Web Application Architectures

Module 1: Introduction and Background
Lecture 4: Design Patterns

THE UNIVERSITY *of* NEW MEXICO

Web Application Architectures

- We have already seen that modern web applications involve a significant amount of complexity, particularly on the server side.
- A typical web application involves numerous protocols, programming languages and technologies spread throughout the web stack.
- Developing, maintaining and extending a complex web application is extremely difficult – but, building it using a foundation of solid design principles can simplify each of these tasks.
- Software engineers use abstraction to deal with this type of complexity.
- Design patterns provide useful design abstractions for object-oriented systems.

Design Patterns

Definition (Design Pattern)

A design pattern is a description of interacting objects and classes that interact to solve a general design problem within a particular context.

- A design pattern is an abstract template that can be applied over and over again.
- The idea is apply abstract design patterns in order to solve specific design problems that occur while building real systems.
- Design patterns provide a way to communicate the parts of a design, i.e., it's the vernacular software engineers use to talk about designs.

Client-Sever Model

- The whole point of a client-server architecture is to distribute the components of an application between the client and the server in some way. E.g., this makes sense if your trying to share a database or files among some users, share printers, etc.
- What gets put where determines the particular type of the client-server architecture.
- In order to build complex web applications, we're going to make use of numerous design patterns that will help us organize how pieces are placed within the client-server architecture.

n-Tier Architecture

- The *n*-tier architecture is a highly useful design pattern that maps to the client-server model.
- This design pattern is based on the concept of breaking a system into different pieces or tiers that can be physically separated:
  - Each tier is responsible for providing a specific functionality.
  - A tier only interacts with the tiers adjacent to it through a well-defined interface.

Ex.

- Print server – 2-tier architectural pattern.
- Early web applications – 2-tier client-server architecture:
  - User interface (browser) functionality resided on the (thin) client.
  - Server provided static web pages (HTML).
  - Interface between the two via the hypertext transfer protocol (HTTP).

n-Tier Architecture

- Additional tiers show up when the application functionality is further partitioned.
- What are the advantages of such a design?
  - The abstraction provides a means for managing the complexity of the design.
  - Tiers can be upgraded or replaced independently as requirements or technology change — the new tier just needs to use the same interfaces as the old one.
  - It provides a balance between innovation and standardization.
  - Tiered systems are much easier to build, maintain, scale and upgrade.

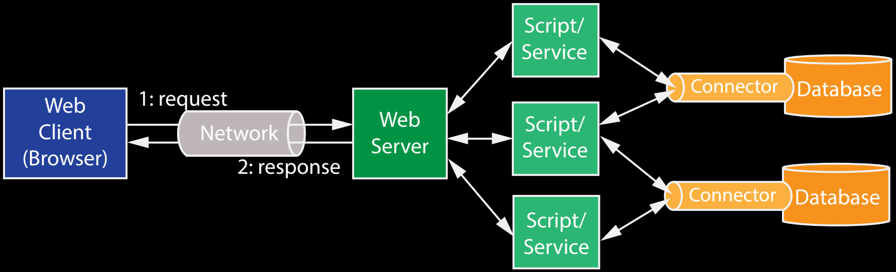
3-Tier Architecture

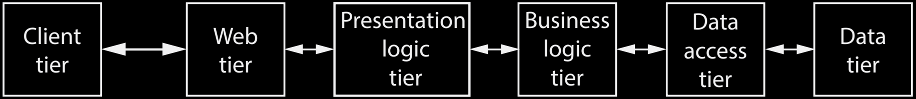
- One of the most common is the 3-tier architecture:
  - Presentation tier – The user interface.
  - Application (logic) tier – Retrieves, modifies and/or deletes data in the data tier, and sends the results to the presentation tier. Also responsible for processing the data itself.
  - Data tier – The source of the data associated with the application.
- A modern web application is often deployed over the Internet as a 3-tier architecture:
  - Presentation tier – User's web browser.
  - Application (logic) tier – The web server and logic associated with generating dynamic web content, e.g., collecting and formatting the results of a search.
  - Data tier – A database.

6-Tier Web Application Architecture

- The Application tier is often subdivided into two tiers:
  - Business logic tier – Models the business objects associated with the application, e.g., accounts, inventories, etc., and captures the business rules and workflows associated with how these processes can be processed and manipulated.
  - Data access tier – Responsible for accessing data, and passing it to the business logic tier, e.g., account balances, transactions, etc.
- The Presentation tier is often subdivided into two tiers:
  - Client tier – client-side user interface components.
  - Presentation logic tier – server-side scripts for generating web pages.
- Finally, the web server is often separated out into its own Web tier.

Web Application Architecture Tiers


Web Client (Browser)
1: request
Network
2: response
Web Server
Script/ Service
Script/ Service
Script/ Service
Connector
Database
Connector
Database


Client tier
Web tier
Presentation logic tier
Business logic tier
Data access tier
Data tier