UNIT-1

1. Distributive Systems.

2.Tier.

3. J2EE Multi-tier Architecture.

3.1)ClientTier(Presentation Tier)

3.1.1)ClientTier Implementation

3.2)Web Tier

3.2.1)Web Tier Implementation

3.3)EnterPrise JavaBeans Tier(Business Tier)

3.3.1)EJB tier Implementation

3.4)EnterPriseTier(Data Tier/EIS Tier)

3.4.1) EIS Tier Implementation.

4. Introduction to J2SE and J2EE.

5. EnterPrise Application.

6.Client

6.1) Client Input Validation.

6.2)Client Control.

6.3)Client Duplicate Request.

7.Session Management.

7.1) Client-side Session state.

7.1.a) Hidden field

7.1.b) Rewriting URL's

7.1. c) Cookies.

7.2) Server-side Session State.

8. Web Tier and Java Server Pages

8.1)Presentation & Processing.

8.2)Inclusion Strategy.

8.3) Simplify Error Handling.

9) Enterprise JavaBeans Tier.

9.1)Efficient Data Exchange.

9.2) Entity to EJB Relationship

9.3)MVC.

10)Power of Interfaces.

11)Maintainable Classes.

a)Coupling.

b)Cohension.

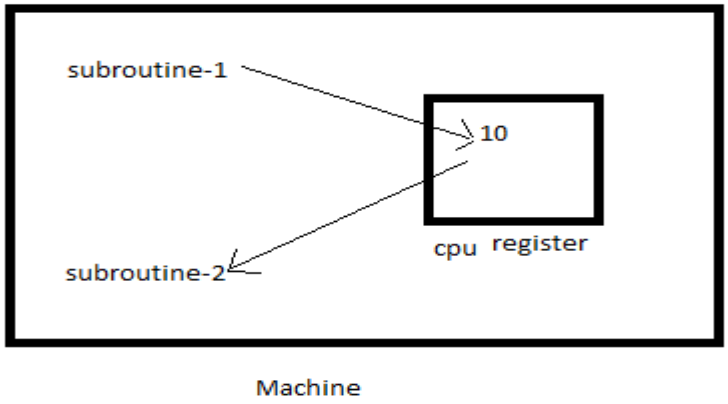
12.Power of Threads.

13. Performance Enhancements.

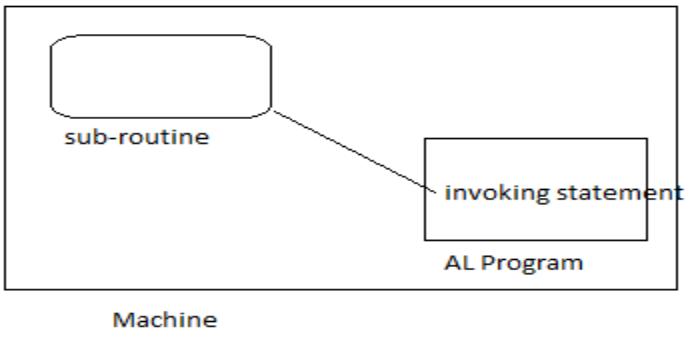
14. Myth of Using Inheritance.

1. Distributive Systems:- The key objective of distirbued systems is to share resources amongs clients.

-> the first evolution in programming languages is assembly language. Software services consists of subroutines written in assembly language that communicate with each other using machine registers,which are memory spaces with in the CPU of machine.



Whenever programmer required functionality provided by software service, the programmer called appropriate assembly language subroutine from with in the program.



Drawback:- The drawback of this is assembly language subroutines were machine specific and could not be easily replicated on different machines. This meant that subroutines had to be rewritten for each machine.

➔ The next evolution is FORTRAN and COBOL . Programs written in fortran could share functionalty by using functions instead of assembly language subroutines.

Advantage:- The functions are not machine specific,So functions could run on different machines by recompiling the function.

At that time there was drawback with data exchange that is magnetic tapes were used to transfer data ,programs and software services to another machine. There was not real-time transmission system.

The real-time transmission came about with introduction of unix operating system. The unix operating system contains support for TCP/IP Protocol.

RPC(Remote procedure call) defined way to share functions written in any procedural language such as FORTRAN,COBOL and C language. This mean that software services were no longer limited to machine.

➔ The next evolutionary step in programming language is object-oriented languages such as C++ and java.Programs written in object-oriented language were organized into software objects not by functionality. A software object is software service that can be used by program. A new protocol was needed that could naturally call software objects. The protocols were developed to access software objects. These were Common object request broker architecture(CORBA), and Distributed common object model(DCOM).

