BITS Pilani
Pilani Campus

# Network Programming

K Hari Babu
Department of Computer Science & Information Systems

innovate achieve lead

BITS Pilani
Pilani Campus

## Outline

## Outline

- Forms
- CGI
- Fast CGI
- Web Server Extension Interfaces
    - ISAPI
    - Apache Interface

## Support for Dynamic Content

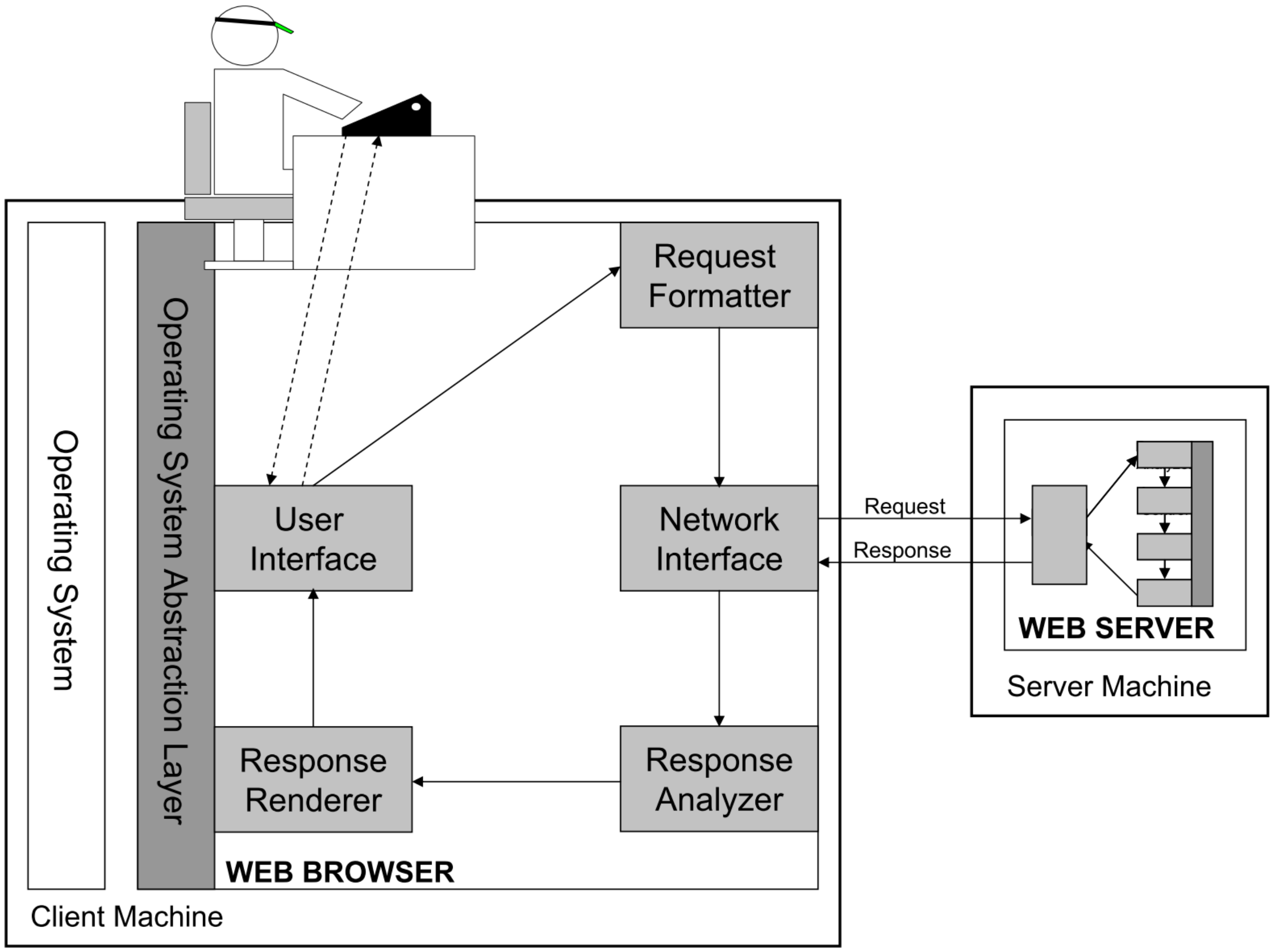
- The HTTP request-response cycle provides an architectural foundation for distributed hypertext applications.
- Web servers and browsers communicate through message-passing, browser initiated requests by URI name for resources (HTML pages, etc.)
- Initial dynamic interaction supported by HTML forms and CGI.

innovate achieve lead

BITS Pilani
Pilani Campus

## Client Side Technologies

# Browser Reference architecture



## HTML

- Display static information on web pages
- Content is written hard coded into HTML file
    - Change content means change HTML file
- No reaction to user actions except for hyperlinks
- Data in forms cannot be processed

## Dynamic HTML

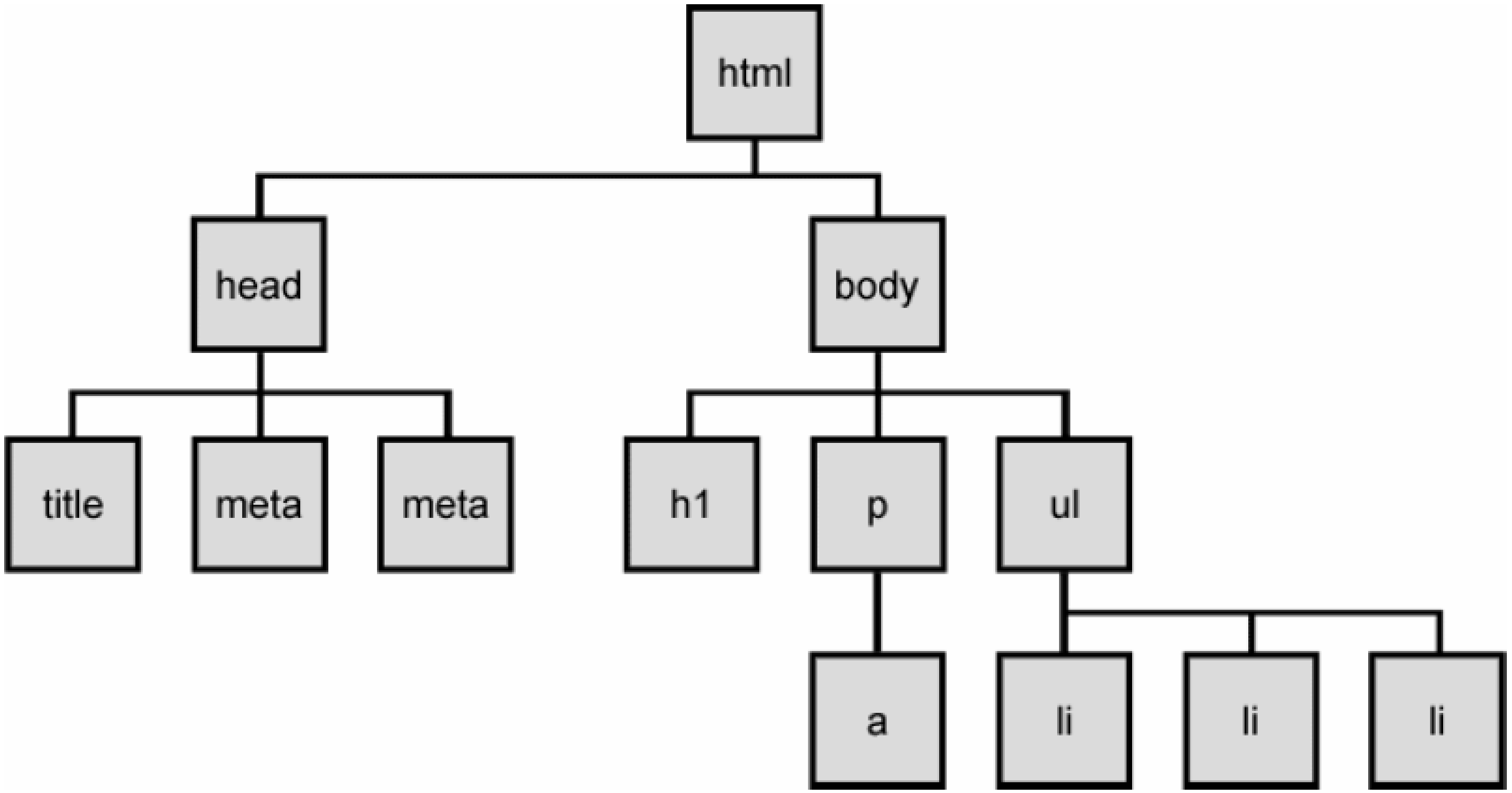
- To add dynamic features and user interactions to web pages we need some way of adding programming logic to HTML pages
- Client-side scripting
  - that detects user actions and responds to it
  - dynamic content based on user actions or choices
  - code that can read and process form information
- Dynamic HTML is the combination of HTML, Client-side scripting and DOM
  - Popular language for client-side scripting is **Java-script**

