Multi Tier Computing

Unit 1

Introduction

INTRODUCTION AND BUSINESS NEEDS

- The Business opportunity- Driving forces- Major issues in information Technology Right sizing- Review of Host & Non-distributed computing.
- Basis of Distributed computing – Decomposition approaches Layers vs Tiers. Basis of Client / Server computing –Components.Client / Server computing – Approaches –Applications development-Cost – Implementation.
- OPEN SYSTEM STANDARDS FOR CLIENT/SERVER COMPUTING: Understanding Client/Server computing –Dispelling the Myths-Obstacles-Upfront and Hidden-Standards Setting Organizations-Factors for Success

The Business opportunity

- Businesses today use Information Technology (IT) to gain strategic, operational, and competitive advantages.

- **Digital transformation** enables new business models and revenue streams.
- **Information systems** provide competitive advantage through automation, speed, and intelligence.
- **Data-driven decisions** help organizations act quickly and accurately.
- **Customer experience enhancement** through personalized services, mobile apps, AI, etc.
- **Operational efficiency** through automation, optimization, and workflow improvements.

Driving Forces in Information Technology

These forces push organizations to modernize and invest in IT.

- **Globalization** → increased competition and worldwide markets.
- **Technological innovation** (Cloud, AI, IoT, Big Data) enabling smarter systems.
- **Demand for real-time information** → instant insights and faster decisions.
- **Automation & cost efficiency** → reducing manual work and errors.
- **User expectations** for accessibility, speed, personalization.

Major Issues in Information Technology

- Despite the advantages, organizations face several challenges:

- **Security threats & data privacy risks** (cyberattacks, data breaches).
- **Integration challenges** due to multiple legacy and modern systems.
- **Legacy systems** that are expensive and difficult to modernize.
- **Scalability and performance issues** as data volume and users grow.
- **High IT costs** (infrastructure, training, maintenance).
- **Skill gaps** in modern technologies (AI, cloud architecture).

Cost–Performance Trade-offs

- High performance increases infrastructure cost
- Low cost systems may reduce response time
- Balance between performance and budget is essential
- Impacts hardware, software, and networks
- Goal: Optimal performance at minimum cost

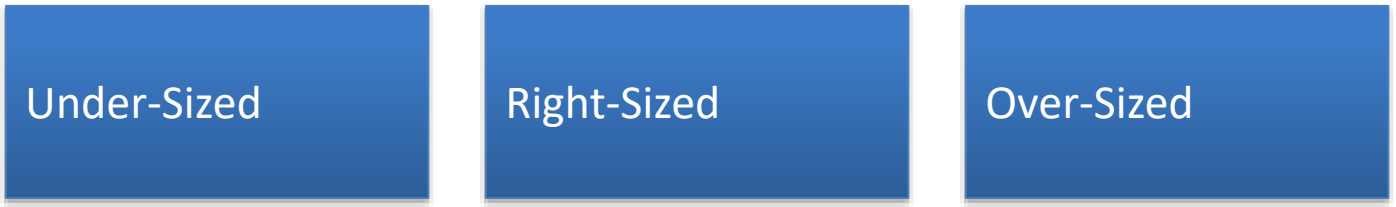
Diagram: Cost vs Performance

Right Sizing

Right sizing means **aligning IT resources exactly with business requirements.**

- Avoiding **over-sized systems** that waste money.
- Avoiding **under-sized systems** that cause delays and failures.
- Using scalable models like **distributed computing** and **cloud computing**.
- Ensuring **cost-effective performance** and future expansion capability.
- Choosing the correct mix of hardware, software, and human resources.

Diagram: Right Sizing



Host & Non-distributed Computing

- **Host (Centralized) Computing**
- Based on **mainframe systems** where all processing happens in one place.
- High reliability but **low flexibility**.
- Expensive and difficult to scale.
- Used historically by banks, government, airline reservation systems.

Non-distributed Computing

- All computing tasks occur on one system (no distribution across multiple machines).
- Limited scalability.
- Easier to manage but harder to adapt to modern needs.
- Often replaced today by **distributed** and **client/server** models.

