HNDIT 4012 Software Engineering

Software Architecture

Topics covered

- Software Architectures
- Software Patterns
- MVC
- Layered
- Client Server
- Repository
- Pipe and Filter

What is Software Architecture?

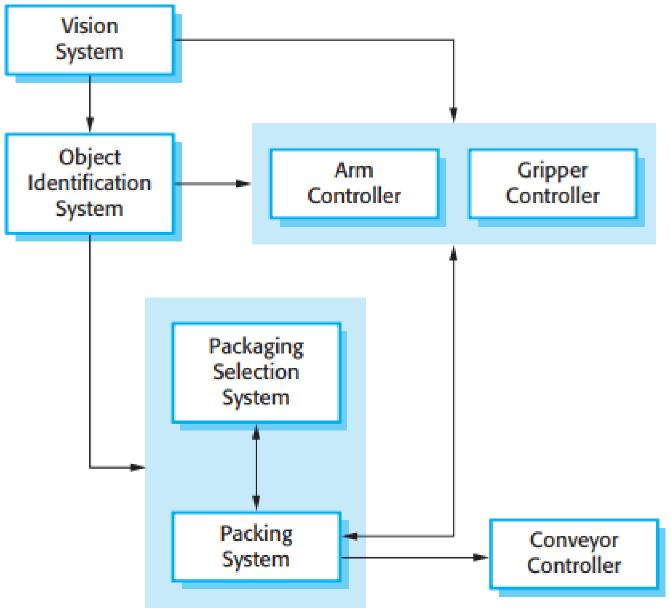
- Software architecture refers to the high-level structure or organization of a software system, which defines the components, relationships, and interactions that make up the system.

Key Aspects of Software Architecture

1. **Components:**
    - The software system is decomposed into modular components, each responsible for a specific set of functionalities or services. Components may include modules, classes, libraries, services, databases, and external dependencies.
2. **Connectors:**
    - Connectors define the communication and interaction mechanisms between components, facilitating data exchange, control flow, and coordination within the system. Examples of connectors include method calls, messages, APIs, and protocols.
3. **Architectural Styles and Patterns:**
    - Architectural styles and patterns provide reusable solutions to common design problems, guiding the selection and arrangement of components and connectors in the system. Examples include client-server, layered architecture, microservices, and event-driven architecture.

Example- Software Architecture

- The architecture of a packing robot control system.