## Document Object Model (DOM)

- DOM = Document Object Model
  - Defines a hierarchical model of the document structure through which all document elements may be accessed
- Nodes
  - The W3C DOM defines element of a document is a node of a particular type
- Node Types
  - Common types are: document node, element node, text node, attribute node, comment node, document-type node

# The DOM tree

- Example



# Types of DOM nodes

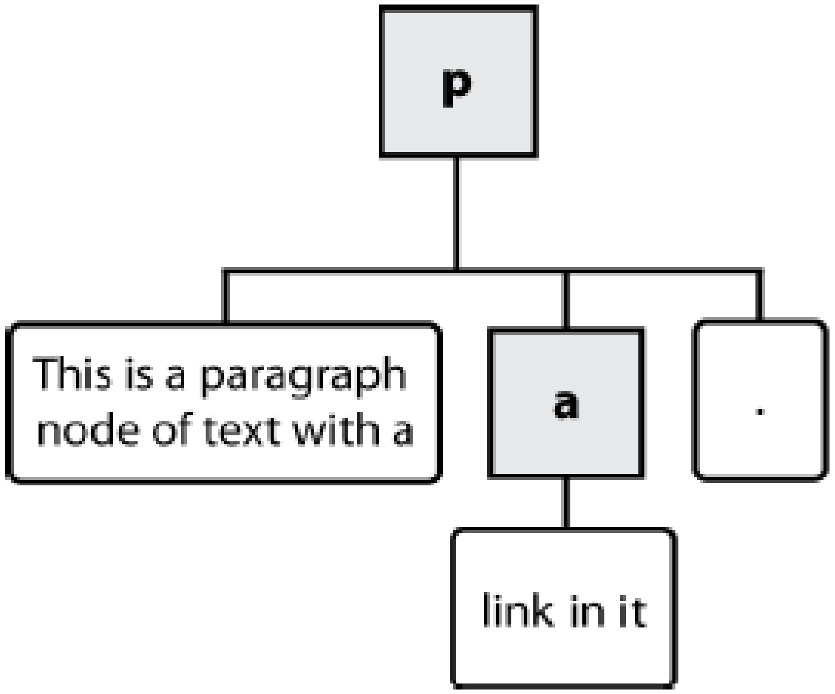
- element nodes (HTML tag)
  - can have children and/or attributes
- text nodes (text in a block element)
- attribute nodes (attribute/value pair)
  - text/attributes are children in an element node
  - cannot have children or attributes
  - not usually shown when drawing the DOM tree

```html
<p>
This is a paragraph of text with a
<a href="/path/page.html">link in it</a>.
</p>
```
*HTML*

## Types of DOM nodes

```html
<p>
This is a paragraph of text with a
<a href="/path/page.html">link in it</a>.
</p>
```

HTML



# Elements vs text nodes

- Q: How many children does the div below have?
- A: 3
  - an element node representing the <p>
  - two text nodes representing "\n\t" (before/after the paragraph)
- Q: How many children does the paragraph have?

```html
<div>
	<p>
		This is a paragraph of text with a
		<a href="page.html">link</a>.
	</p>
</div>
```

*HTML*

# DOM element methods

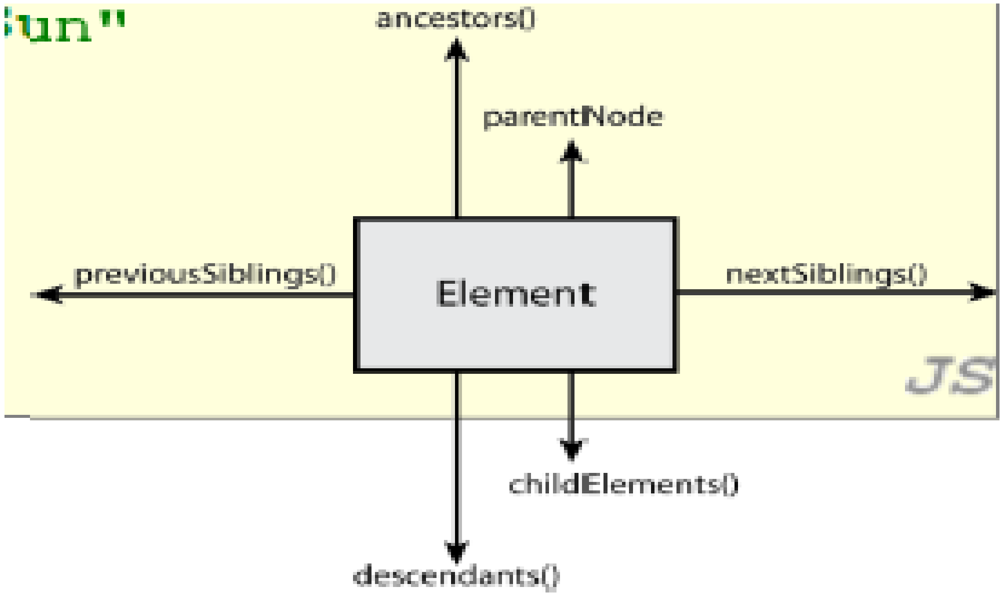
| | | | | |
|---|---|---|---|---|
| absolutize | addClassName | classNames | cleanWhitespace | clonePosition |
| cumulativeOffset | cumulativeScrollOffset | empty | extend | firstDescendant |
| getDimensions | getHeight | getOffsetParent | getStyle | getWidth |
| hasClassName | hide | identify | insert | inspect |
| makeClipping | makePositioned | match | positionedOffset | readAttribute |
| recursivelyCollect | relativize | remove | removeClassName | replace |
| scrollTo | select | setOpacity | setStyle | show |
| toggle | toggleClassName | undoClipping | undoPositioned | update |
| viewportOffset | visible | wrap | writeAttribute | |

## DOM tree traversal methods

| method(s) | description |
|---|---|
| ancestors, up | elements above this one |
| childElements, descendants, down | elements below this one (not text nodes) |
| siblings, next, nextSiblings, previous, previousSiblings, adjacent | elements with same parent as this one (not text nodes) |

## DOM tree traversal methods

```
// alter siblings of "main" that do not contain "Sun"
var sibs = $("main").siblings();
for (var i = 0; i < sibs.length; i++) {
        if (sibs[i].innerHTML.indexOf("Sun") < 0) {
                sibs[i].innerHTML += " Sunshine";
        }
}                                                              JS
```