The next evolution of software services was born and was called web services. There new standards were developed with introduction of web services. These were

a) WSDL(web services Broker Architecture):- The programmers use WSDL to publish their web service,there by making the web service available to other programmers over the network.
b) UDDI(Universal Desctiption Discover and integration):- The programmer uses UDDI to locate web services that have been published.
c) SOAP(Service oriented Arichitectural protocol):- The programmer uses SOAP to invoke a particular web service that have been published.

2.Tier:-

Def1:- The tier consists of Resources/Components.

Def2:- The functionality of Application is separated into areas. Each area is said to be Tier.

Def3:- The tier is abstract concept that defines group of technologies that provide one or more services to clients.

Example:-

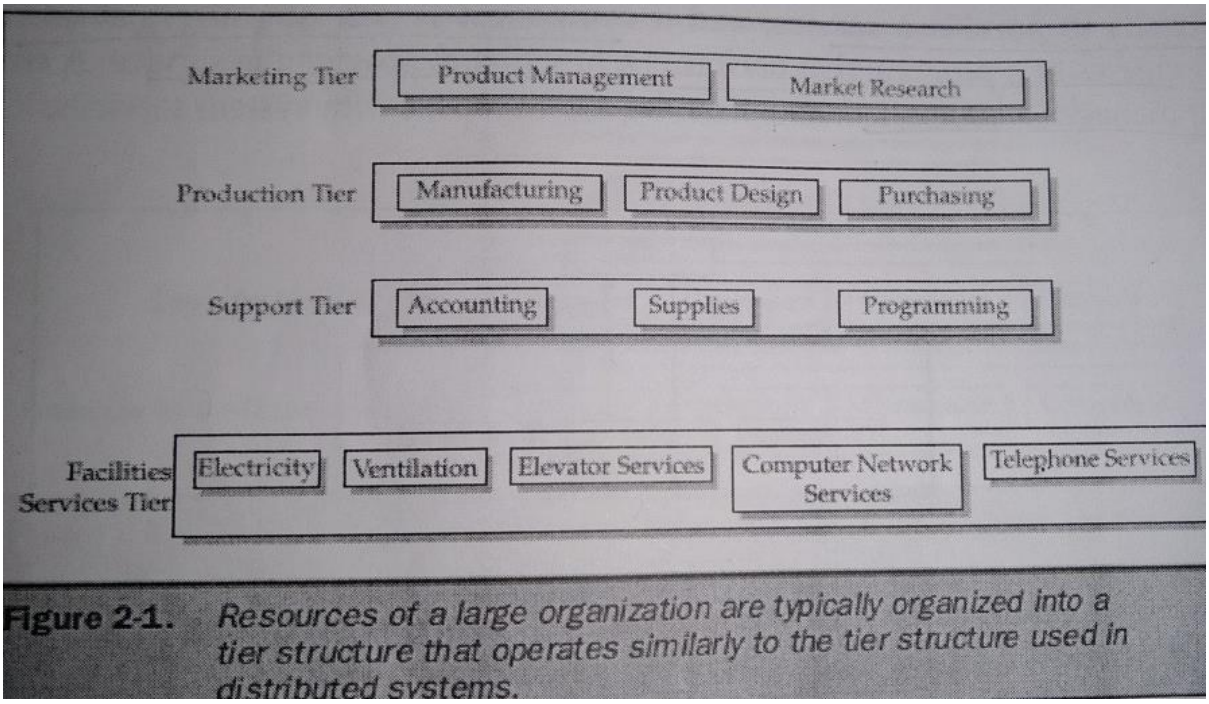


Figure 2-1. *Resources of a large organization are typically organized into a tier structure that operates similarly to the tier structure used in distributed systems.*

At lowest level of corporation are facilities services tier that consist of resources necessary to maintain the office building .

The next tier in organization contains support resources such as accounting ,supplies, compuer programming That support main activity of the company.

Above tier is production tier. The production tier has resources necessary to produce producdts.

Above tier is marking tier. This tier consists of resources used to determine products and services to sell to customers.

Client:- Any resource is considered a client when resource sends request for service to service provider(also referred to as service).

Service:- A service is any resource that receives and fulfills a request from client and that resource itself might have to make request to other resources to fulfill client's request.

Example to client and service:- the product manager requests an accountant to conduct formal cost analysis of manufacturing widget. The product manager is client and accountant is service.

3.) J2EE Multi-tier Architecture:- The J2EE is four-tier architecture. These consist of

3.1)ClientTier(Presentation Tier)

3.2)Web Tier

3.3)EnterPrise JavaBeans Tier(Business Tier)

3.4)EnterPriseTier(Data Tier)

Each tier providing a specific type of functionality to an application. Two or more tiers can reside on same JVM.

