS)	
Mac OS	
PowerMac Hardware	

Client environments

Sample Java Applet

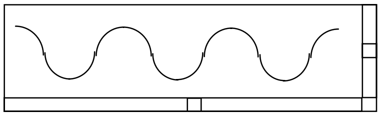
Sample Java applet that draws a sine wave

```
import java.awt.*;
import java.applet.*;

public class Wave extends Applet
{ int n = 1
  public void paint(Graphics g)

   { double y=0.0, oy=0.0;
      for(int x=0; x < size().width; oy=y, y=0, x++)
     {for (int j=0; j<n; j++)
       y += Math.sin((2*j+1)* x / 15.0)/(2*j+1);
       y = 0.47 * y + 0.5;
       if (x>0) g.drawLine(x, (int)(oy*size().height), x+1, (int) (y*size().height));
     }
   }
 public boolean mouseDown (java.awt.Event evt, int x, int y)
 { n = n < 15 ? n+1 : 1; repaint (); return true; }
}
```

Web page with Java applet embedded

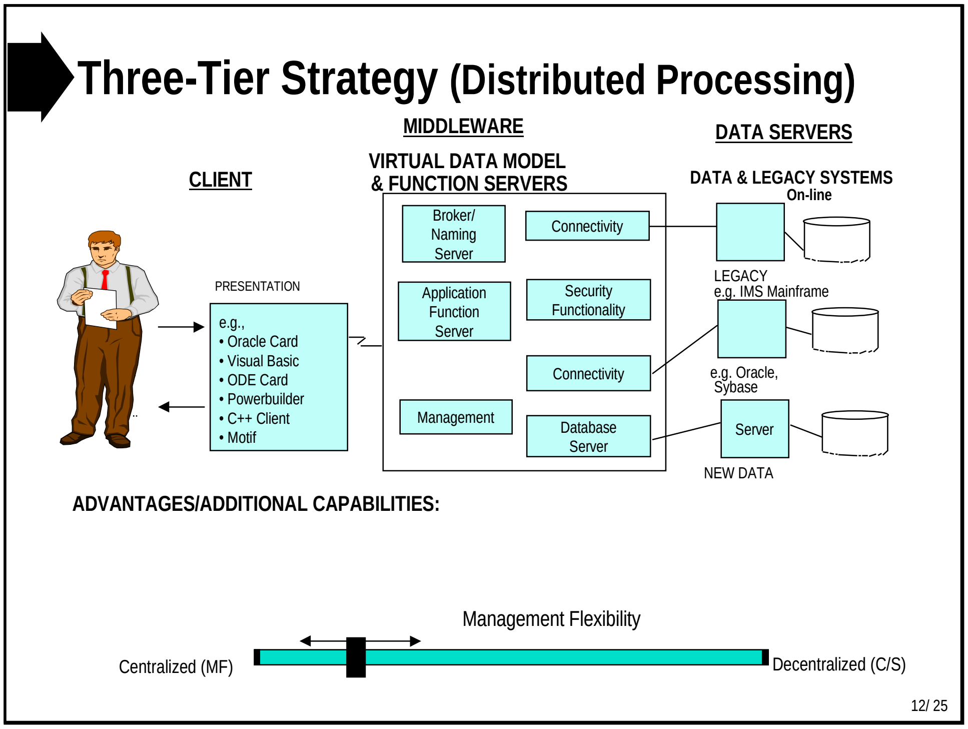
```
<html>
This simple applet example draws a sine wave.
<hr>
<applet codebase=“classes”
          code=“Wave.class”
          width=600 height=100>
</applet>
</html>
```

