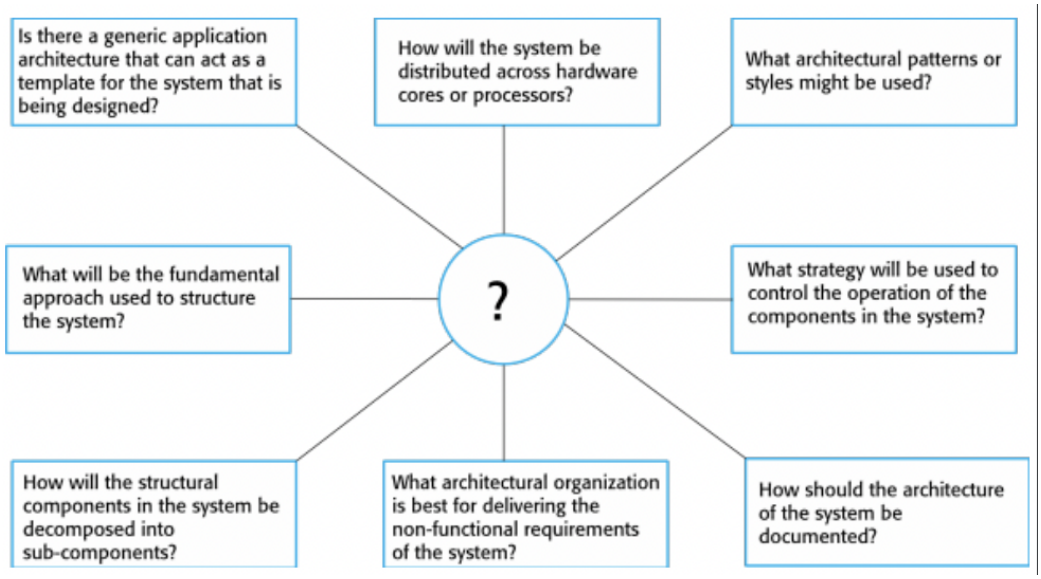
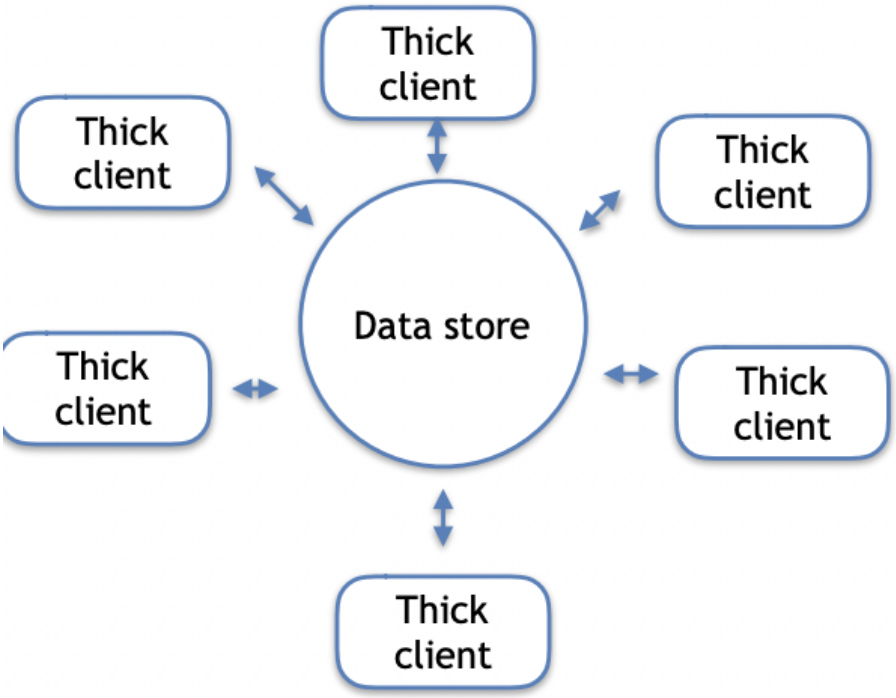
CAS CS 411
Lec 7