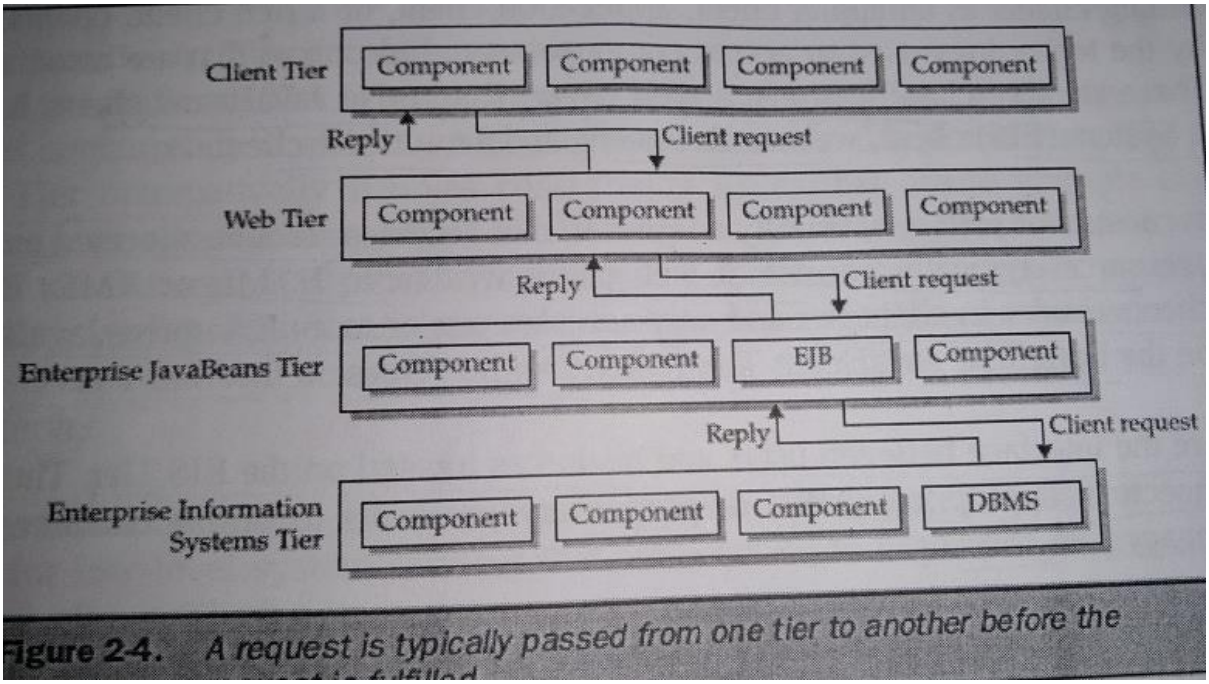


Figure 2-4. A request is typically passed from one tier to another before the

3.1) Client Tier:- Client tier consists of programs that interact with user.

- The programs prompt user for input.
- The programs convert user's response into requests.
- The programs forwarded request to components that can operate on any tier.

- The programs also translate servers response into text and screens that are present to user.

3.1.1)Client Tier Implementation:- Usually the client components are

a)applet Client:- The applet is client application that executes in JVM installed in web browser.

b)Application Client:- It has graphical user interface created from swing or AWT API, but command-line interface is certainly possible.

c)webclient:- web client is software(usually Browser) . That access the resources located on web tier. These resources are web pages written in HTML and XML.

d) EJB clients:- EJB clients can access one or more enterprise java beans that are located on EJB tier.

e)EIS client:- It is interface between users and resources located on EIS tier.

Note:- All clients are executed at only client machine.

3.2) Web Tier:- The two major components of web tier are

1)servlet

2)JSP

The servlet is java class. The request for servlet contains servlets url and is transmitted from client tier to web tier using HTTP. The servlet is assocated url. The request generates instance of servlet or reuses an existing instance. The servlet generates HTML output stream that is returned to web server. The output stream is dynamic web page. The web server then transmits data to client.

JSP is similar to servlet. JSP is associated with a URL. When we call jsp from client tier. The contains translate JSP into servlet the first time by compiling. The compiled servlet is loaded into memory. Sub sequent calls to jsp cause web server to recall servlet with out translating jsp and compiling result code.

3.2.1) web tier Implementation:- The following Java EE technologies are using in web tier in java EE applications.

a)servlet b)JSP c)EL(Expression Language) d) JSP standard Tag Library.

The web tier also consists of web server. When we install web server on server machine automatically, server software will be available in the form of following modules.

1)Main Server(web server)

2)Web Container.

a)Webserver:- The web server responsibilities are

- listen to client request continuously.
- Take the client generated HTTP request.
- Provides container software to execute component.
- Gather output generated by web component.
- Passes the output to client tier as http response.

b) web container:- It manages execution of web components(i.e) it implements life cycle of servlets.