Basics of Distributed Computing

- Computing spread across multiple nodes
- Improves scalability and performance
- Enables resource sharing and parallel processing
- Supports fault tolerance and high availability

Decomposition Approaches

- Functional decomposition
- Data decomposition
- Domain-based decomposition
- Object-oriented decomposition

Layers vs Tiers

- Layers: Logical separation of responsibilities
  - Presentation layer
  - Business logic layer
  - Data layer
- Tiers: Physical separation across machines
  - 1-tier, 2-tier, 3-tier, N-tier architectures

Layers Explained

- Presentation Layer: User interface
- Business Logic Layer: Processing & rules
- Data Layer: Storage and retrieval

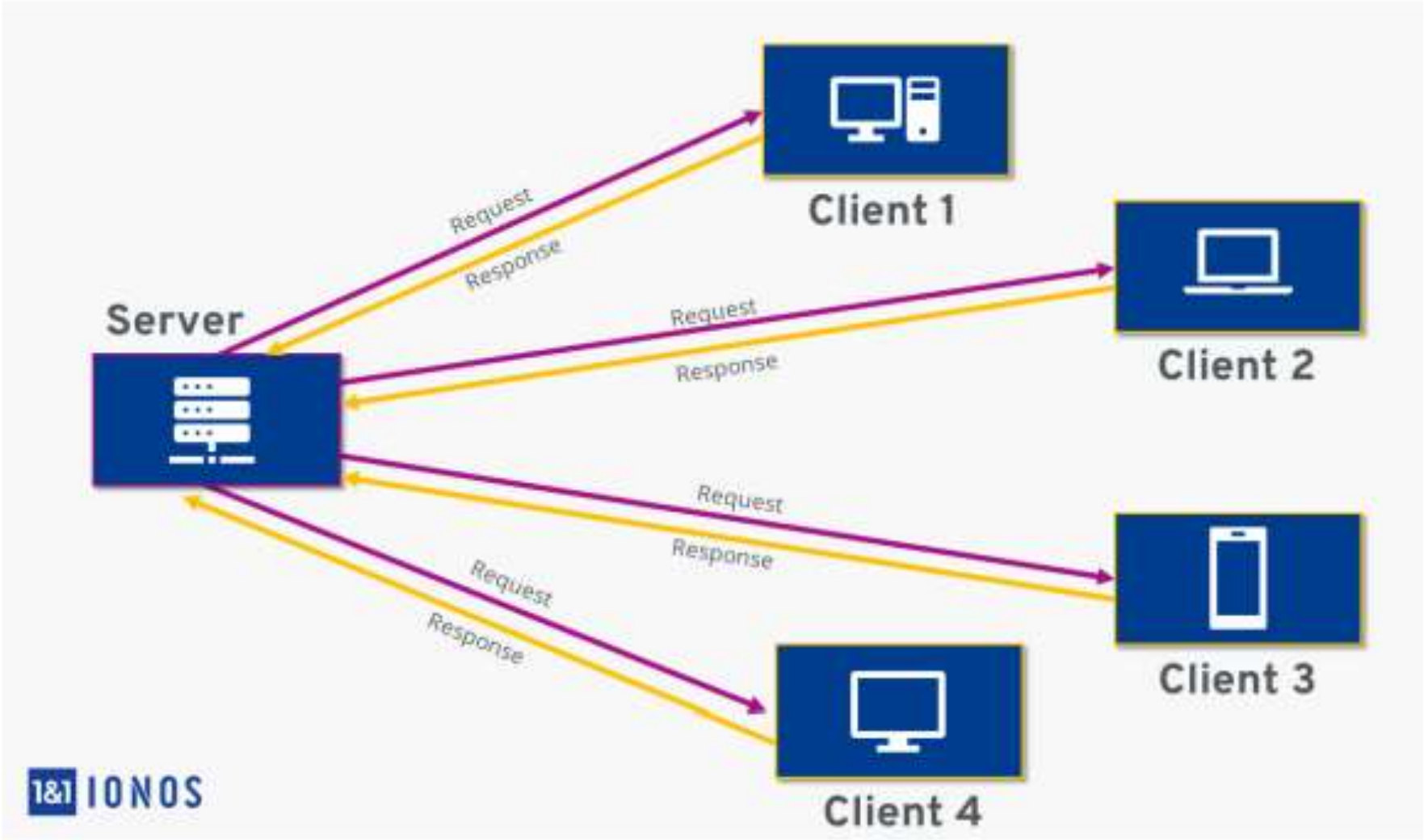
Tiers Explained

- 1-tier: All components on one system
- 2-tier: Client + Server
- 3-tier: Client + Application Server + Database
- N-tier: Multiple distributed services

WHAT IS CLIENT/SERVER COMPUTING?

- A Client is any process that requests specific services from the server process.
- A Server is a process that provides requested services for the Client.
- Client and Server processes can reside in same computer or in different computers linked by a network.

Basic Model


Server
Client 1
Client 2
Client 3
Client 4
Request
Response
Request
Response
Request
Response
Request
Response
1&1 IONOS

A Server for Every Client

- File Server
  - File Server provides clients access to records within files from the server machine.
  - UNIX: Network File Services (NFS) created by Sun Micro systems.
  - Microsoft Windows “Map Drive”
- Print Server
  - This machine manages user access to the shared output devices, such as printers.