**The dollar ($) sign is commonly used as a shortcut to the function document.getElementById().**

## Selecting groups of DOM objects

- methods in document and other DOM objects for accessing descendants:

| name | description |
|---|---|
| getElementsByTagName | returns array of descendents with the given tag, such as "div" |
| getElementsByName | returns array of descendants with the given name attribute (mostly useful for accessing form controls) |

## Getting all elements of a certain type

```
var allParas = document.getElementsByTagName("p");
for (var i = 0; i < allParas.length; i++) {
      allParas[i].style.backgroundColor = "yellow";
}                                                          JS
```

```
<body>
      <p>This is the first paragraph</p>
      <p>This is the second paragraph</p>
      <p>You get the idea...</p>
</body>                                                    HTML
```

# Combining with getElementById

```
var addrParas = $("address").getElementsByTagName("p");
for (var i = 0; i < addrParas.length; i++) {
      addrParas[i].style.backgroundColor = "yellow";
}                                                      JS
```

```
<p>This won't be returned!</p>
<div id="address">
      <p>1234 Street</p>
      <p>Atlanta, GA</p>
</div>                                                 HTML
```

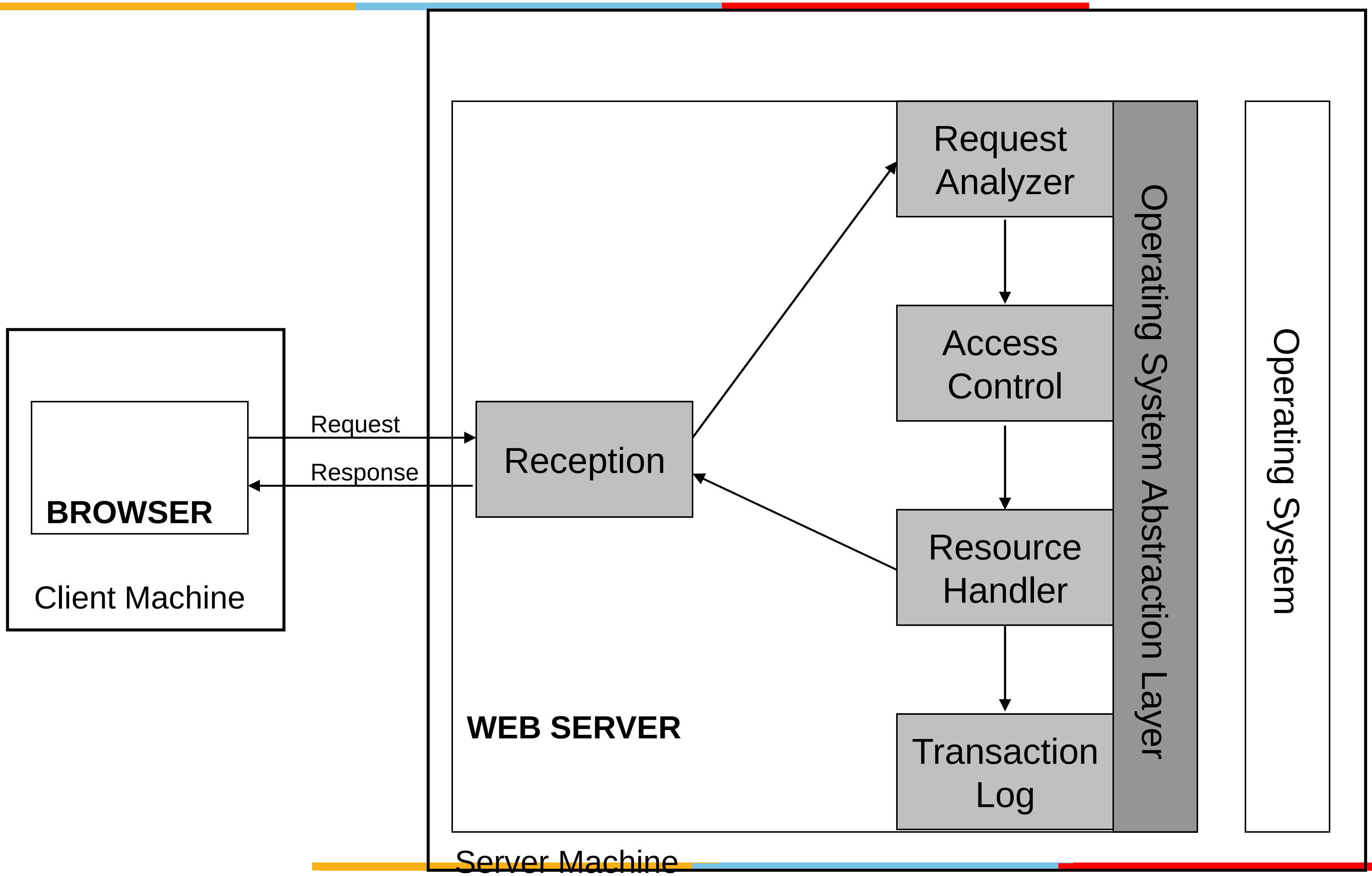
## Other Technologies

- Ajax
- XHTML
- CSS
- XML
- XSLT
- XML-HttpRequest

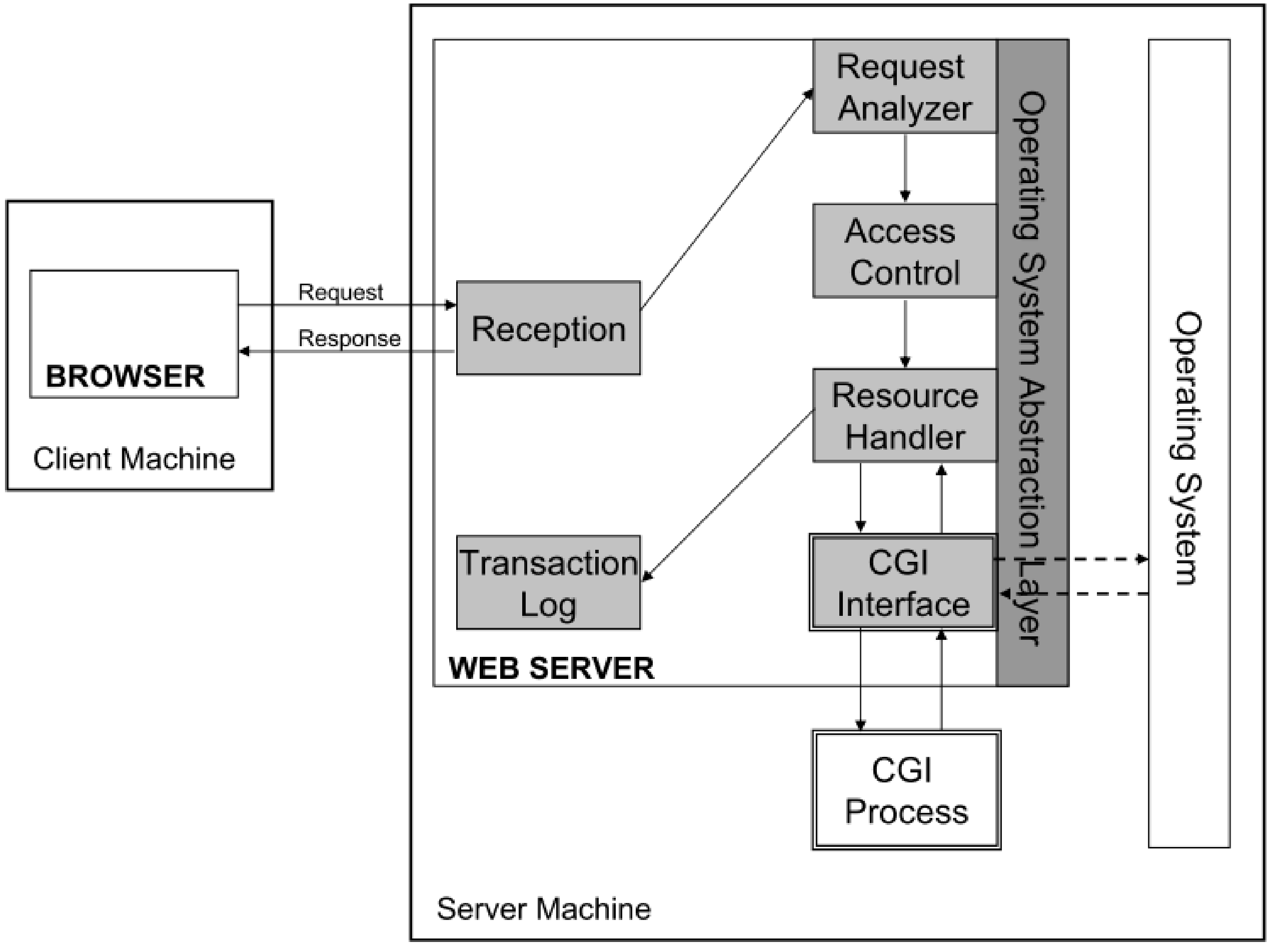
BITS Pilani
Pilani Campus

## Server Side Technologies

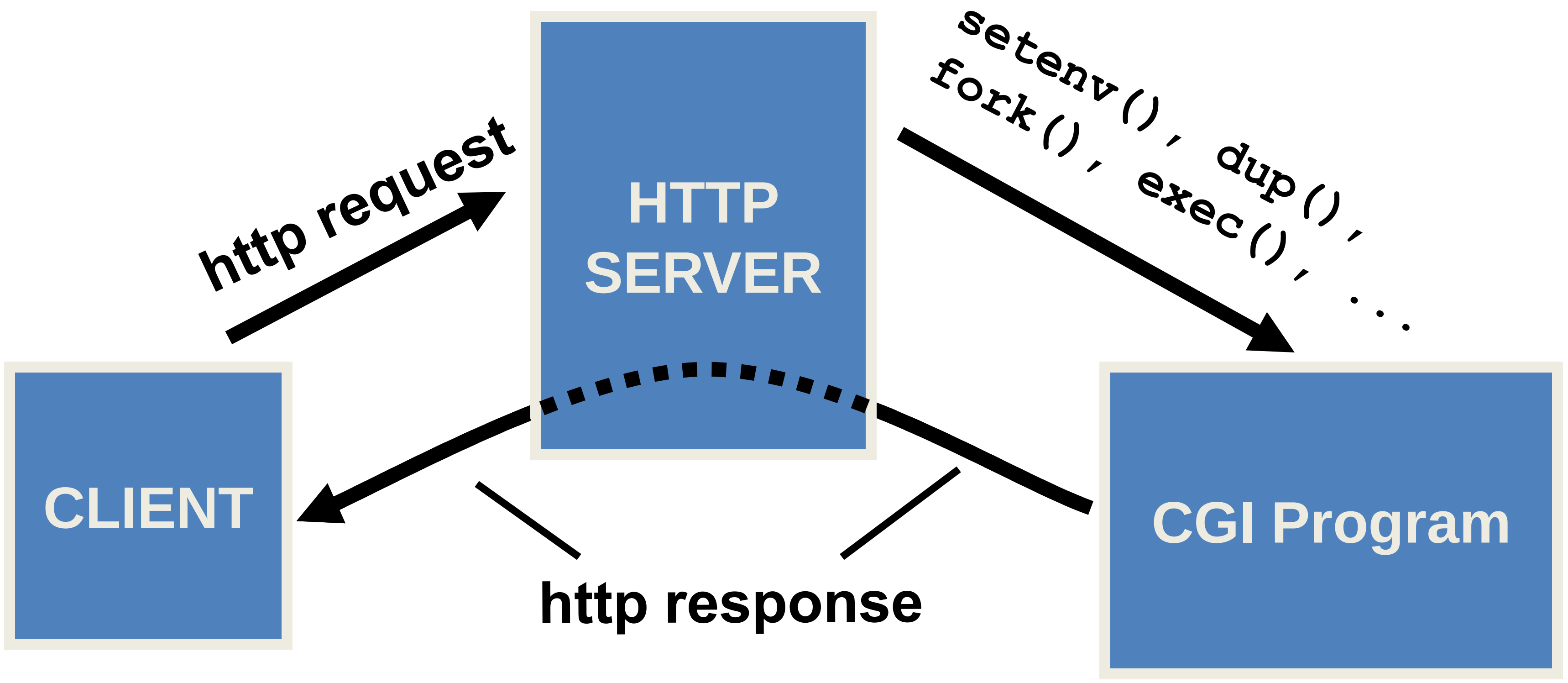
# A Web server reference architecture



# CGI


BROWSER
Client Machine
Request
Response
Reception
Request Analyzer
Access Control
Resource Handler
Transaction Log
CGI Interface
WEB SERVER
Operating System Abstraction Layer
Operating System
CGI Process
Server Machine