Software Architectures

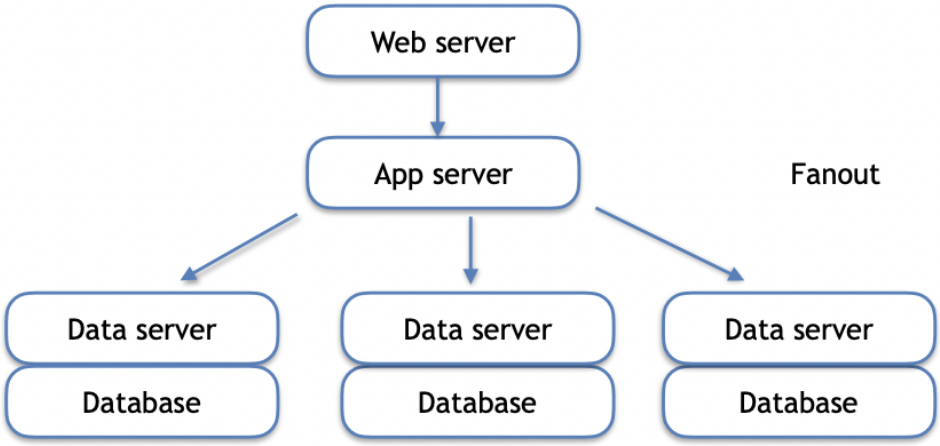
1. Architectural Design Decisions
   a. 

2. Architecture reuse
   a. Systems in the same domain often have similar architectures that reflect domain concepts.
   b. Application product lines are built around a core architecture with variants that satisfy particular customer requirements.
   c. The architecture of a system may be designed around one of more architectural patterns or ‘styles’.
      i. These capture the essence of an architecture and can be instantiated in different ways.
3. Architecture and system characteristics
   a. Performance
      i. Localise critical operations and minimise communications. Use large rather than fine-grain components.
   b. Security
      i. Use a layered architecture with critical assets in the inner layers.
   c. Safety
      i. Localise safety-critical features in a small number of sub-systems.
   d. Availability
      i. Include redundant components and mechanisms for fault tolerance.
   e. Maintainability
      i. Use fine-grain, replaceable components.

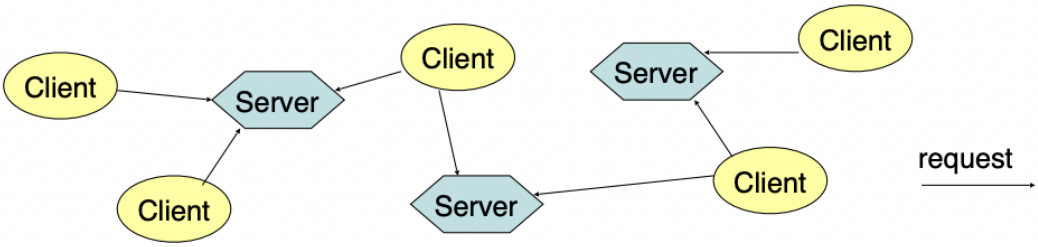
4. Data-Centered Architecture

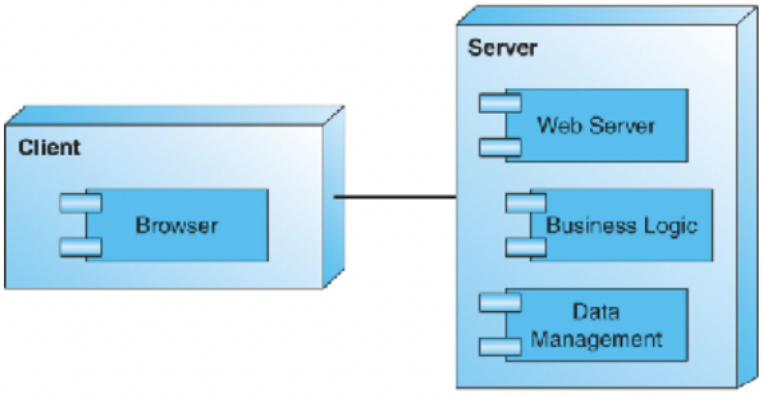


   a.
5. Data-centered issues
   a. The biggest of course is the bottleneck in the middle
   b. Mitigation strategies include sharding (horizontal scaling) and vertical scaling
   c. The network can also be a bottleneck
6. Call and Return Architecture



   a.
   b. Where is the bottleneck?
   c. Where can we improve this architecture?
   d. Are there opportunities for abstraction?