3.3) EJB Tier:- The EJB is class that contains business logic (usually including db access) and callable from web tier and client tier . This tier contains EJB's.

Note:- The EJB components are run at server Machine.

3.3.1) EJB Tier Implementation:- This tier includes one or more EJB servers. The EJB server includes EJB container. The EJB server And EJB container together provides low-level system services(system services). The services are

a)Resouce pooling
b)Transaction Management
c) Fault-Tolerance
d)Concurrency
e) Security ...etc

This tier provides concurrency, fault tolerance, life cycle management. This tier automatically handles concurrency issues that assure multiple clients have simultaneously access same component.

Some vendors include features that provide fault-tolerance operation by making it possible to have multiple EJB servers available through the tier. This means back up enterprise servers can be contacted immediately upon failure of primary EJB server.

3.4)EIS Tier and Its Implemenation:- This layer consists of DBMS server,Legacy Systems(Main frames),off-shelf software. These resources are located on separate machine than java EE server, and are accessed by components on business tiers, client tier. This tier provides connectivity to those resources.

4. Introduction to J2SE and J2EE:- SUN micro systems divided java concepts into 3 categories to support all three types of domains mobile,desktop and internet applications. In java a category is called as Edition. So we can say concepts are divided into 3 editions.

a)Java ME(Micro Edition)

b)Java SE(standard Edition)

c)Java EE(Enterprise Edition)

4.1) J2SE:- Java standard Edition is used for developing the Desktop based Applications .Java SE has both concepts(datatypes,operators,control statements,oops,string handling...etc) and technologies(JSBC,applet,..etc).

4.2)J2EE:- java Enterprise Edition is used for developing the Internet Applictons and Enterprise Applications. It has only technologies(servlet,jsp,EJB,Webservices,JSF...etc).

5. EnterPrise Application:-

Appplication used by more than one person to conduct business could be considered as EnterPrise application. Corporate users having high expectations from enterprise application. They want enterprise application to

- Be available 24 hours a day.
- Having an acceptable response time even in the face of increasing usage.
- Have flexibility to be modified quickly without requiring a redesign of the application.
- Be able to interact with existing system.

6.Clients:- The software working on client tier has several functions,many of which are easily developed by programmers. However, there are few functions that pose a challenge to programmers. These functions are to

- Collect & validate information from user
- Control the client's access to resources.
- Prevent clients from sending duplicate requests.

6.1) Client Input Validation:- Input Validation means checking the accuracy and quality of data before using and processing data.

The developer can implement 3 kinds of validation strategies:

a)syntactic:- syntactic validation determines if the information is well-formed.

b)Lexical:- lexical validation determines if the information is correct data type.

c)semantic:- semantic validation determines if the information is with in range.

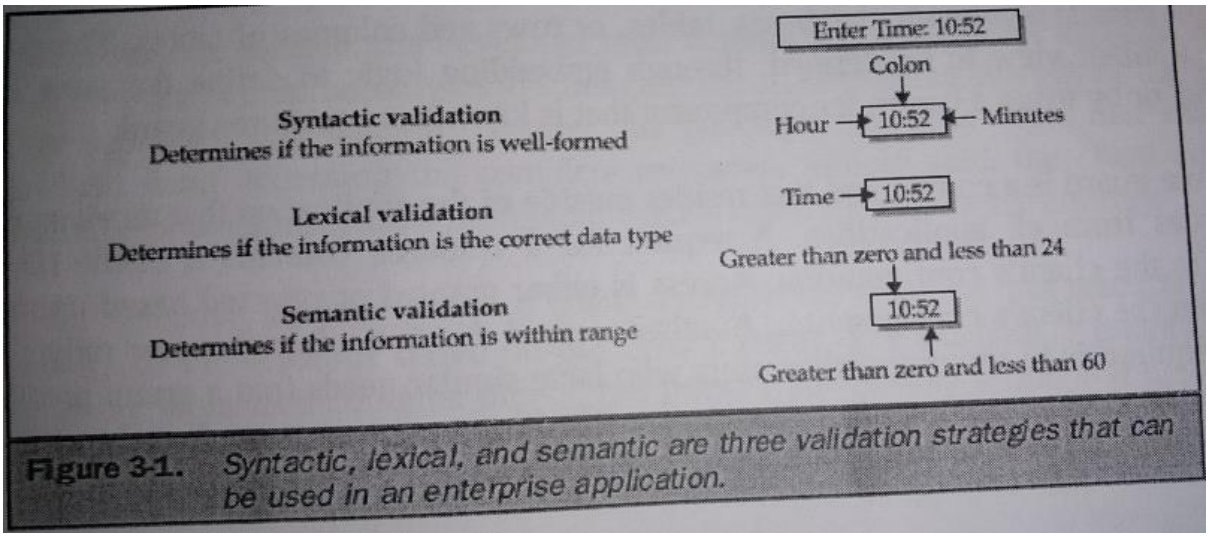


Figure 3-1. *Syntactic, lexical, and semantic are three validation strategies that can be used in an enterprise application.*

The developer has two places where validation can occur:

1)on the Client side.