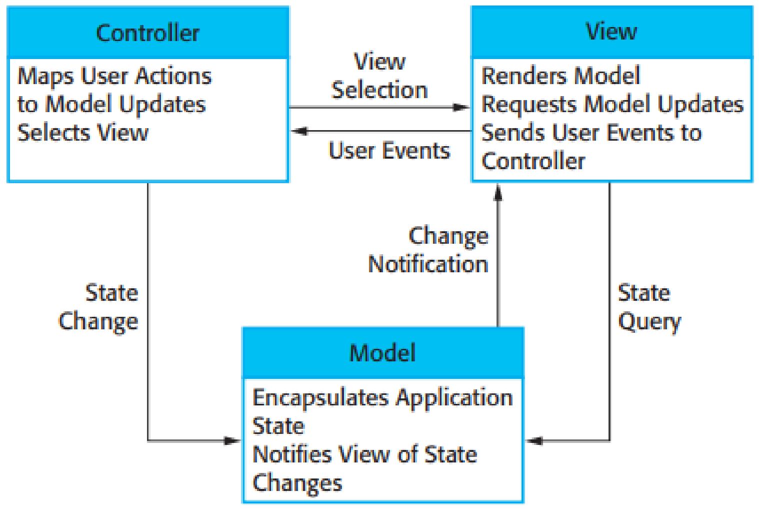


Architectural Patterns

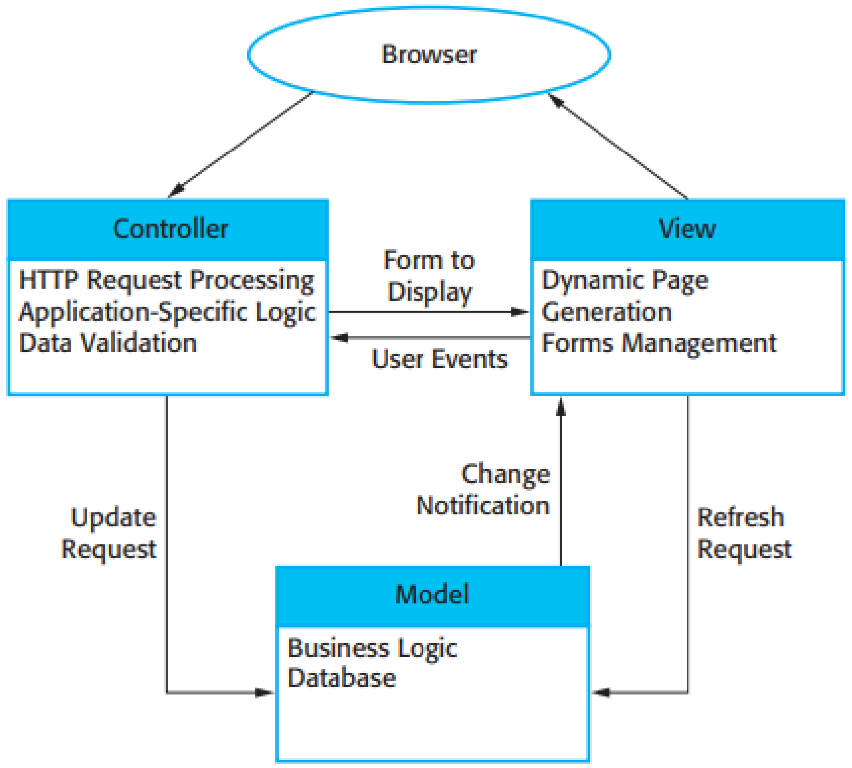
- You can think of an architectural pattern as a stylized, abstract description of good practice, which has been tried and tested in different systems and environments.

Model-View-Controller

- This pattern is the basis of interaction management in many web-based systems.



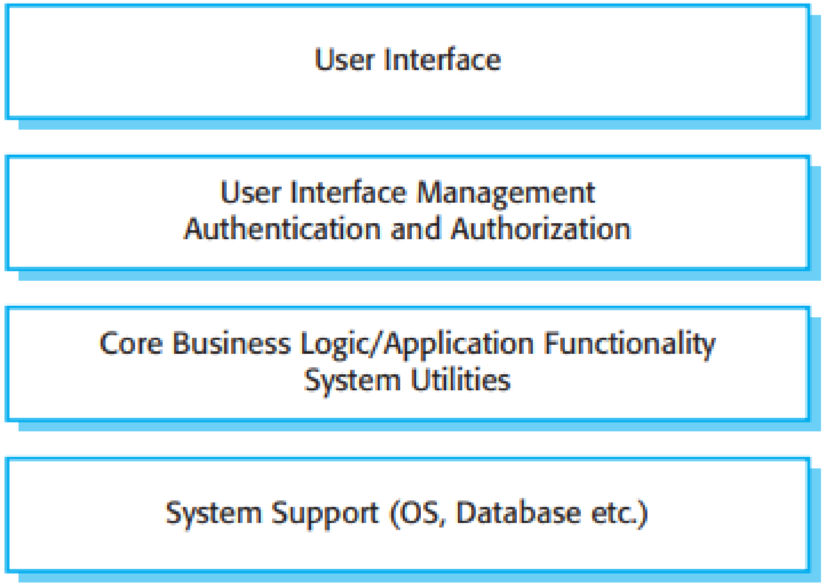
Example-MVC


Browser
Controller
HTTP Request Processing
Application-Specific Logic
Data Validation
Form to Display
User Events
View
Dynamic Page Generation
Forms Management
Update Request
Change Notification
Refresh Request
Model
Business Logic
Database