7. Client-Server Architecture
    a. Each component of a client-server system has the role of either client or server
        i. Client: a component that makes requests clients are active initiators of transactions
        ii. Server: a component that satisfies requests servers are passive and react to client requests
    b. The Web is a client-server system
    c. Web browsers act as clients, and make requests to web servers
    d. Web servers respond to requests with requested information and/or computation
    e. 

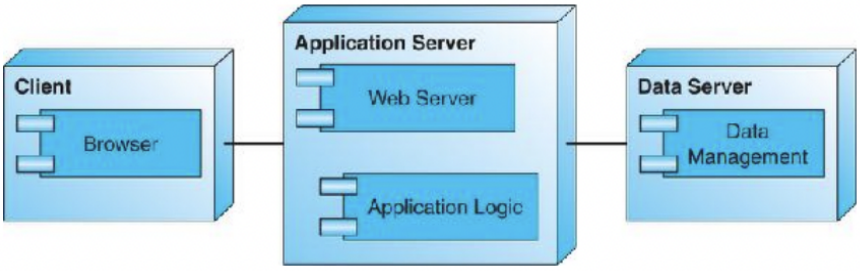
8. Thick vs Thin Client
    a. Thick client: Work is done on the client
    b. Thin client: Work is done on the server
9. Tired Web Architectures
    a. Web applications are usually implemented with 2-tier, 3-tier, or multitier (N-tier) architectures
    b. Each tier is a platform (client or server) with a unique responsibility
10. 2-Tier C-S Architecture
    a. Tier 1: Client platform, hosting a web browser
    b. Tier 2: server platform, hosting all server software components
    c. 

11. 2-Tier Characteristics
    a. Advantage:
        i. Inexpensive (single platform)
    b. Disadvantages
        i. Interdependency (coupling) of components

ii. No redundancy
iii. Limited scalability

c. Typical application
i. 10-100 users
ii. Small company or organization, e.g., law office, medical practice, local non-profit

12. 3-Tier C-S Architecture
a. Tier 3 takes over part of the server function from tier 2, typically data management



b.

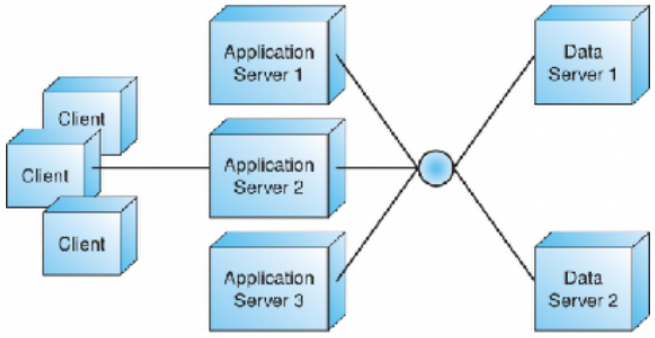
13. 3-Tier Characteristics
a. Advantages
i. Improved performance, from specialized hardware
ii. Decreased coupling of software components
iii. Improved scalability

b. Disadvantages
i. No redundancy

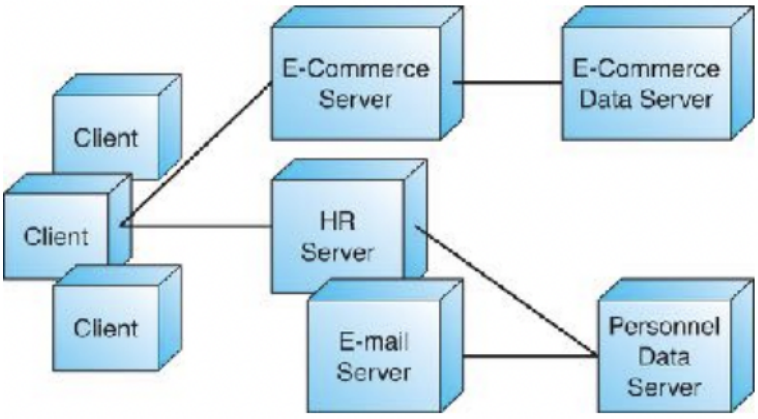
c. Typical Application
i. 100-1000 users
ii. Small business or regional organization, e.g., specialty retailer, small college