2)On server side

Avoid the Validating on client side because client-side validation routines frequently must be updated when a new version of browser is released. The best practice when developing a validation strategy is to perform as much of validation on server side.

Another best practice is to reduce the opportunity for user to enter incorrect information by using presentation elements that limits choices. These elements include radio buttons,checkboxes,..etc.

6.2)Client Control:- The scope of resources a client can access is commonly referred to as client view.

A client view consists of databases,tables , or rows and columns of tables. Once client view is defined , the developer must determine a strategy for implementing the client view. There are two commonly used strategies:

a)All-or-nothing strategy:- The all-or-nothing strategy requires the developer to write logic that enables or prevents client from accessing the complete resource.

b) selective strategy:- The selective strategy grants a client access to resource, but restricts access to selected features of resource based on client needs.

Example:- client does have rights to insert new record or modify existing record. Client can read the record.

The best practice is to use selective strategy because selective strategy can be used to provide same features as provided by all-or-nothing strategy.

The client view logic is embedded in web application.(or) The client view logic is implemented as resource guard(EJB component).

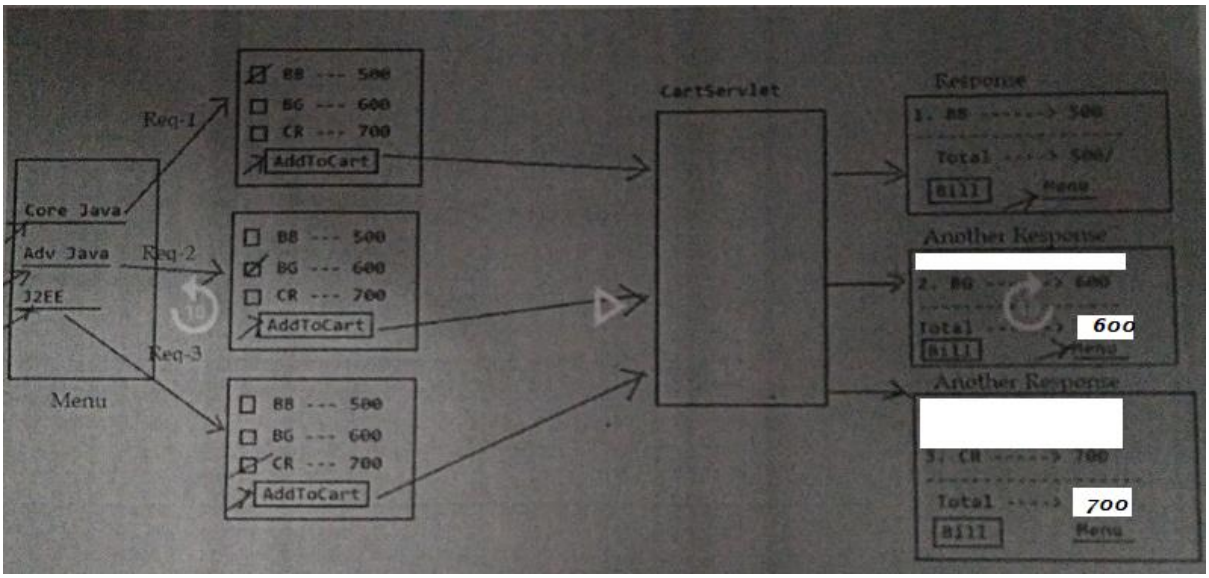
The best practise to use when a resource becomes sharable is to remove embedded logic that defines the client logic that defines client view from application and place the logic into resource guard.

6.3)Duplicate Client Request:- The client(browser) contains that can lead to duplicate request being sent. Namely Back and Stop buttons. The back button causes browser to recall previously displayed web page from the web server. The stop button halts the implementation of request. This means the browser processes some,but not all, so it displays a partial page.

The best practice is to flush session explicitly and updating the session object without waiting for page to complete. The session is still available and checks can be made on server to see if the form was previously submitted.

7.Session Management:- The session is time duration. The session begins with an initial request for service and ends once the client no longer requests services. During the session, client and components exchange information, which is called session state. The client id, client profile or choices client makes in form can be session state.