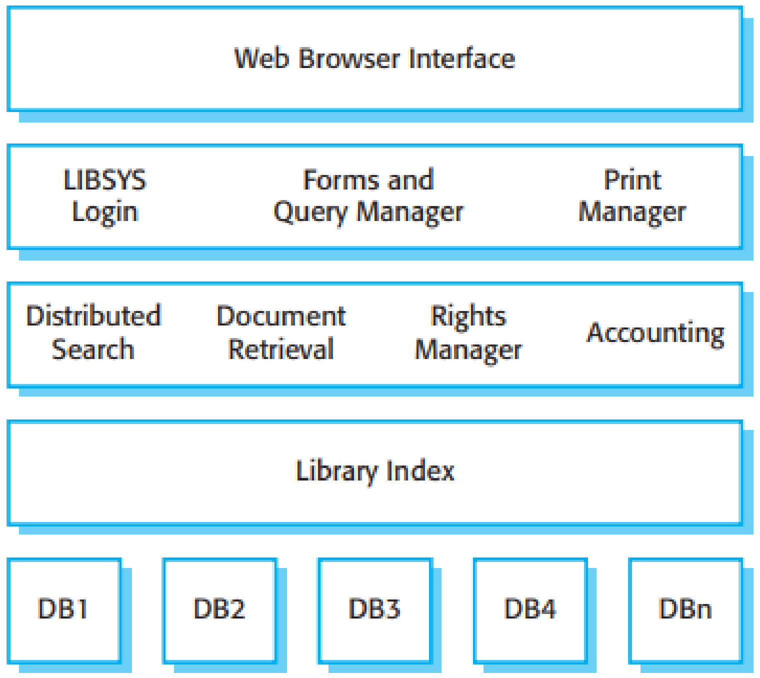
Layered architecture

- This layered approach supports the incremental development of systems.
- As a layer is developed, some of the services provided by that layer may be made available to users.
- The architecture is also changeable and portable.

Generic Layers


User Interface
User Interface Management
Authentication and Authorization
Core Business Logic/Application Functionality
System Utilities
System Support (OS, Database etc.)

Example- Layered Architecture

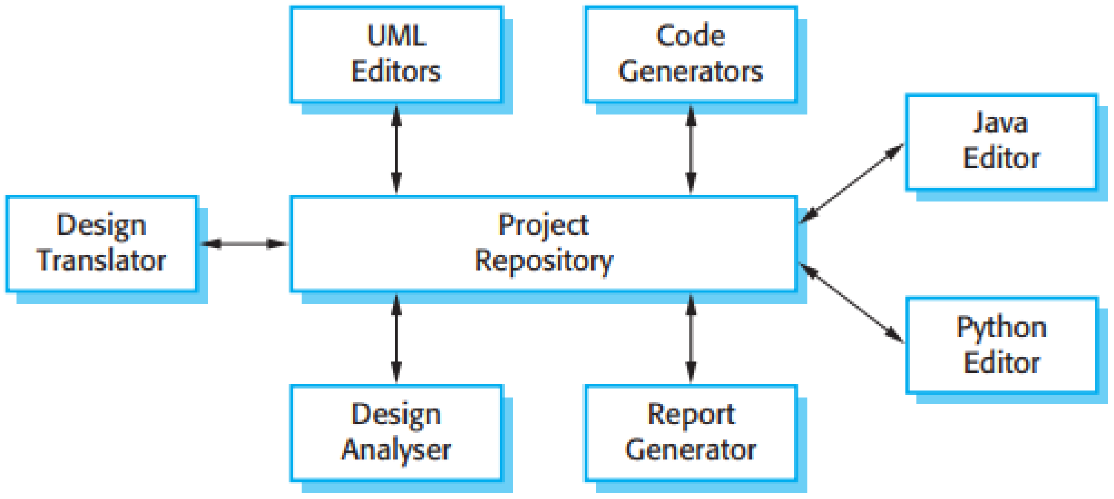


Repository Architecture

- All data in a system is managed in a central repository that is accessible to all system components. Components do not interact directly, only through the repository.

Example- Repository Architecture

- A repository architecture for an IDE.