14. Multitier C-S Architecture
a. A multitier (N-tier) architecture is an expansion of the 3-tier architecture, in one of several different possible ways
i. Replication of the function of a tier
ii. Specialization of function within a tier
iii. Portal services, focusing on handling incoming web traffic

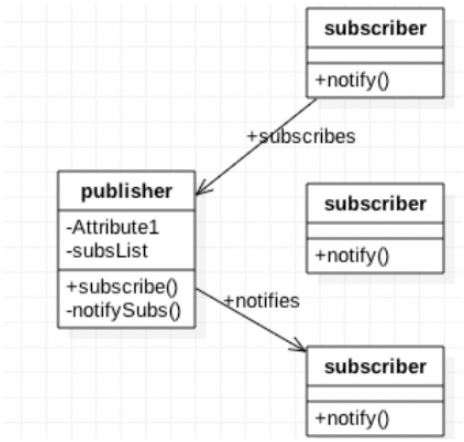
15. Replication
a. Application and data servers are replicated
b. Servers share the total workload

c. 

16. Specialization
    a. Servers are specialized
    b. Each server handles a designated part of the workload, by function

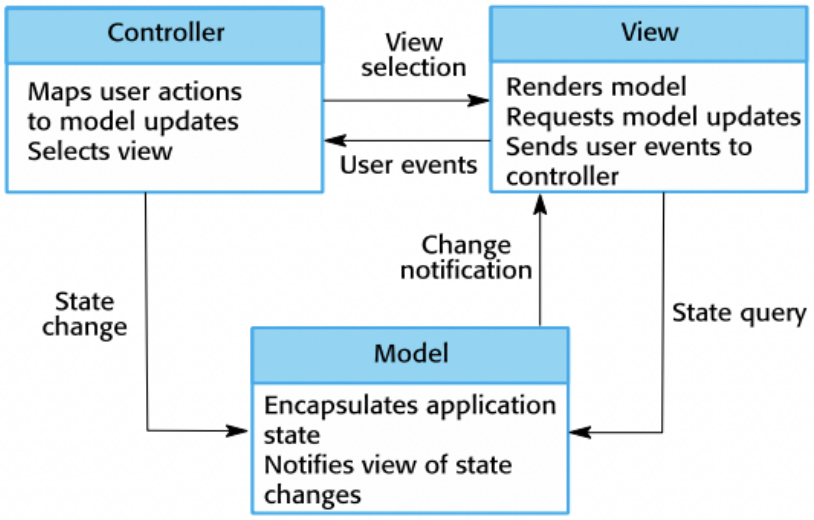
c. 

17. Pub-Sub
    a. The Publish-Subscribe (pub-sub) model implements the Gang of Four "Observer" pattern
    b. In it, an object provides a register() method that clients may call to indicate their interest in the object's state
    c. When the object's state changes it goes down its list of subscribed client objects and calls their notify() method

d. 

e. Pub-sub is often found in systems that provide a ‘pipeline’ of data
f. Subscribers can choose which data to be notified of
g. For example, Reuters and IDC provide data feeds of trading floor activity via a message-based, pub-sub service
h. It is usually more efficient than polling
i. One thing to watch out for is saturation of the publisher if the sub lists get large…decouple it and replicate to solve this

18. The Model-View-controller (MVC) pattern
    a. Before discussing MVC we need to understand the state of an object or group of objects
    b. When we talk about object we are interested in their attributes (their internal variables) and their behaviors (the methods on the object)
    c. State is the current instantaneous value of all of an object’s attributes…IOW what values do the variables hold
    d. The MVC pattern gives us a way to isolate state from other parts of the architecture
19. The organization of the Model-View-Controller



    a.

