Centralized Architecture
Two Tier Application Architecture

- In the 90’s, systems should be *client-server*
Client-Server Architecture

- Client-Server divides programs into two types
- **Server** – manages information system resources or provides well defined services for client
- **Client** – communicates with server to request resources or services
- **Advantage** – Deployment flexibility
  - scalability, maintainability
- **Disadvantage** – Potential performance, security, and reliability issues from network communication
3 Layer Architecture
Three-Layer Architecture

- Layers can reside on one processor or be distributed to multiple processors

- **Data layer** – manages stored data in databases

- **Business logic layer** – implements rules and procedures of business processing

- **Presentation (View) layer** – accepts user input and formats and displays processing results
Web Architecture

DHTML

Browser (IE, FireFox, Opera)

Desktop (PC or MAC)

HTTP

PHP script

Web Server (Apache, IIS)

SQL

data

Database

Database Server

Presentation
HTML+DHTML

Dynamic HTML Generation, Business Logic

Persistence
Enterprise Application Model
Internet and Web-based Application Architecture

- Web is complex example of client-server architecture
- Can use Web protocols and browsers as application interfaces
- Benefits
  - Accessibility
  - Low-cost communication
  - Widely implemented standards
Negative Aspects of Internet Application Delivery

- security
- Fluctuating reliability of network throughput, slow speed
- Volatile, changing standards
Presentation layer (Client Side)

- Issue request to remote server using appropriate protocol (usually HTTP)
- Accept and render the returned HTML (or JPEG, ..) file
- Allow plug-ins to handle new file types
- execute client-side scripts in JavaScript
- Accept user input via a variety of controls on a form
Application Layer: Server Side Presentation + Business Logic

- Server (Apache, IIS)
  - Identifying appropriate action to take – fetch a file
  - Sending output back to caller
  - Support for:
    - thousands of concurrent users
    - multi-threading [allow multiple processes to run concurrently]
    - caching [holding results in a temporary store to reduce re-calculation]

- Server Script (e.g. in PHP)
  - Interacting with the server (accessing input and generating output)
  - Interpreting the requests according to business rules
  - Requesting the appropriate data from the Persistence layer
  - Computing derived data
  - Creating the HTML (or GIF, ...) for the page
Persistence layer

- interaction with the database using standard languages e.g. SQL queries using database-specific protocol over TCP/IP
- define and modify the data structures (e.g. tables)
- insert, update and delete data
- maintain data persistently, with backup and recovery
- handle transactions to support **concurrent** access to the database via locking
- optimise access by compilation of queries, indexing, replication of tables etc.
Enterprise Application Model
J2EE clients

- **Web clients**
  - Dynamic web pages with HTML, rendered by web browsers.
  - Can include applets.
  - Communicates with server typically using HTTP.

- **Application clients**
  - User interface using GUI components such as Swing and AWT.
  - Directly accesses the business logic tier.
Web-tier Components

- Client can communicate with the business tier either directly or through servlets or JSP that are located in the web-tier.
- Servlets are special classes to realize the request-response model (get, post of HTTP).
- JSP is a developer-friendly wrapper over the servlet classes.
Business-tier Components

- Business logic, rules, computation
- Enterprise Java Beans (EJB) can be used to implement this tier.
- This tier receives the data from the web-tier and processes the data and sends it to the EIS-tier and takes the data from the EIS and sends it to the web-tier.
Enterprise Information System (EIS) Tier

- In general this corresponds to the database (relational database) and other information management system.
- The other information management systems may include Enterprise Resource Planning (ERP) and legacy system connected through open database connectivity.
Enterprise Java Bean (EJB) (1)

- EJBs are *distributed components* used to implement business logic (no UI)

- Client of EJBs can be JSPs, servlets, other EJBs and external applications

- Clients see *interfaces*
At run-time, an enterprise bean resides in an EJB container.

An EJB container provides services such as security, transaction, deployment, …

Process of installing an EJB in a container is called EJB deployment.
Enterprise Java Bean (EJB) (3)

- **Entity Beans**
  - Represent persistent business Entity
  - Persisted in storage system (usually Database)

- **Session Beans**
  - Encapsulate complex business logic
  - Can coordinate transactional work on multiple entity beans
  - Stateful/Stateless

- **Message Driven beans**
JMS

- Point-to-point
  - Destination is “queue”
Publish-subscribe

Destination is “topic”
What is Application Server

- Application servers enable the development of multi-tiered distributed applications. They are also called “middleware”

- An application server acts as the interface between the database(s), the web servers and the client browsers
J2EE Application Servers

- Major J2EE products:
  - BEA WebLogic
  - IBM WebSphere
  - Sun iPlanet Application Server
  - Oracle 9iAS
  - HP/Bluestone Total-e-Server
  - Borland AppServer
  - Jboss (free open source)
  - ...
J2EE Benefits