A component whose sole purpose is to receive information from client , process that information, and return information to client when necessary. Information used by component is retained until the request from client is fulfilled. Afterwards, Information is destroyed. Component lacks persistence . persistence is inherent capability of retaining information between requests.



Therefore session state has to be maintained until session is completed.

There are two common ways to manage session state:

- on the client side
- on server side on the enterprise javabeans tier.

7.1) Client-side Session state:- session state can be maintained at client side using three techniques.

a) By Hidden field

b) By Rewriting URL's

c) by Cookies.

A) Hidden Field:- The component can include in the HTML form field that is not displayed on the form. This field is called hidden field. A hidden field is similar to other fields on form in that hidden field can hold a value. However, this value is assigned to hidden field by component rather than by user.

Example:-

<input type='hidden' name='accountnumber' value='1234'>

The best practice for using hidden field to maintain session state is to do so only when small amounts of string information need to be retained.

Drawback:- Hidden field contains only string values and not numeric ,dates and other data types.

B)URL Rewriting:- URL is element with in web page that uniquely identifies a component and is used by browser to request a service. In this technique developer will append parameter value name and parameter value to URL. When the user clicks on this URL, it is sent to server. The server will read the parameter names and parameter values present in the URL and used them as session data.

Example:-

 Click to place new order

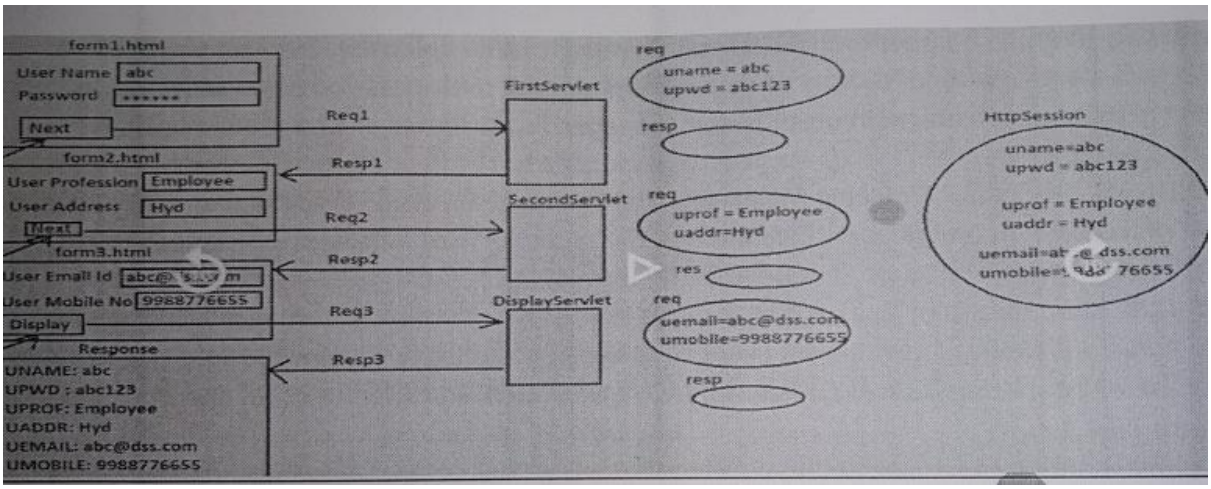
c)cookie:- The cookie is small object . It contains single data in the form of name and value pair. The cookie is created at server side. But cookie is maintained and managed by client machine. Cookie can be used to store session state.

Drawbacks:-

a) Cookies contains minimum data(maximum of 4KB).
b) First cookies can be disabled, there by prohibiting the enterprise application from using cookie to manage session state.

7.2) ServerSideSession State:- The session state is lost if client machine fails. An alternative to maintaining session state on client side is to store session state on server.

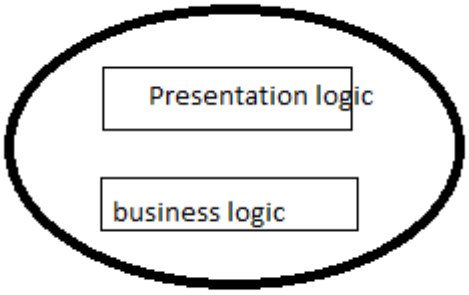
The best practice is to maintain session state on EJB tier using Enterprise java Beans or on web Tier using HttpSession Interface.For each and every user, we create HttpSession object. The session data is placed in Httpsession object for the sake of future reusability. The session state automatically becomes invalid and removed when the session ends.



The best practice is to always set session timeout,which automatically invalidates session state after time has passed and session state has not been accessed. Session state is removed.