20. The Model
    a. In MVC, the model is the code that carries out some task.
    b. It is built with no necessary concern for how it will "look and feel" when presented to the user.
    c. A model is an object representing data or even activity, e.g. a database table or even some plant-floor production-machine process.
    d. The model
        i. manages the behavior and data of the application domain,
        ii. responds to requests for information about its state and
        iii. responds to instructions to change state.

e. The model often represents enterprise data and the business rules that govern access to and updates of this data. Often the model serves as a software approximation to a real-world process, so simple real-world modeling techniques apply when defining the model.
f. The model is the piece that represents the state and low-level behavior of the component. It manages the state and conducts all transformations on that state.
g. The model has no specific knowledge of either
    i. its controllers or
    ii. its views.
h. However, a model must be able to "register" views and it must be able to "notify" all of its registered views when any of its functions cause its state to be changed.
i. The system itself maintains links between model and views and notifies the views when the model changes state. The view is the piece that manages the visual display of the state represented by the model.
j. A model can have more than one view.
k. Note that the model may not necessarily have a persistent data store (database), such as when it deals with a control on a GUI screen.
l. It has a purely functional interface, meaning that it has a set of public functions that can be used to achieve all of its functionality.
m. Some of the functions are query methods that permit a "user" to get information about the current state of the model. Others are mutator methods that permit the state to be modified.
    i. setTemperature(temp) and getTemperature() – Temperature Model
    ii. addItem(item, category), setScore(item, student, score)… - Gradebook

21. Model Example
    a.

```
public class TemperatureModel extends java.util.Observable {
  public double getF(){return temperatureF;}
  public double getC(){return (temperatureF - 32.0) * 5.0 / 9.0;}
  public void setF(double tempF) {
       temperatureF = tempF;
       setChanged();
       notifyObservers();
  }
  public void setC(double tempC) {
       temperatureF = tempC*9.0/5.0 + 32.0;
   setChanged();
       notifyObservers();
  }

  private double temperatureF = 32.0;
}
```

22. View
    a. A view is some form of visualization of the state of the model.
    b. The view manages the graphical and/or textual output to the portion of the bitmapped display that is allocated to its application.

c. The view renders the contents of a model. It accesses enterprise data through the model and specifies how that data should be presented.
d. The view is responsible for mapping graphics onto a device. A view typically has a one-to-one correspondence with a display surface and knows how to render to it. A view attaches to a model and renders its contents to the display surface.
e. A model in MVC can have several views. Example:
    i. the rows and columns view of a spreadsheet and
    ii. the pie chart view of some column in the same spreadsheet.
f. A view provides graphical user interface (GUI) components for a model. It gets the values that it displays by querying the model of which it is a view.
g. When a user manipulates a view of a model, the view informs a controller of the desired change.

23. View - Code Segment

a.

```
TemperatureGUI(String label, TemperatureModel model, int h, int v)
   {
         this.label = label;
         this.model = model;
         temperatureFrame = new Frame(label);   temperatureFrame.add("Center",
    display);
         Panel buttons = new Panel();
         buttons.add(upButton);
         buttons.add(downButton);
         temperatureFrame.add("South", buttons);
         temperatureFrame.addWindowListener(new CloseListener());
         model.addObserver(this); // Connect the View to the Model
         temperatureFrame.setSize(200,100);
         temperatureFrame.setLocation(h, v);
         temperatureFrame.setVisible(true);
   }
```