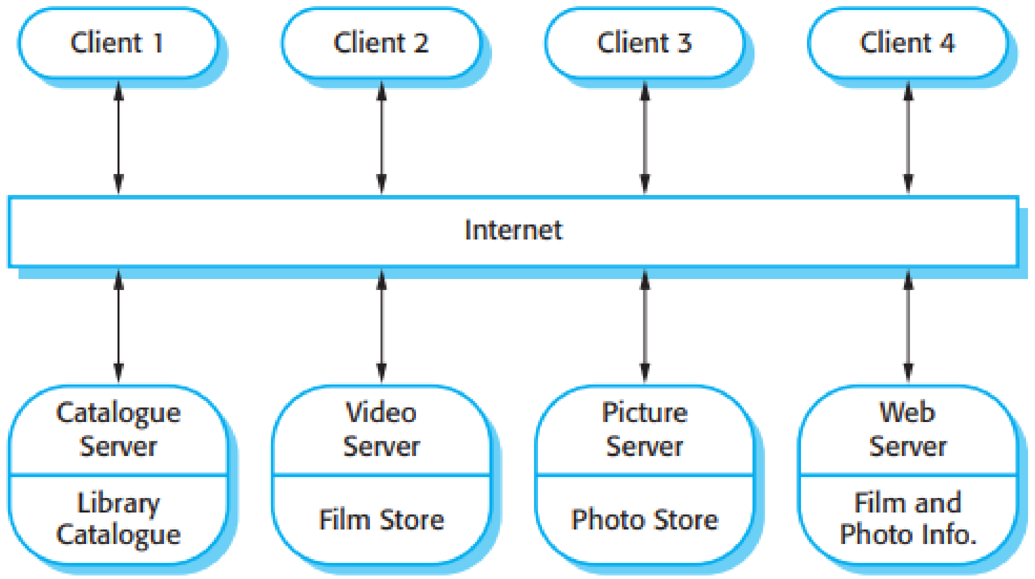


Client Server Architecture

- In a client–server architecture, the functionality of the system is organized into services, with each service delivered from a separate server.
- Clients are users of these services and access servers to make use of them.

Example- Client Server Model

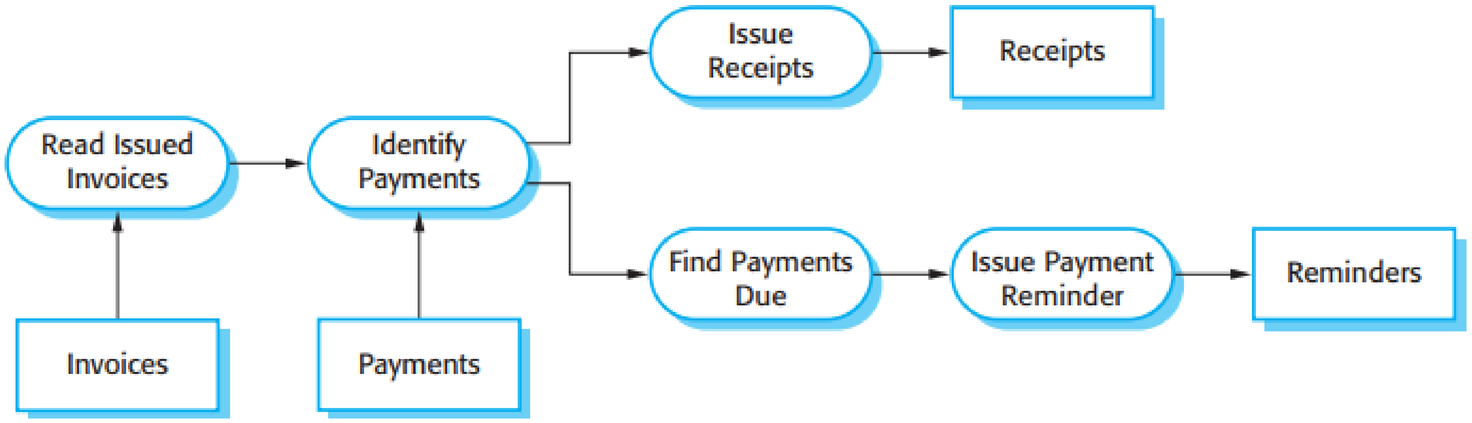
- This is a multi-user, web-based system for providing a film and photograph library.