8)Web Tier and Java Server Pages:-

8.1)presentation and processing:-



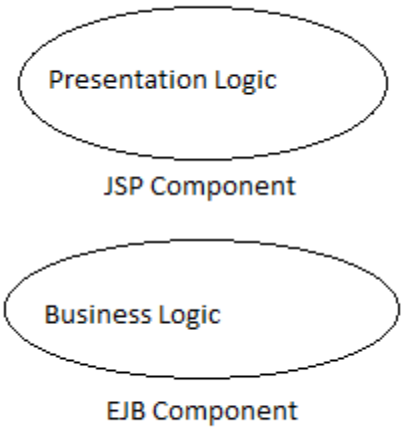
JSP Component

Placing both presentation and processing logic components in same component lead to non maintainable code.This is because of two reasons.

First Reason:- presentation and processing logic components tend to become complex and difficult to understand.

SecondReason:- Programmer who is proficient in HTML writes presentation component. A java programmer writes processing logic component. This means that two programmers must work on same JSP program which can be in efficient.

The best practice for writing jsp program is to separate the presentation code and processing code.place presentation code in jsp program and place processing code in EJB.



8.2)Inclusion Strategy:- If the same user interfaces code can appear in more than one web page, which is inefficient.

The best practice to avoid redundant code is either include directive or include action.

8.2.1) Include Directive:- The include directive places the called JSP program into the Calling JSP program.

8.2.2)include Action:- The include directive places the Results generated by called JSP program into calling JSP program.

The best practice is use the include directive whenever variables are used in calling JSP program. The best practice is use the include action whenever varaiable values are used in callning JSP program.

8.3) Simplify Error Handling:- It is important that the application translate raw error messages into text that is understood by user of application. Otherwise, the error message is likely to confuse.

Although errors can occur throughout application, a client should present to user errors generated by jsp program or by servlet. Once error is trapped , the EJB forwards error message to jsp program. The jsp program translates error message into message easily understood by user and then displays translated error message in dynamically generated web page.

9) Enterprise JavaBeans Tier:- The EJB provides business logic. The processing logic includes all code and data that is necessary to implement one or more business rules.

The best practice is make each enterprise javabeans self-contained and minimize the interdependence of enterprise java beans where possible. That is, avoid having trail of enterprise javabeans calling each other.

9.1) Efficient Data Exchange:- JSP program and EJB frequently exchange information while enterprise application is executing. There are two common ways in which information is exchanged between jsp program and ejb. These are by individually sending and receiving each data element or by using value object to transfer data in bulk.

The best practice when exchanging information between jsp program and ejb is to use a value object. In this way, there is less stress on the network than sending individual data.

9.2) Entity to EJB Relationship:- Developers creates one-to-one relationship between entities defined in an applications entity relationship diagram and with enterprise javabeans. That is each entity has its own entity EJB that contains all processing logic required by the entity.

Drawback:- creating one-t0-one relationship , as such tends to generate many enterprise java beans and therefore is likely to increase overhead, which results in performance impact.

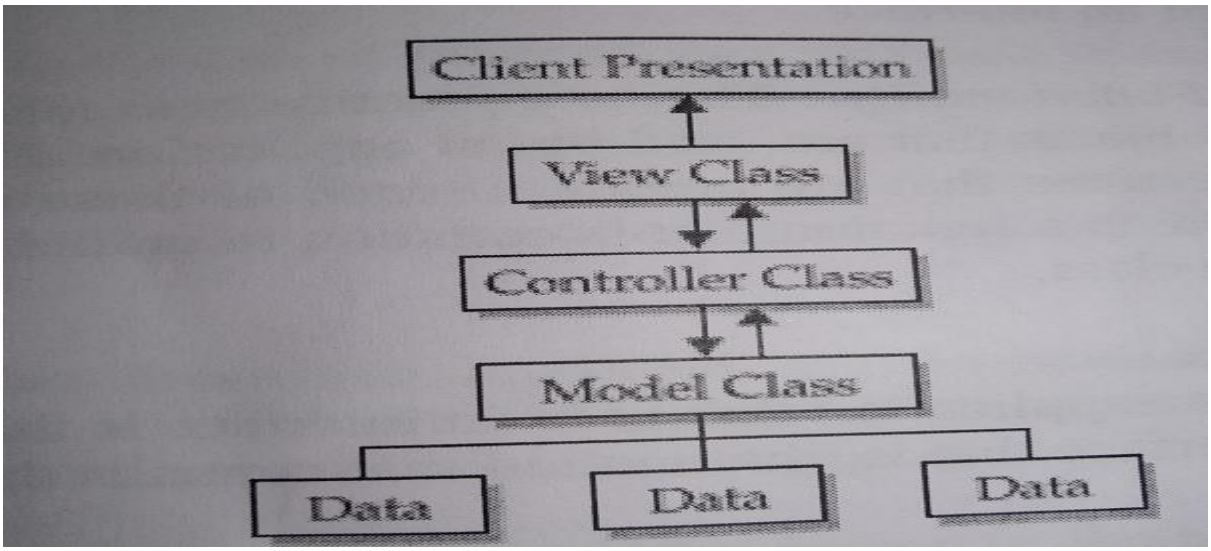
The best practise when translating ER diagram into EJBs is to consolidate related processes that are associated with several entites ito one session enterprise java beans.