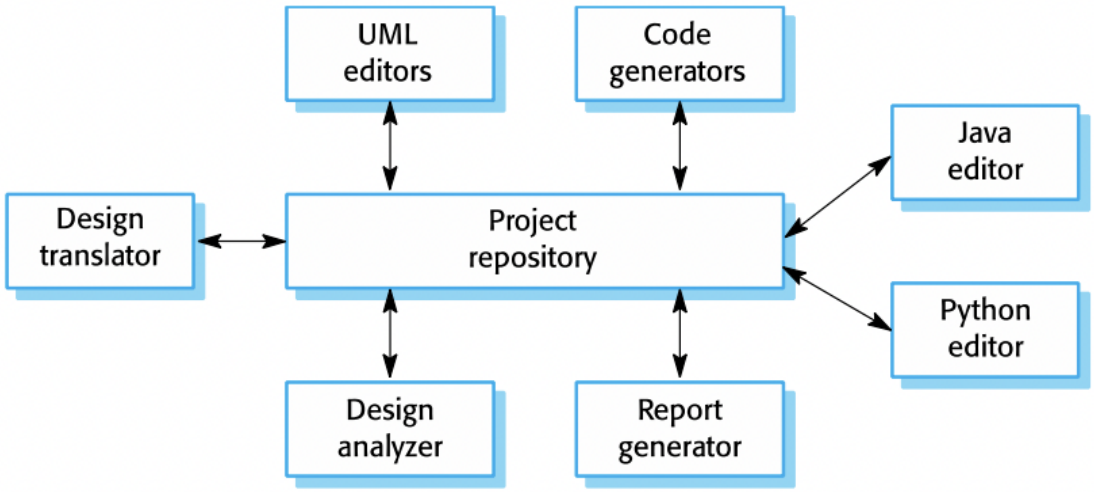
24. Controllers

a. Views in MVC are associated with controllers that update the model as necessary when a user interacts with an associated view.
b. The controller can call mutator methods of the model to get it to update its state. Of course, then the model will notify ALL registered views that a change has been made and so they will update what they display to the user as appropriate
c. A controller offers facilities to change the state of the model. The controller interprets the mouse and keyboard inputs from the user, commanding the model and/or the view to change as appropriate.
d. A controller is the means by which the user interacts with the application. A controller accepts input from the user and instructs the model and view to perform actions based on that input. In effect, the controller is responsible for mapping end-user action to application response.
e. The controller translates interactions with the view into actions to be performed by the model. In a stand-alone GUI client, user interactions could be button clicks

or menu selections, whereas in a Web application they appear as HTTP GET and POST requests.

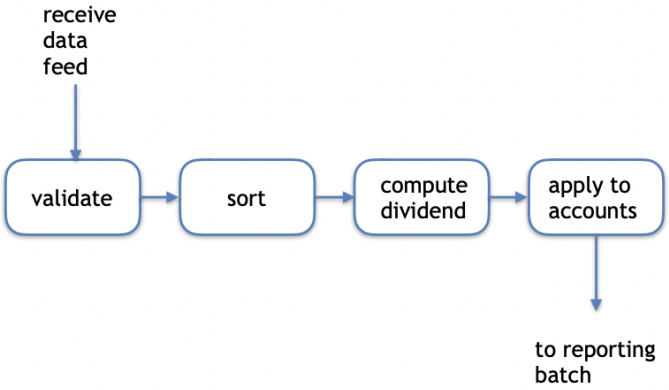
f. The actions performed by the model include activating business processes or changing the state of the model.
g. Based on the user interactions and the outcome of the model actions, the controller responds by selecting an appropriate view.
h. The controller is the piece that manages user interaction with the model. It provides the mechanism by which changes are made to the state of the model.

25. Repository architecture
    a. Sub-systems must exchange data. This may be done in two ways:
        i. Shared data is held in a central database or repository and may be accessed by all sub-systems;
        ii. Each sub-system maintains its own database and passes data explicitly to other sub-systems.
    b. When large amounts of data are to be shared, the repository model of sharing is most commonly used a this is an efficient data sharing mechanism.
26. A repository architecture for an IDE
    a.



27. Pipe and filter (batch) architecture
    a. Functional transformations process their inputs to produce outputs.
    b. May be referred to as a pipe and filter model (as in UNIX shell).
    c. Variants of this approach are very common. When transformations are sequential, this is a batch sequential model which is extensively used in data processing systems.
    d. Not really suitable for interactive systems.

28. An example of batch processing of dividend payments



a.
b. Batch is used quite a lot
c. Typically the batch is run on a schedule, often after business hours
d. Products like IBM Maestro are used to coordinate and schedule batch jobs
e. An advantage: Individual batch components can be worked on or replaced without modifying the workflow…just swap in the new component (assuming it implements the old interface)
f. Disadvantage: Bottlenecks possible in any component

29. Bottom line
    a. Each project is different; choose an architecture that best fits the application
    b. Sometimes this is obvious, but sometimes several architectures might work well
    c. Design in such a way that you can easily move between architectures if necessary
    d. It often is that case that we are stuck with an architecture based on the environment we are in