Browser display of wave applet being executed



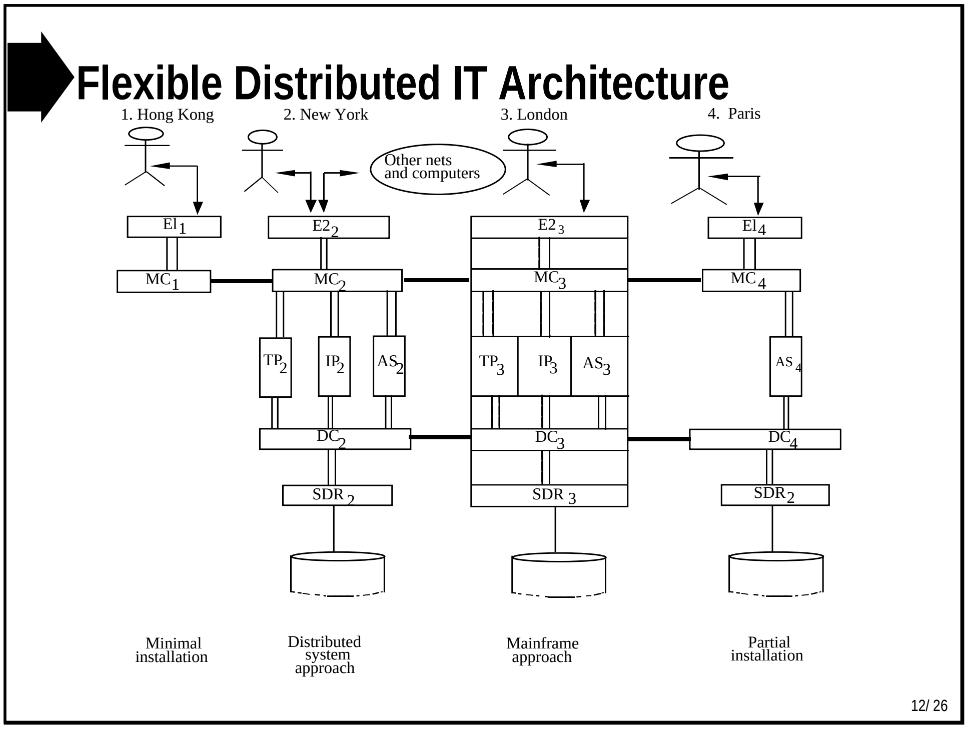
Java Reality Check / Challenges

- **Highly interactive** ⟶ <u>or</u> too slow (interpretive)
- **“Nice” programming language** ⟶ <u>or</u> too limited
- **Secure environment** ⟶ <u>or</u> (1) not secure enough or
  (2) too secure (restrictive)
- **Portable** ⟶ only if Java interpreter available
  (Java “dialects”)

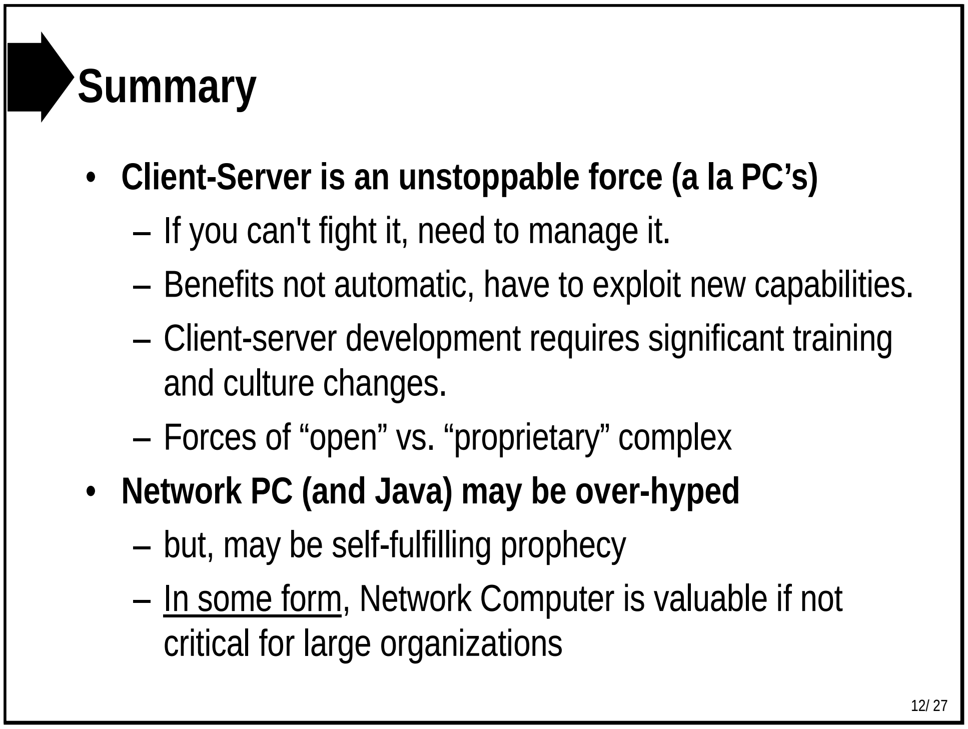


Three-Tier Strategy (Distributed Processing)

ADVANTAGES/ADDITIONAL CAPABILITIES:



Flexible Distributed IT Architecture



Summary

- **Client-Server is an unstoppable force (a la PC’s)**
  - If you can't fight it, need to manage it.
  - Benefits not automatic, have to exploit new capabilities.
  - Client-server development requires significant training and culture changes.
  - Forces of “open” vs. “proprietary” complex
- **Network PC (and Java) may be over-hyped**
  - but, may be self-fulfilling prophecy
  - <u>In some form</u>, Network Computer is valuable if not critical for large organizations