CS 4320 / 7320 Software Engineering

Module 5 – Design:
Structure & Architecture, UI Design

Software Structure and Architecture:
Architectural Design:

A problem-solving, creative process
with decisions involving trade-offs,
often on quality attributes

Software Structure and Architecture: *Architectural Styles*

General structures: layers, pipes, filters

Distributed systems: client-server, 3-tiered, broker

Interactive systems: Model-View-Controller,

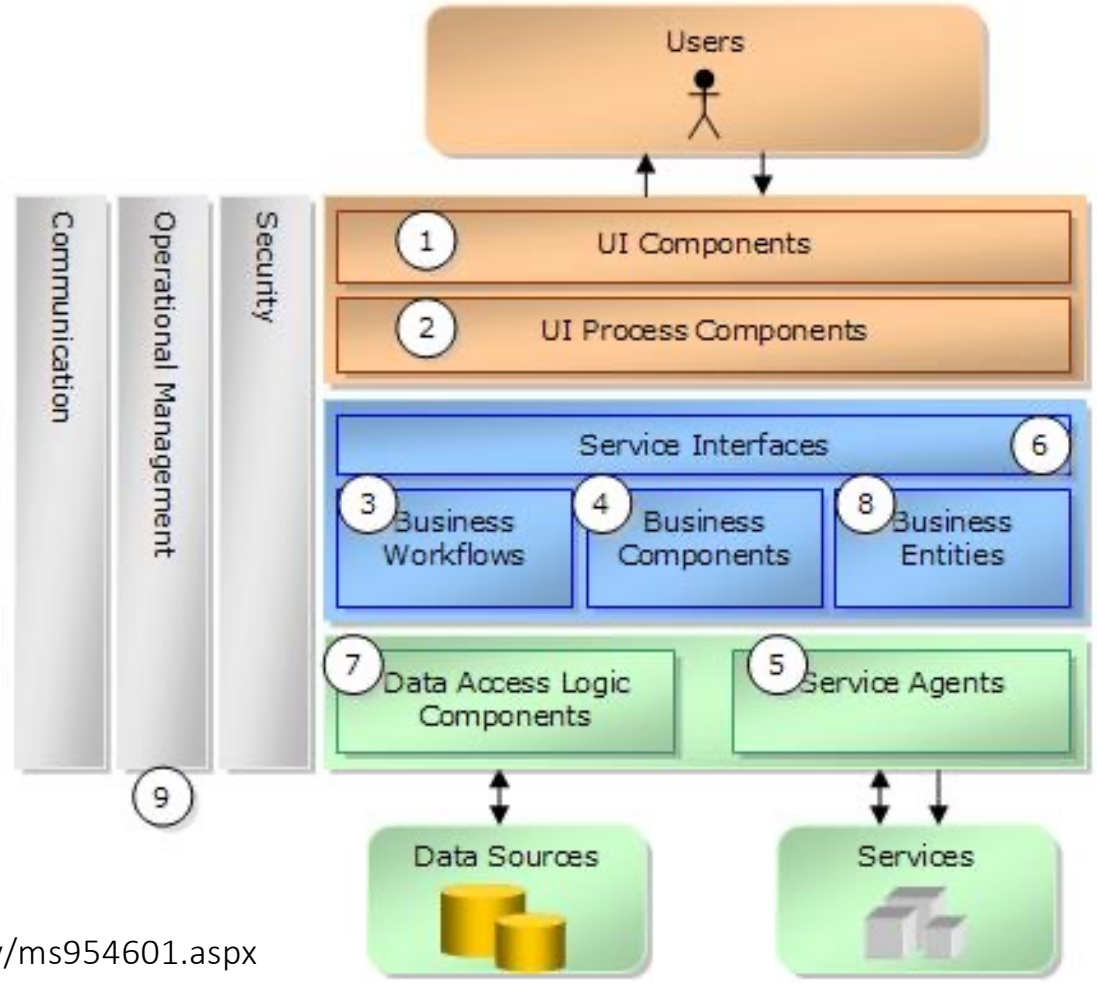
Presentation-Abstraction-Control

Others…

Software Structure and Architecture: *Architectural Styles*

What do you know about architecture styles for a typical web application?

Typical Layered Design



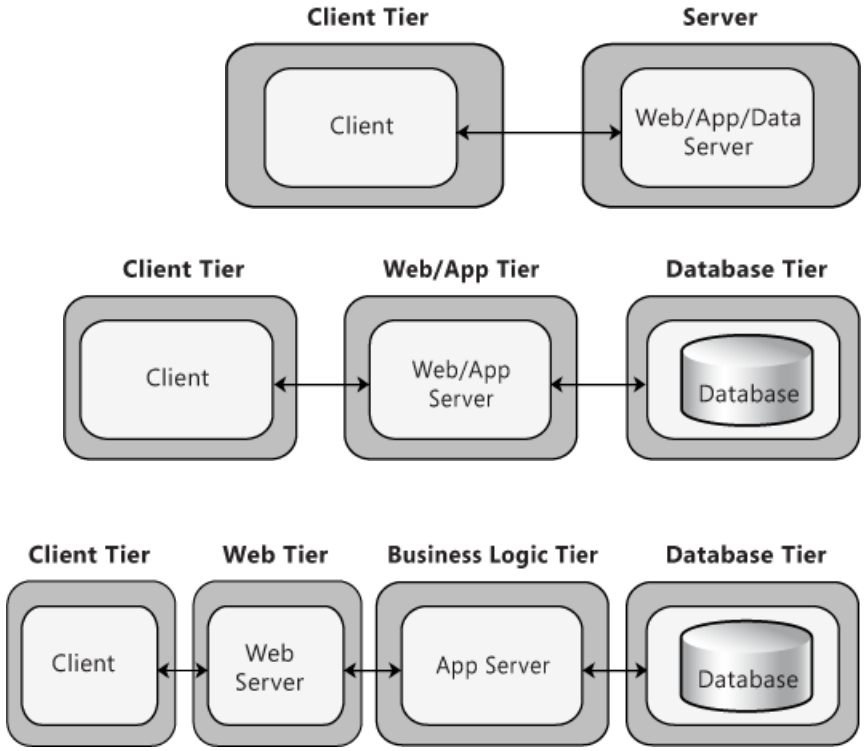
Source: https://msdn.microsoft.com/en-us/library/ms954601.aspx