A Server for Every Client

- Database Server
  - Database server provides access to data to clients, in response to SQL requests.
- Transaction Servers
  - The data and remote procedures reside on the server.
  - The Server provides access to high-level functions, and implements efficient transaction processing.

A Server for Every Client

- Application Server
  - This machine manages access to centralized application software; for example, a shared database.
- Web Server
  - This machine stores and retrieves Internet (and intranet) data for the enterprise.
- Mail Server
- Fax Server and many more...

Client/Server: Fat or Thin

- A thin client is one that conducts a minimum of processing on the client side.
- A **fat client** is one that carries a relatively larger proportion of processing load.

Client/Server: Fat or Thin

- Fat Server:
  - This architecture places more application functionality on the server machine(s).
  - Opposite is Thin Server

Client/Server: Stateless or Stateful

- A **stateless server** is a server that treats each request as an **independent** transaction that is unrelated to any previous request.
  - example of a stateless server is a World Wide Web server.
- The stateful server remembers what client requested previously and at last maintains the information as an incremental reply for each request.
  - example of stateful server is remote file server.

Client/Server Functions

- Managing the user interface.
- Accepts and checks the syntax of user inputs.
- Processes application logic.
- Generates database request and transmits to server.
- Passes response back to server

- Accepts and processes database requests from client.
- Checks authorization.
- Ensures that integrity constraints are not violated.
- Performs query/update processing and transmits responses to client.
- Provide concurrent database access.
- Provides recovery control.

Client/Server Topologies

Fig.1.2: Single Client, Single Server

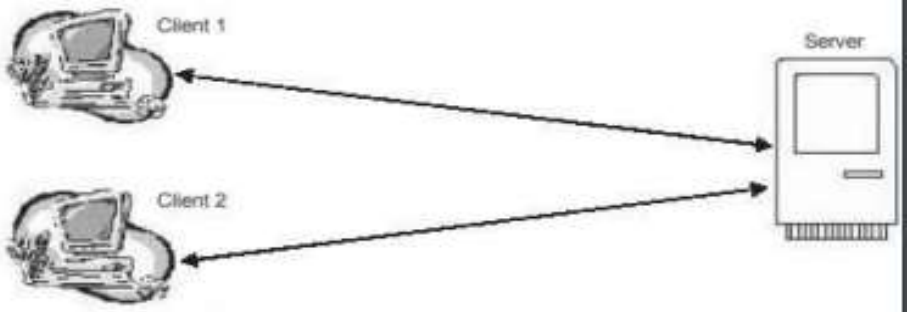


Fig.1.3: Multiple Clients, Single Server

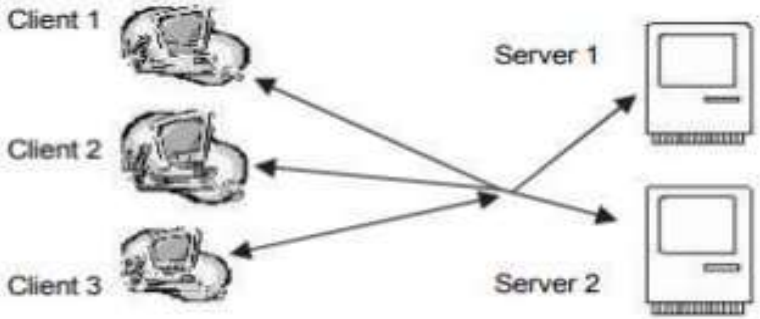


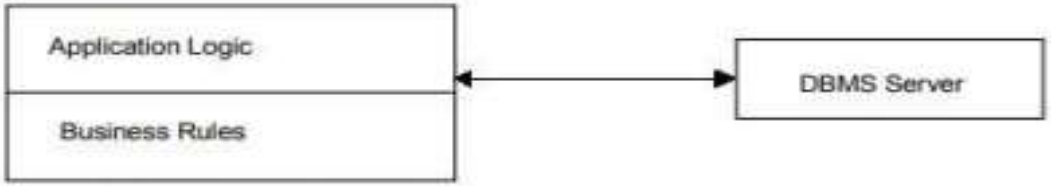
Fig.1.4: Multiple Clients, Multiple Servers