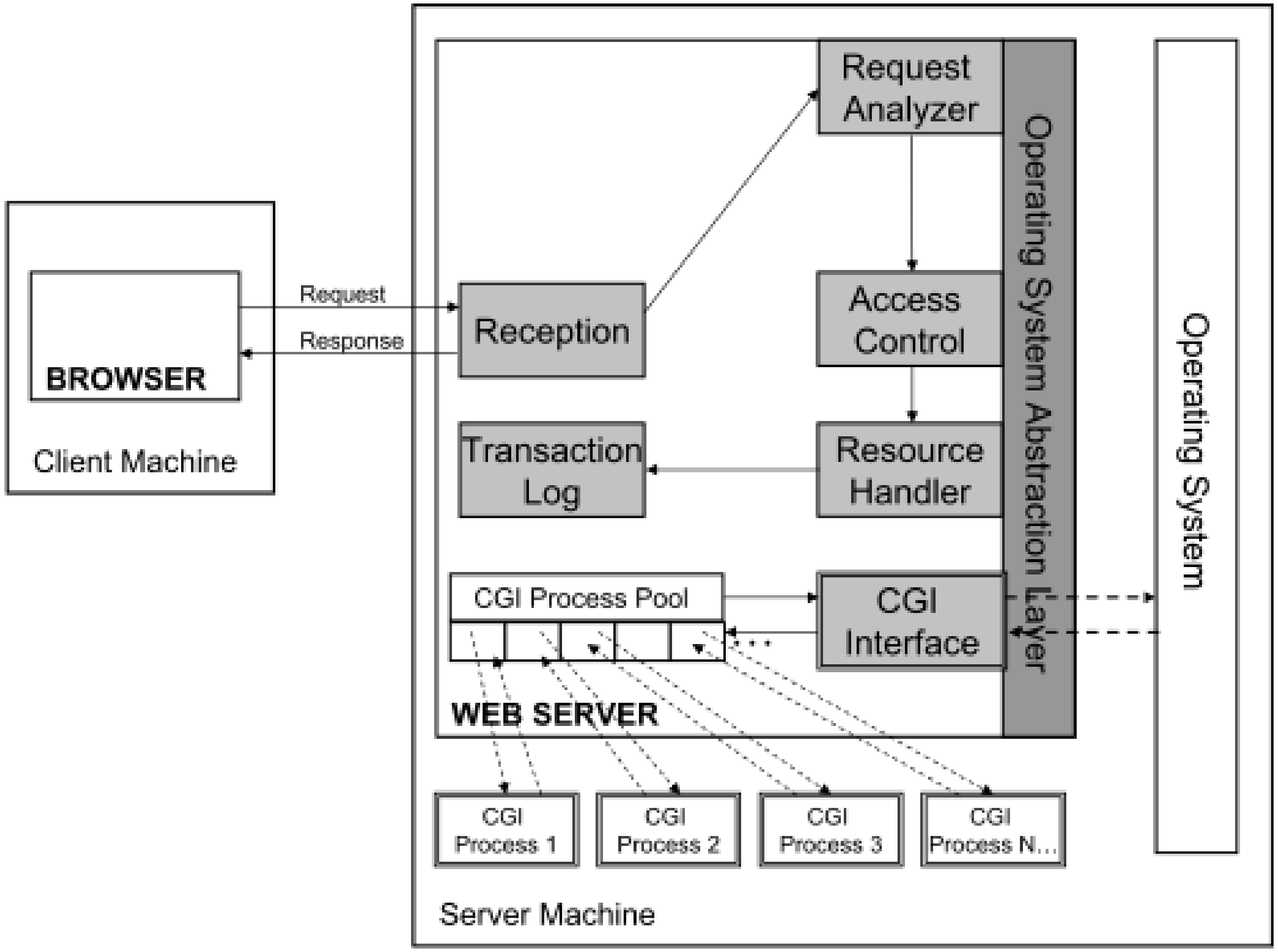
## CGI Programming



## CGI advantages / disadvantages

- Simple, implemented on all well-known Web servers out-of-the-box.
- Combined with scripting languages are a portable solution.
- Not process efficient.
- HTML generation from within code, not providing separation between the HTML designers and programmers.

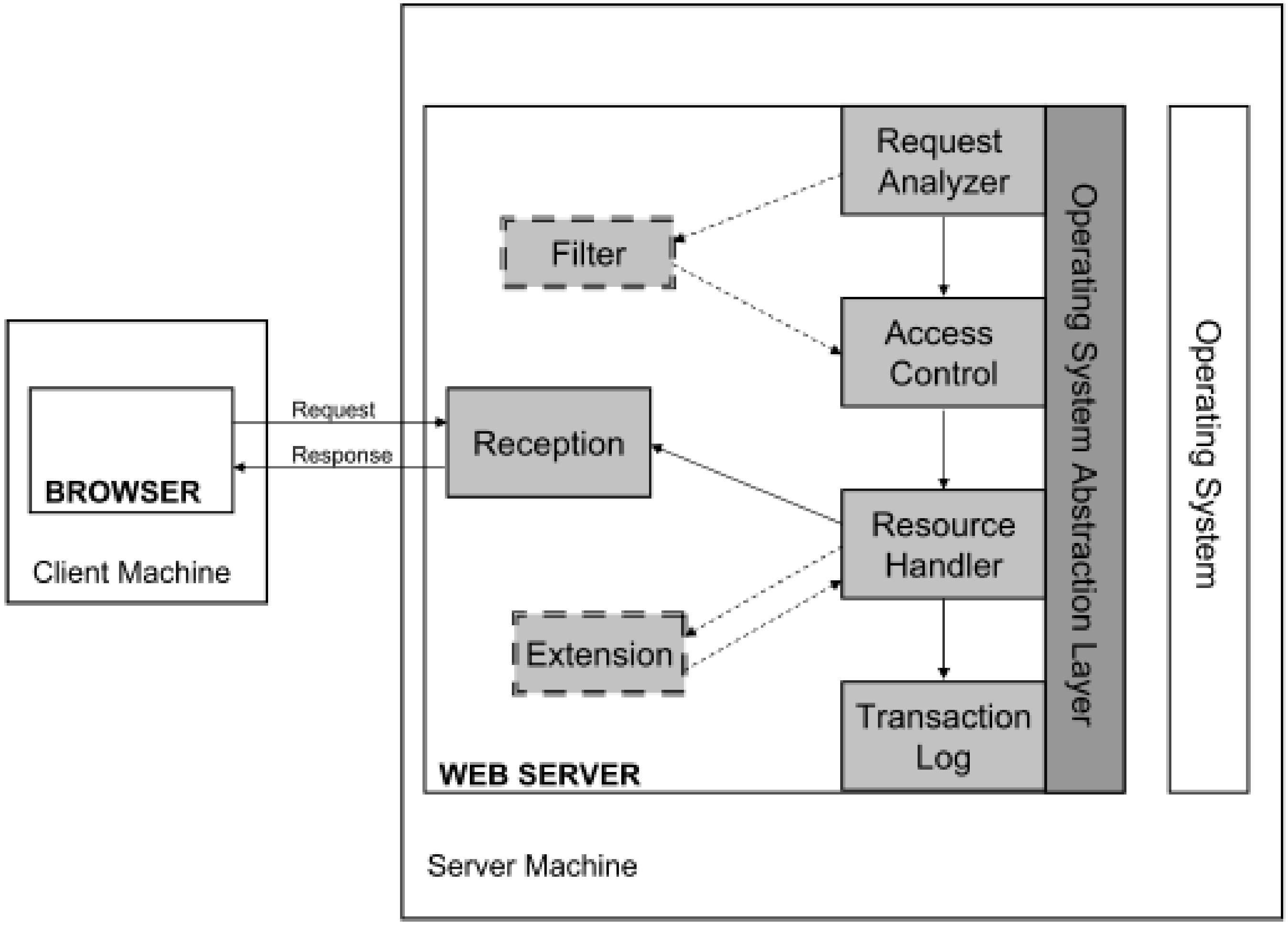
# Scalable CGI



## Scalable CGI advantages / disadvantages

- FastCGI is the most well-known implementation.
- Performance is very good, still better than more recent technologies.
- The usability disadvantages of CGI still apply, programmers are responsible for everything and must know details of HTTP.

# Web server APIs



## Web Server APIs

- NSAPI, ISAPI, Apache API.
- Very efficient since compiled extension modules run within the Web server's address space…
- …but also dangerous since a bug in an extension module can crash the Web server.
- Not commonly used for applications, but for performance reasons, most server-side technologies that support dynamic content are based on Web server extension modules.