9.3) MVC:- MVC strategy basically divides applications into three broad components. These are the

a) model class:- It consists of components(EJBs) that control data used by application.

b) view class:- It is composed of components(Servlets,JSP) that present data to client.

c) control class:- It is responsible for event handling and co-ordinating activities between model and view class.(session Enterprise java beans).



10.Power of Interface:- An interface is collection of method signatures. A method signature consists of name and number and type of parameter for a method.

Syntax: inferfacename(datatype1,datatype2,...)

The best practice is to create an interface whenever an application contains common behaviours where algorithms used to process the behaviors differ.

Example:- an acceleration interface provides acceleration functionality to any real-world object regardless if the real-word object is motor vehicle,aircrart. In this way, un related real-world objects that have same functionality can share same data and methods of interface.

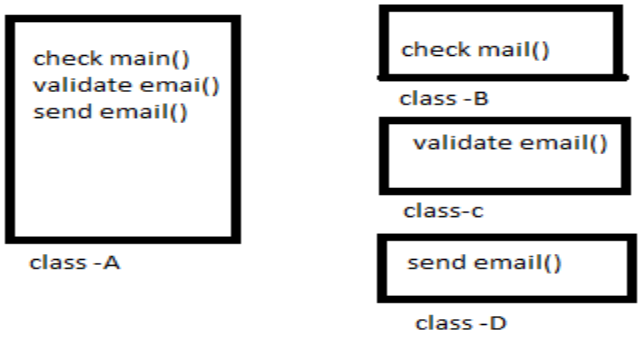
11. Maintainable classes:- There are two factors determine if classes are maintainable. These are

a)coupling
b) cohesion.

Coupling:- coupling occurs when there is class dependent on another class. Before change can be made to either the derived class or base class, developer must asses impact on coupled class. Changes to base class might negatively impact a derived class. Changes to derived class won't affect the base class, but could inadvertely modify functionality inherited from base class. In either scenario, additional precautions must be taken by developer.

Cohension describes how well class functionality is focused.

Example:-



Class A has low cohension than all other classes.

The best practice is design an enterprise application with highly cohesive classes and minimum coupling.

12. Power of Threads:- A thread is stream of executing program statements. More than one thread can be executed with in enterprise application. This means that multiple statements can run parallel. There are several best practices to employ when using threads in enterprise application. Avoid using threads unless multiple processes require access to same resources concurrently. This is because using multiple threads incurs processing overhead that can actually decrease response time.

Keep the number of threads to minimum; otherwise you will notice a gradual performance degradation. If your application experiences a decrease in performance,prioritize threads. Assign a higher propriety to critical processes.

13. Performance Enhancements:- Although bytecode is optimized, bytecode still needs to interpreted by java virtual machine(JVM). It is this overhead that detracts from applications performance. There are two ways to reduce or practically eliminate amount of byte code that is interpreted at runtime:

By using hotspot, new in the JIT(just in time compiler) or by using native compiler.

A new hybrid strategy is developing to increase performance of java application where an application is divided into static and dynamic modules. Static modules such as enterprise java beans container are compiled into native libraries that are executables and dynamic modules such as Enterprise java beans are compiled into byte code. Only dynamic modules are optimized at runtime.

14. Myth of using inheritance:- There are two kinds of classes used in inheritance:

1)Base class.

2) Derived class.

The base class contains methods and data some or all of which are inherited by derived class. An object of the derived class has access to some or all of the data and methods of base class and all the data and methods of derived class.

Example:-

Base class is motor vehicle. It has engine,wheels,body and among other components. An automobile and truck are two kinds of motor vehicles. They are derived classes and inherit an engine,wheels body and other components from motor vehicle class.

The relationship between base class and derived class is referred to as coupling.

The best practice when translating an entity relationship diagram to an application class model is to use an interface rather than use coupling , where possible.

14.1)Problems with Inheritance:- The problem is ripple effect. The ripple effect occurs whenever a change is made to base class. Changes to base class ripple down to all the derived classes and might negatively impact the implementation of attributes and functionality of derived class.This means that the developer who is responsible for maintaining a base class must examine impact any change in base class has on derived classes before making the change.

The best practice is to minimize the use of inheritance in enterprise application. Only use inheritance when there is commonality among objets that is not functionality.