Distributed Deployment Patterns

Client-Server

3-tier

4-tier



Source: https://msdn.microsoft.com/en-us/library/ee658120.aspx

Software Structure and Architecture: *Architectural Styles*

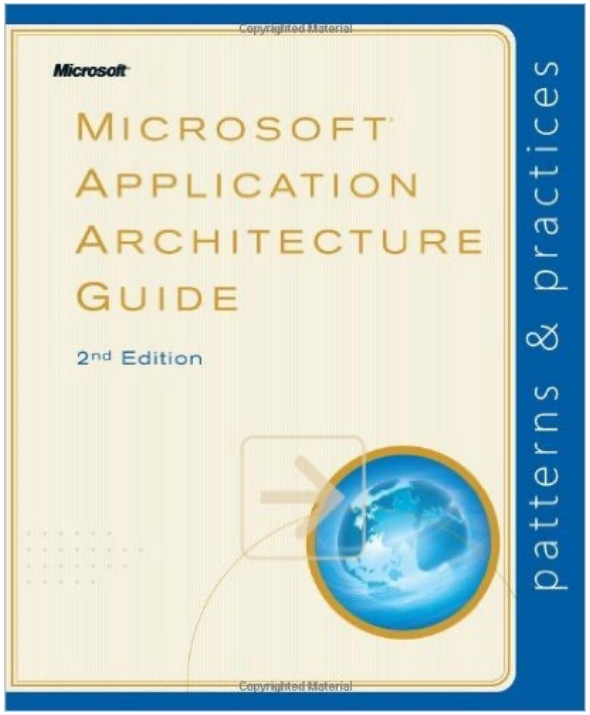
Microsoft Application Architecture Guide, 2nd Edition by Microsoft Patterns & Practices Team

ISBN-13: 978-0735627109 ISBN-10: 073562710X

Publisher: Microsoft Press (November 22, 2009)

Online Book: https://msdn.microsoft.com/en-us/library/ff650706.aspx

Chapter 3: Architectural Patterns and Styles



Software Structure and Architecture: *Design Patterns*

Common solutions to a common problem

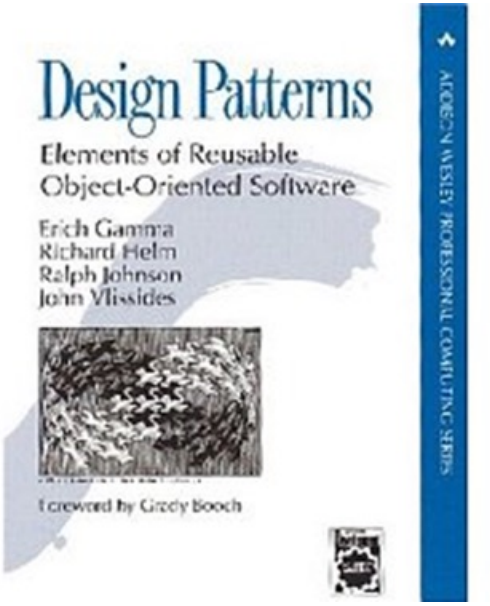
Object-oriented design patterns

- Creational: builder, factory, prototype, singleton
- Structural: adapter, bridge, composite, façade, proxy
- Behavioral: command, interpreter, iterator, observer

Software Structure and Architecture: *Design Patterns*

Design Patterns: Elements of Reusable Object-Oriented Software by ErichGamma, RichardHelm, RalphJohnson, and JohnVlissides (the GangOfFour)
ISBN 978-0201633610, ISBN 0-201-63361-2
Publisher: AddisonWesley Professional (November 10, 1994)

http://wiki.c2.com/?DesignPatternsBook



Software Structure and Architecture: *Frameworks*

Framework:

A software system providing some generic functionality to facilitate development of software solutions.

Examples: Sprint MVC (java), .Net Framework (Microsoft), Django (python), Bootstrap (html, css, js), *many, many, more….*

User Interface Design: *General Principles*

First, know your users. Then consider…

- Learnability
  - User familiarity
    - Consistency
      - Minimal surprise
        - Recoverability (from errors)
          - User guidance (feedback)
            - User diversity (accessibility)

User Interface Design:

https://www.usability.gov/what-and-why/user-interface-design.html

https://www.w3.org/WAI/intro/accessibility.php