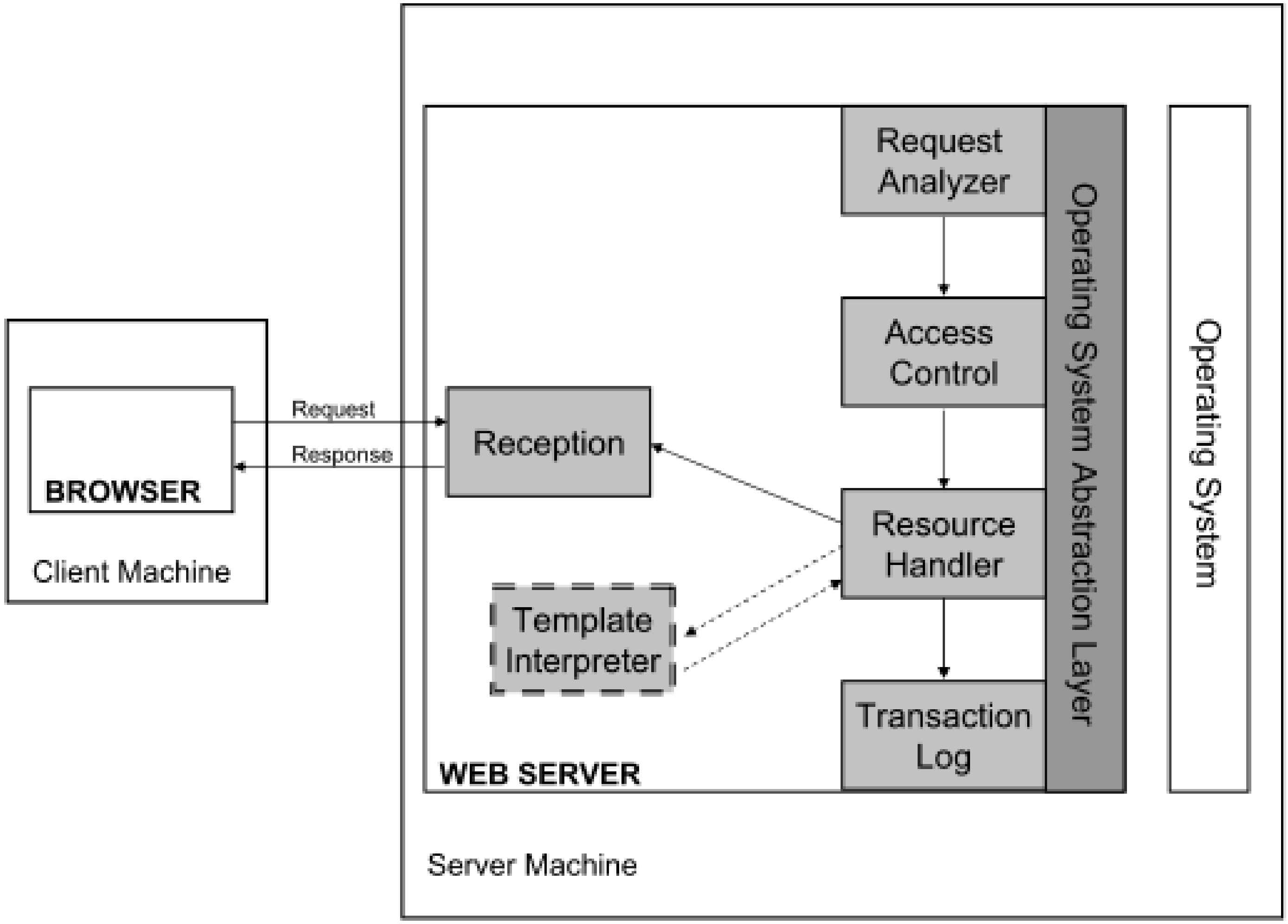
## Interpreted Template-based Scripting

- A template is a model for a set of documents composed of fixed and variable parts.
- Variable parts encapsulate programming logic.
- Many websites consist of fixed HTML pages with small amounts of variable content
  - Server-Side Includes (SSI)
  - Extended SSI (XSSI)
  - ColdFusion
  - Server-side Java Script (SSJS)
  - Active Server Pages (ASP)
  - PHP
- CGI processing is more suitable when pages contain lot of variable content.

## Templates

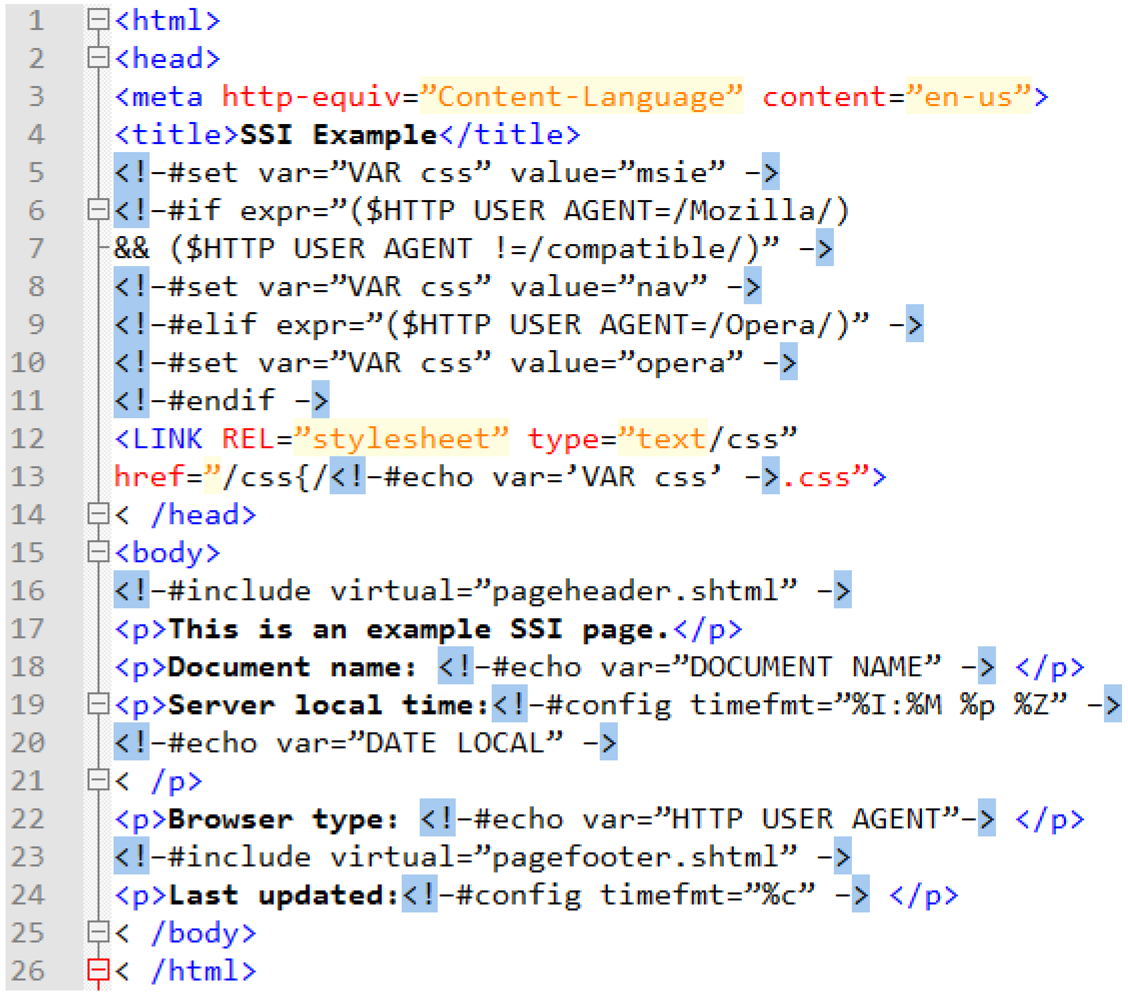
- Template model better supports the common web pattern where the logic is embedded at few places.
- Templates reduce the programming skill needed to create dynamic content
- Role separation is well-supported. Designers and Programmers. Programming logic can be delegated.