- Component based model
- Container provided services
- Highly Scaleable
- Simplified Architecture
- Flexible security model
- Integration with existing systems
- Freedom to choose vendors of application servers, tools, components
- Multi-platform
J2EE Application Scenarios

- Multi-tier typical application

Browser

HTML, HTTP, XML

Web Container

JSP Pages, Servlets, XML, JavaMail

EJB Container

Enterprise Beans, JMS, JDBC (or connectors)

EIS Resources
J2EE Application Scenarios

- Stand-alone client
J2EE Application Scenarios

- Web-centric application
J2EE Application Scenarios

- Business-to-business
J2EE Services and APIs

- **JNDI - Naming and directory services**
  - Applications use JNDI to locate objects, such as environment entries, EJBs, datasources, message queues

- **Transaction service:**
  - Controls transactions automatically

- **Security**

- **J2EE Connector Architecture**
  - Integration to non-J2EE systems, such as mainframes and ERPs.
  - Standard API to access different EIS
.NET Developer Tools

- Visual Studio.NET is an integrated development environment for developing .NET applications

- It includes support for multiple languages
  - Visual Basic.NET
  - Visual C#.NET
  - Visual C++.NET
  - ASP.NET
.NET High Level Description

- Common Language Runtime

- Platform Interoperability
  - "You very seldom want to port an existing application between platforms, but you do want to interoperate between applications regardless of platform; therefore, industry standard support for Web Services is of great importance."
  - Dan Fox, Solutech, Inc. Author of: *Building Distributed Applications with Visual Basic.NET*
ASPX

- Microsoft Active Server Pages, .NET
  - IIS Server

- Old ASP
  - HTML with embedded server scripts
  - Session Support
  - VB Scripts

- ASPX.NET
  - Separation of HTML and code
  - Multiple Languages
IIS

- **Internet Information Services** (IIS, sometimes **Server** or **System**) is a set of Internet-based services for servers using MS Windows.

- Used for corporate, commerce and secure websites.

- It is integrated with Windows 2000 and Windows 2003 Server.
Forrester Report: The State of Technology Adoption

Source: [http://download.microsoft.com/download/c/7/5/c75837dc-90bb-44d8-ae70-db7bcc5980b9/TheStateofTechnologyAdoption.pdf](http://download.microsoft.com/download/c/7/5/c75837dc-90bb-44d8-ae70-db7bcc5980b9/TheStateofTechnologyAdoption.pdf)

### Which one platform will be used for the majority of your development work in 2004?

<table>
<thead>
<tr>
<th>Category</th>
<th>J2EE</th>
<th>.NET</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overall</td>
<td>44%</td>
<td>56%</td>
</tr>
<tr>
<td>Utilities and telecom</td>
<td>65%</td>
<td>35%</td>
</tr>
<tr>
<td>Finance and insurance</td>
<td>56%</td>
<td>44%</td>
</tr>
<tr>
<td>Manufacturing</td>
<td>45%</td>
<td>55%</td>
</tr>
<tr>
<td>Retail and wholesale trade</td>
<td>42%</td>
<td>58%</td>
</tr>
<tr>
<td>Media, entertainment, and leisure</td>
<td>38%</td>
<td>62%</td>
</tr>
<tr>
<td>Business services</td>
<td>36%</td>
<td>64%</td>
</tr>
<tr>
<td>Public sector</td>
<td>35%</td>
<td>65%</td>
</tr>
</tbody>
</table>

**Action**

It’s unrealistic for firms to put all new development on a single platform. Instead, firms should understand the main differences between the platforms, segment their applications into specific domains, and select the appropriate platform for each domain.

Base: 322 software decision-makers at North American companies
Pros and Cons of J2EE

- Pros
  - Portability
  - Vendor choices for tools and application servers
  - Rich developer community, many free tools.

- Cons
  - Complex application development environment
  - Tools can be difficult to use
  - Java Swing limited for developing GUI's (?)
  - Performance
Pros and Cons of .Net

- **Pros**
  - Easy to use tools
  - Strong framework for building GUI's
  - Language support
  - Performance

- **Cons**
  - Portability
  - Choice of IDE's Limited
.Net and J2EE key differences

- Fundamental Philosophy:
  - “One language, Many Systems” vs. “Many Languages, One System.”
Questions
.NET Online Resources

- [http://www.startvbdotnet.com](http://www.startvbdotnet.com)
- [http://www.asp.net](http://www.asp.net)
- [http://www.dotnetwire.com](http://www.dotnetwire.com)
J2EE Online Resources

- [http://java.sun.com/j2ee/index.jsp](http://java.sun.com/j2ee/index.jsp)
- [http://www.javaworld.com](http://www.javaworld.com)
Assignment 2

- Compare .Net and J2EE
- See these papers:
- Analyze the benchmarks and prepare a report.
  - Benchmark approach, fairness, results