SYSTEM ARCHITECTURE

What is System Architecture?

Formal Meaning

System Architecture is the **fundamental organization of a system**, including:

- System elements (hardware, software, people, processes)
- Their relationships
- Their interfaces
- Constraints
- Governing principles

It defines **how the entire system works as one integrated whole**.

Simple Meaning

System Architecture answers:

- What are the major parts of the system?
- How do they interact?
- Where do they operate?
- What rules control them?
- How does the system behave in real life?

It is the **highest-level structural design of a system in its environment.**

Scope of System Architecture

System Architecture is broader than software.

It includes:

🔹 Hardware Architecture

- Servers
- Devices
- Sensors
- Routers
- Physical infrastructure

🔹 Software Architecture

- Applications
- Services
- Modules
- APIs
- Databases

🔹 Network Architecture

- Communication links
- Protocols
- Data routing

🔹 **Operational Architecture**

- How users interact
- Business processes
- Maintenance procedures

🔹 **Security Architecture**

- Access control
- Encryption
- Firewalls
- Risk management

Core Elements of System Architecture

1. Components

Components are **building blocks** of the system.

They can be:

- Physical (servers, devices)
- Logical (software modules)
- Human (operators, administrators)

Example – Smart Traffic System

Components:

- Traffic cameras

- Central server
- Monitoring software
- Traffic control officers

2. Connectors

Connectors define how components interact.

They may represent:

- Physical connections (network cables)
- Logical connections (API calls)
- Communication protocols

Example

Camera → sends data → Server
Server → sends alert → Mobile App

3. Interfaces

Interfaces define interaction boundaries.

Example:

- REST API endpoint
- Hardware port
- Network protocol (TCP/IP)

Interfaces reduce system coupling and increase modularity.

4. Properties

Properties describe non-structural characteristics:

- Performance
- Reliability
- Scalability
- Security level
- Latency

Example:

- System must handle 10,000 concurrent users.
- System uptime must be 99.99%.

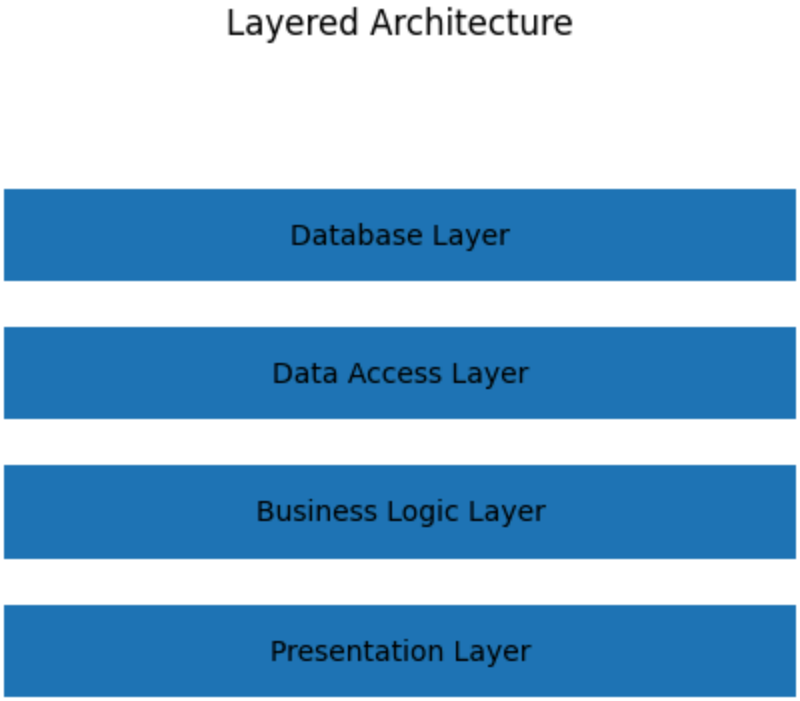
5. Architectural Style

An architectural style defines:

- Design vocabulary
- Allowed component types
- Interaction rules
- Constraints

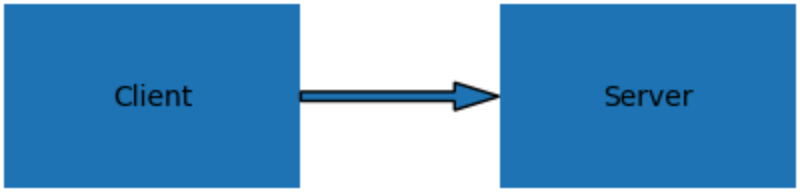
Examples:

- Layered Architecture

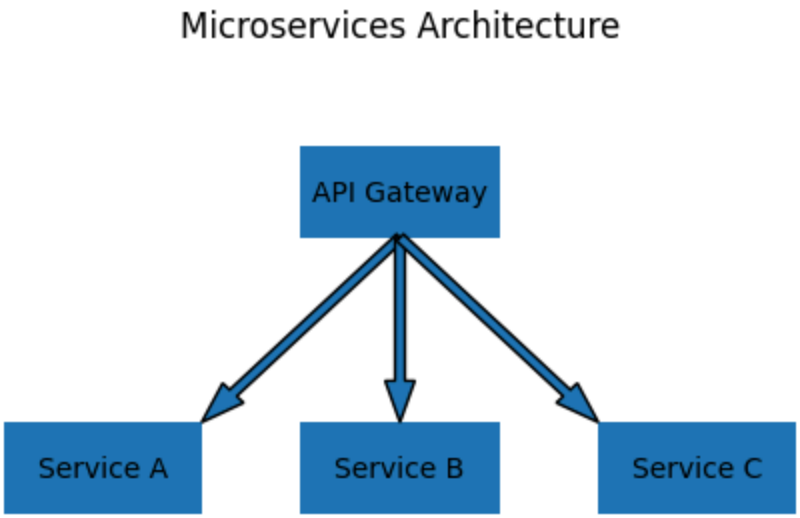


- Client-Server

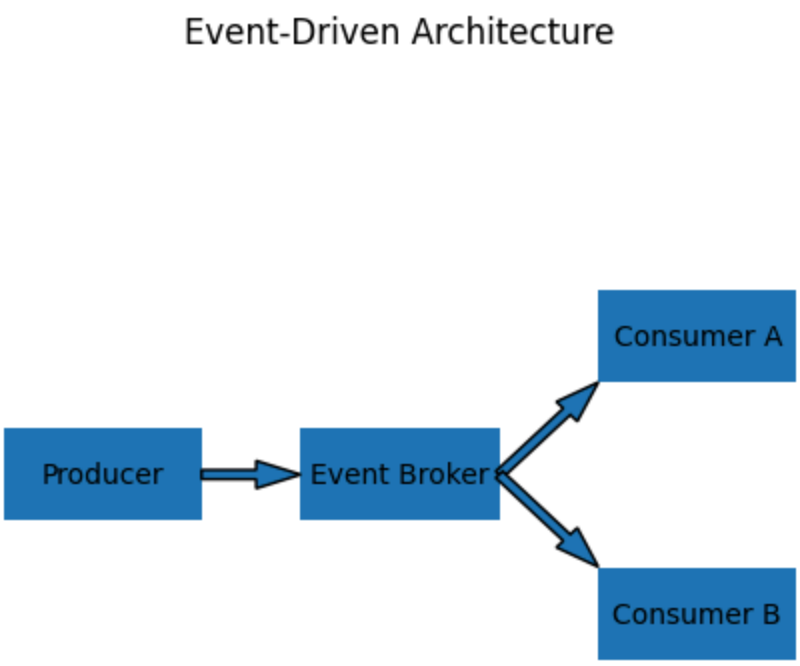
Client-Server Architecture



- Microservices



- Event-Driven



Types of System Architecture Views

System Architecture is described using multiple views.

A. Functional Architecture

Describes **what the system does**.

Focus:

- Activities
- Processes
- Data flows

- Operational behavior

Example – Insurance System

Agent creates quotation
↓
Customer pays premium
↓
System validates payment
↓
Policy issued
↓
Notification sent

This defines system behavior — independent of technology.

B. Physical Architecture

Describes **where system elements are located physically**.

Focus:

- Servers
- Infrastructure
- Deployment topology

Example – AWS Deployment

Internet
↓
Load Balancer
↓
App Server
↓
Database
↓
Backup Storage

This shows physical placement of resources.

C. Technical Architecture

Defines **rules and constraints** governing system design.

Focus:

- Standards
- Technologies
- Security policies
- Integration rules

Example:

- All APIs must use HTTPS
- Authentication must use OAuth2
- Database must be encrypted at rest

Operational Concept

Operational Concept describes:

How the system will be used in real-world scenarios.

It defines:

- Stakeholders
- Lifecycle phases
- Usage conditions
- Effectiveness requirements

Example – Hospital System

Stakeholders:

- Patients
- Doctors
- Nurses
- Admin staff

Scenario:
Patient arrives → Registration → Diagnosis → Billing → Pharmacy

This defines how the system behaves operationally.

System Architect Role

A System Architect:

- Understands all components
- Understands their interrelationships
- Balances trade-offs (cost vs performance vs security)
- Makes early structural decisions

They operate at a **strategic level**, not just coding level.

Traditional vs Evolutionary System Architecting

Traditional Approach

- Define requirements first
- Design once

- Build once
- Fixed plan

Works well when:

- Requirements are stable
- Technology is mature

Evolutionary Approach

- Build incrementally
- Prototype
- Adapt to change
- Explore alternatives

Works well when:

- Technology evolves fast
- Requirements are uncertain
- Long system lifecycle

Modern systems prefer this approach.

Architecture Development Process

System Architecture development has three main phases:

Phase 1 – Analysis

- Define requirements

- Identify stakeholders
- Define constraints
- Develop operational concept

Phase 2 – Synthesis

- Create functional model
- Create physical model
- Define technical standards

Phase 3 – Evaluation

- Measure performance
- Assess effectiveness
- Evaluate risks
- Refine architecture

Executable Model

Architecture diagrams are static.

To validate architecture:

- Simulation
- Load testing
- Stress testing
- Performance modeling

Example:
Simulate 10,000 concurrent insurance policy issuances.

Key Challenges in System Architecture

Modern systems face:

- Uncertain requirements
- Rapid technological change
- Long lifecycle
- Security threats
- Integration complexity

Architecture helps manage uncertainty.

Relationship Between System & Software Architecture

System Architecture
⬇
Includes Software Architecture

Software Architecture focuses only on software structure.

System Architecture includes:

- Hardware
- Network
- Software
- Operations
- Security
- Environment