# Templates


Request
Response
BROWSER
Client Machine
Reception
Template Interpreter
Request Analyzer
Access Control
Resource Handler
Transaction Log
Operating System Abstraction Layer
Operating System
WEB SERVER
Server Machine

# Server Side Includes (SSI)

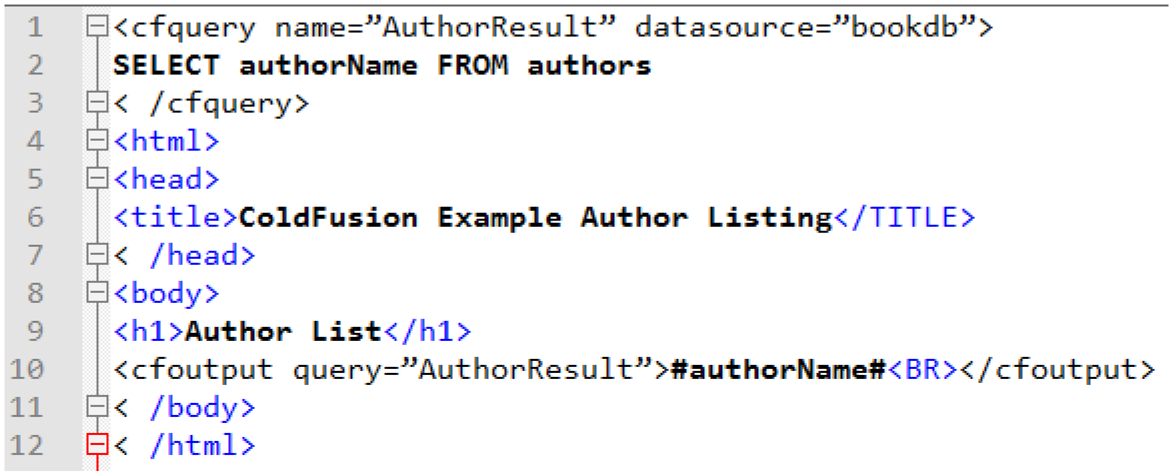
- include command
- echo command
- exec command



```
<html>
<head>
<meta http-equiv=”Content-Language” content=”en-us”>
<title>SSI Example</title>
<!-#set var=”VAR css” value=”msie” ->
<!-#if expr=”($HTTP USER AGENT=/Mozilla/)
&& ($HTTP USER AGENT !=/compatible/)” ->
<!-#set var=”VAR css” value=”nav” ->
<!-#elif expr=”($HTTP USER AGENT=/Opera/)” ->
<!-#set var=”VAR css” value=”opera” ->
<!-#endif ->
<LINK REL=”stylesheet” type=”text/css”
href=”/css{/<!-#echo var=’VAR css’ ->.css”>
< /head>
<body>
<!-#include virtual=”pageheader.shtml” ->
<p>This is an example SSI page.</p>
<p>Document name: <!-#echo var=”DOCUMENT NAME” -> </p>
<p>Server local time:<!-#config timefmt=”%I:%M %p %Z” ->
<!-#echo var=”DATE LOCAL” ->
< /p>
<p>Browser type: <!-#echo var=”HTTP USER AGENT”-> </p>
<!-#include virtual=”pagefooter.shtml” ->
<p>Last updated:<!-#config timefmt=”%c” -> </p>
< /body>
< /html>
```

# ColdFusion

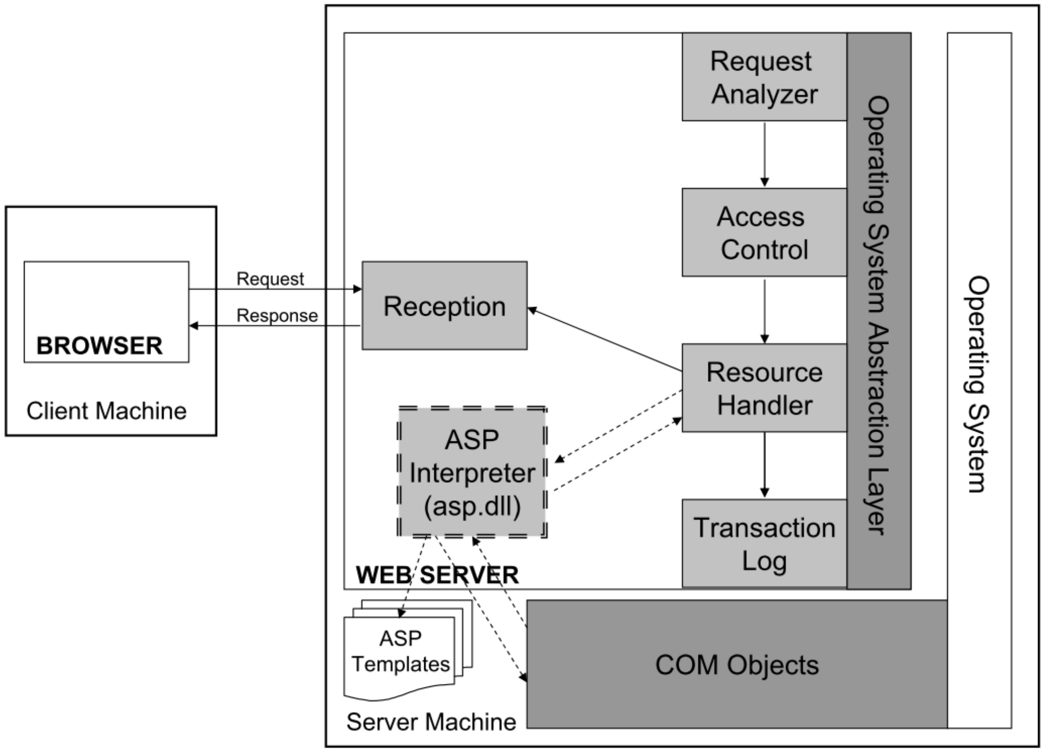
- ColdFusion Markup Language



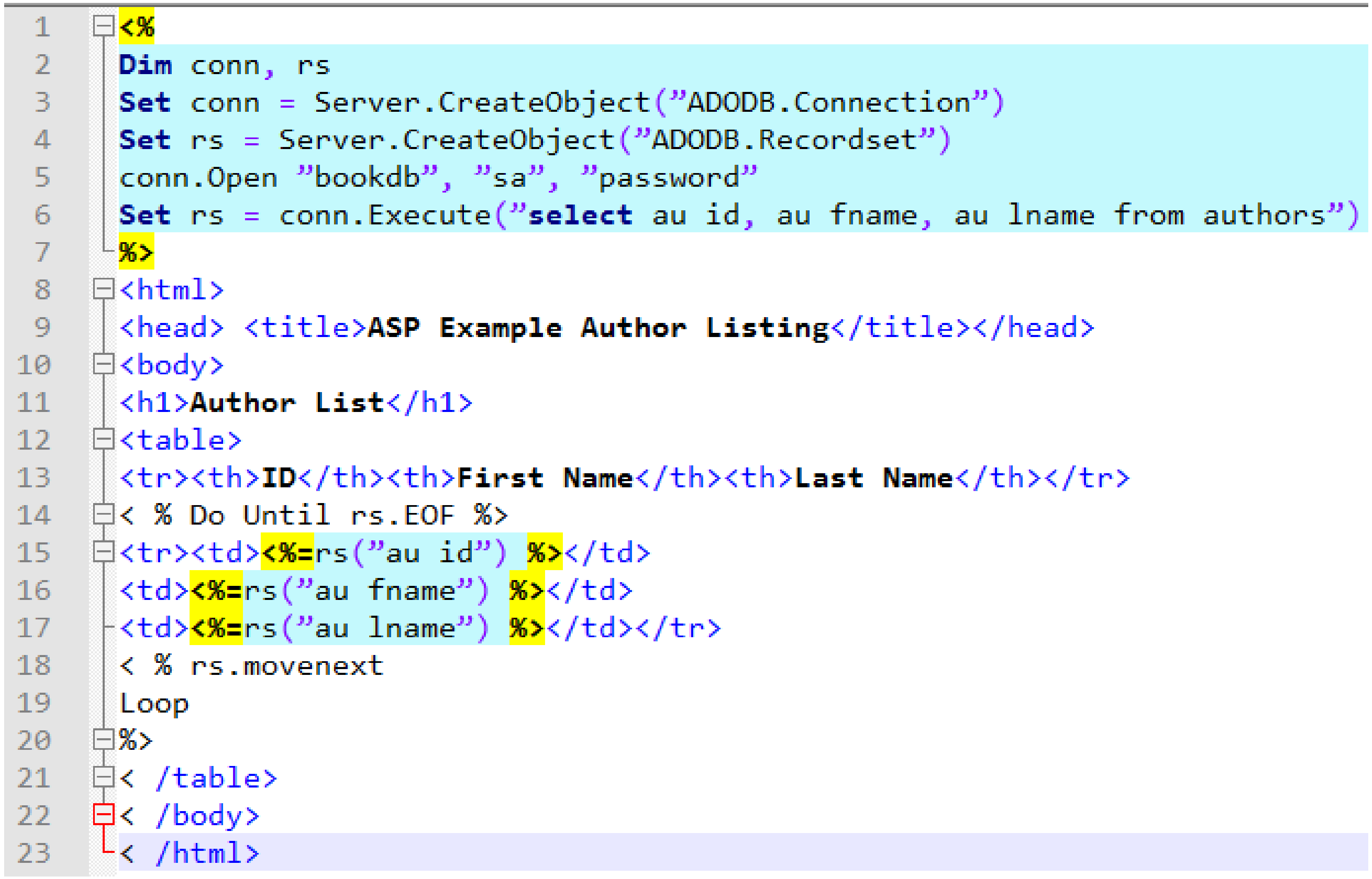
```
<cfquery name="AuthorResult" datasource="bookdb">
SELECT authorName FROM authors
< /cfquery>
<html>
<head>
<title>ColdFusion Example Author Listing</TITLE>
< /head>
<body>
<h1>Author List</h1>
<cfoutput query="AuthorResult">#authorName#<BR></cfoutput>
< /body>
< /html>
```