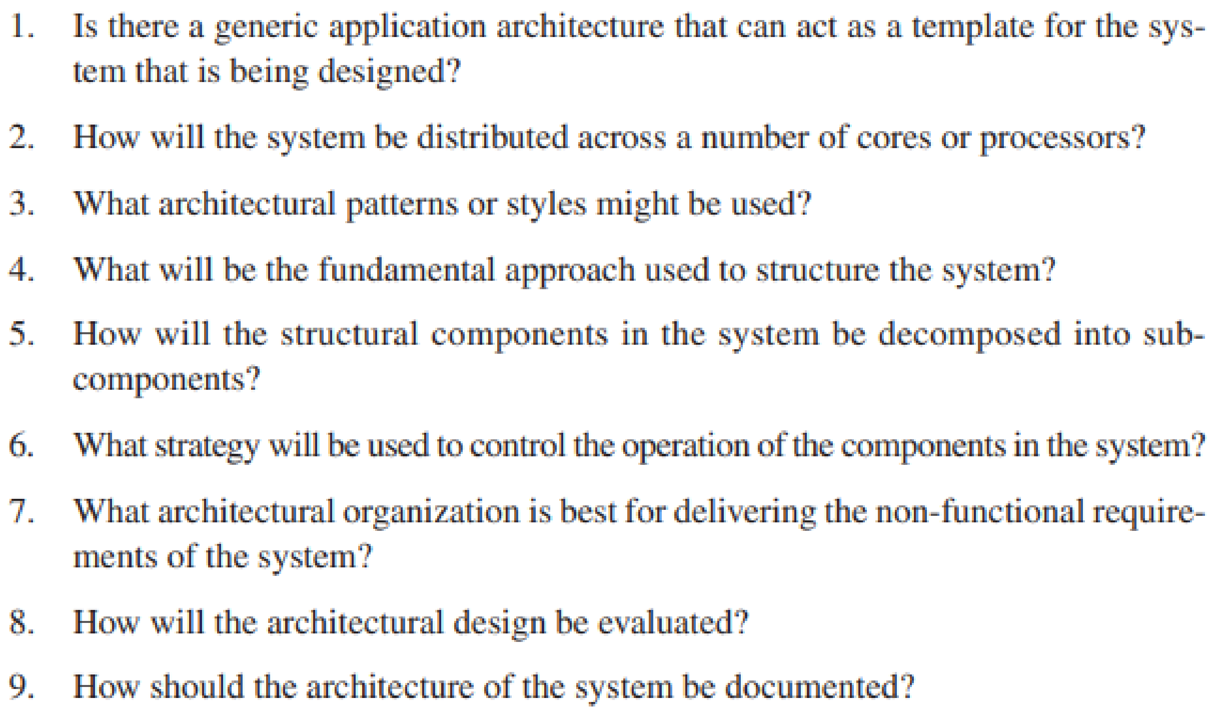
Pipe and Filter Architecture

- This is a model of the run-time organization of a system where functional transformations process their inputs and produce outputs.
- Data flows from one to another and is transformed as it moves through the sequence.

Example- Pipe & Filter



Architectural Design Decisions



1. Is there a generic application architecture that can act as a template for the system that is being designed?
2. How will the system be distributed across a number of cores or processors?
3. What architectural patterns or styles might be used?
4. What will be the fundamental approach used to structure the system?
5. How will the structural components in the system be decomposed into subcomponents?
6. What strategy will be used to control the operation of the components in the system?
7. What architectural organization is best for delivering the non-functional requirements of the system?
8. How will the architectural design be evaluated?
9. How should the architecture of the system be documented?

Advantages

- Stakeholder communication
- System analysis
- Large-scale reuse

Discussion

- What is Service Oriented Architecture?
- SOA decomposes the application into loosely coupled, reusable services that expose functionality via standardized interfaces (e.g., web services).
- Services are designed to be independent, composable, and interoperable across different platforms and technologies.