CLASSIFICATION OF CLIENT/SERVER SYSTEMS

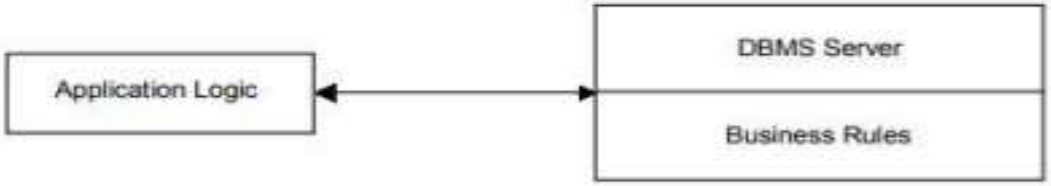
- Two-tier Client/Server Model

(*a*) Centralized Two-tier Representation:

(*b*) Business Rules Residing on the Client:



(*c*) Business Rules Residing on the Server:



CLASSIFICATION OF CLIENT/SERVER SYSTEMS

- Three-tier Client/Server Model

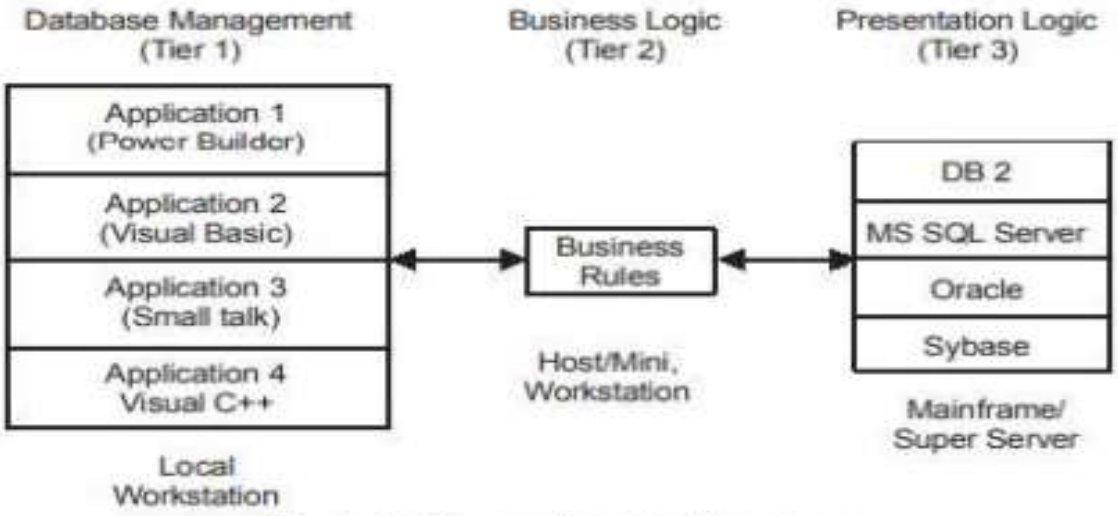
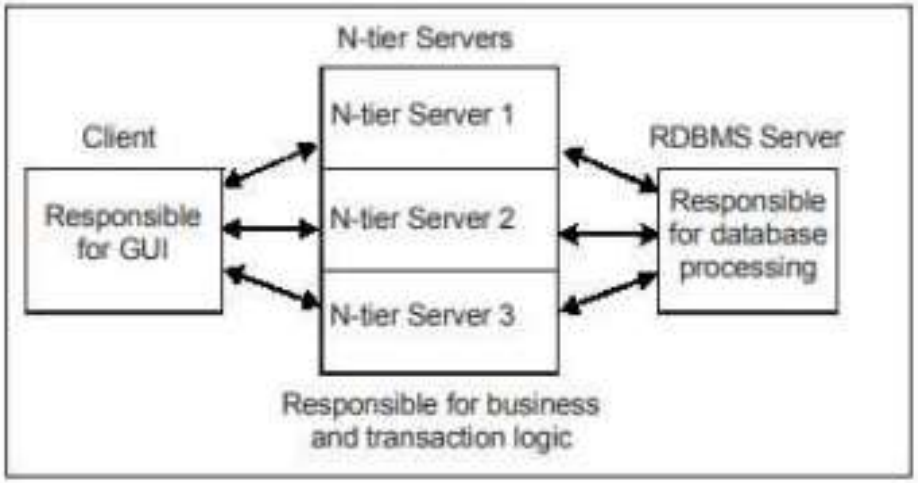


Fig.1.7: Three-tier Architecture

CLASSIFICATION OF CLIENT/SERVER SYSTEMS

- N-tier Client/Server Model



CLIENTS/SERVER—ADVANTAGES

- Performance and reduced workload
- Workstation independence
- System interoperability
- Scalability
- Data integrity
- Data accessibility
- System administration
- Sharing resources among diverse platforms
- Location independence of data processing

CLIENTS/SERVER— DISADVANTAGES

- Maintenance cost
- Training cost
- Hardware cost
- Software cost
- Complexity