## Active Server Pages (ASP)

- Scripting language is VBScript
- Script blocks within HTML <%  %>
- Component Object Model (COM) of Microsoft available



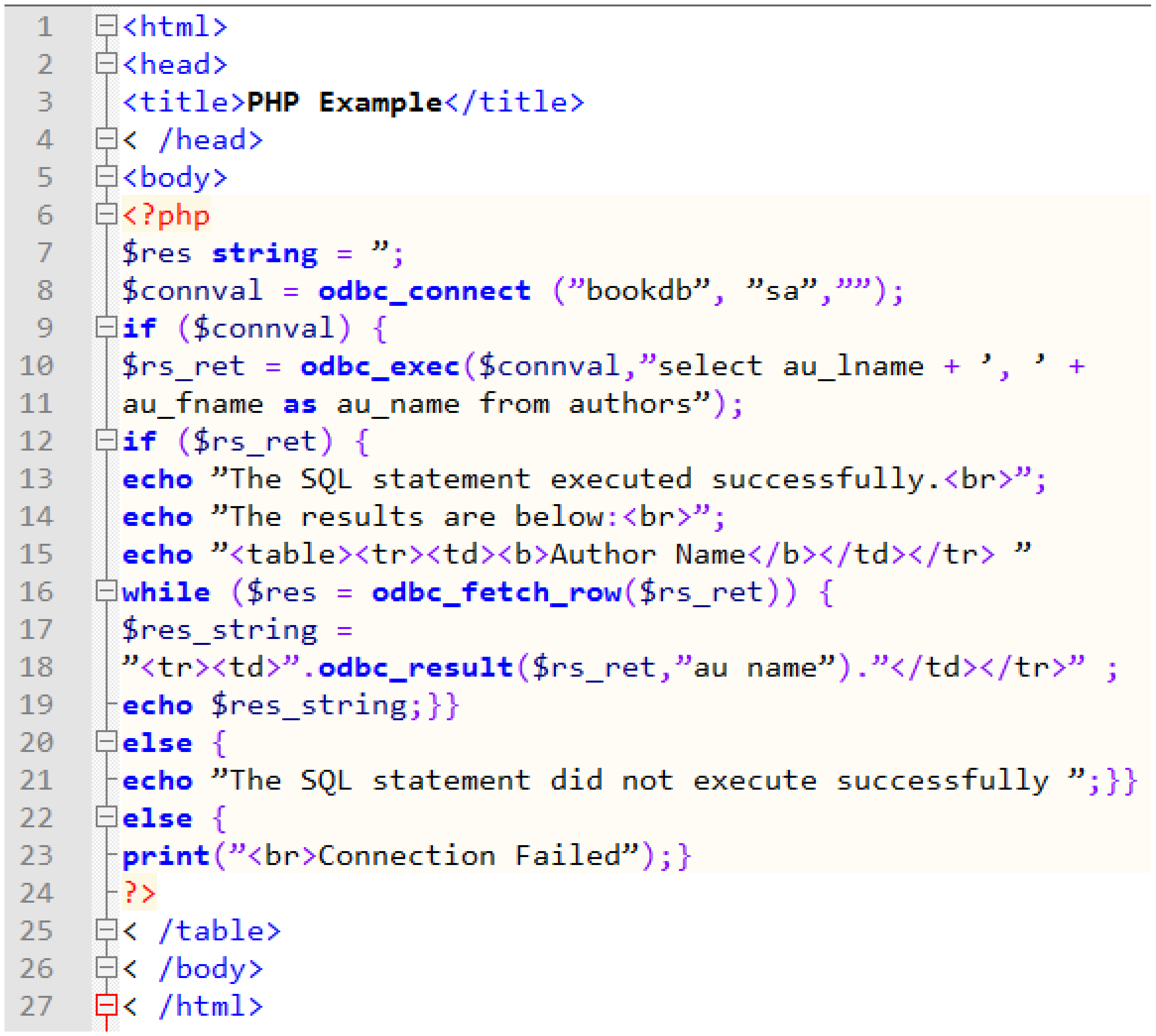
# Active Server Pages (ASP)



```
<%
Dim conn, rs
Set conn = Server.CreateObject(”ADODB.Connection”)
Set rs = Server.CreateObject(”ADODB.Recordset”)
conn.Open ”bookdb”, ”sa”, ”password”
Set rs = conn.Execute(”select au id, au fname, au lname from authors”)
%>
<html>
<head> <title>ASP Example Author Listing</title></head>
<body>
<h1>Author List</h1>
<table>
<tr><th>ID</th><th>First Name</th><th>Last Name</th></tr>
< % Do Until rs.EOF %>
<tr><td><%=rs(”au id”) %></td>
<td><%=rs(”au fname”) %></td>
<td><%=rs(”au lname”) %></td></tr>
< % rs.movenext
Loop
%>
< /table>
< /body>
< /html>
```

# PHP

- Open-source
- Cross-platform
- Object-based scripting language
- Linux, Apache, MySQL, PHP (LAMP)
    - Low cost platform for web applications



```
<html>
<head>
<title>PHP Example</title>
< /head>
<body>
<?php
$res string = ”;
$connval = odbc_connect (”bookdb”, ”sa”,””);
if ($connval) {
$rs_ret = odbc_exec($connval,”select au_lname + ’, ’ +
au_fname as au_name from authors”);
if ($rs_ret) {
echo ”The SQL statement executed successfully.<br>”;
echo ”The results are below:<br>”;
echo ”<table><tr><td><b>Author Name</b></td></tr> ”
while ($res = odbc_fetch_row($rs_ret)) {
$res_string =
”<tr><td>”.odbc_result($rs_ret,”au name”).”</td></tr>” ;
echo $res_string;}}
else {
echo ”The SQL statement did not execute successfully ”;}}
else {
print(”<br>Connection Failed”);}
?>
< /table>
< /body>
< /html>
```

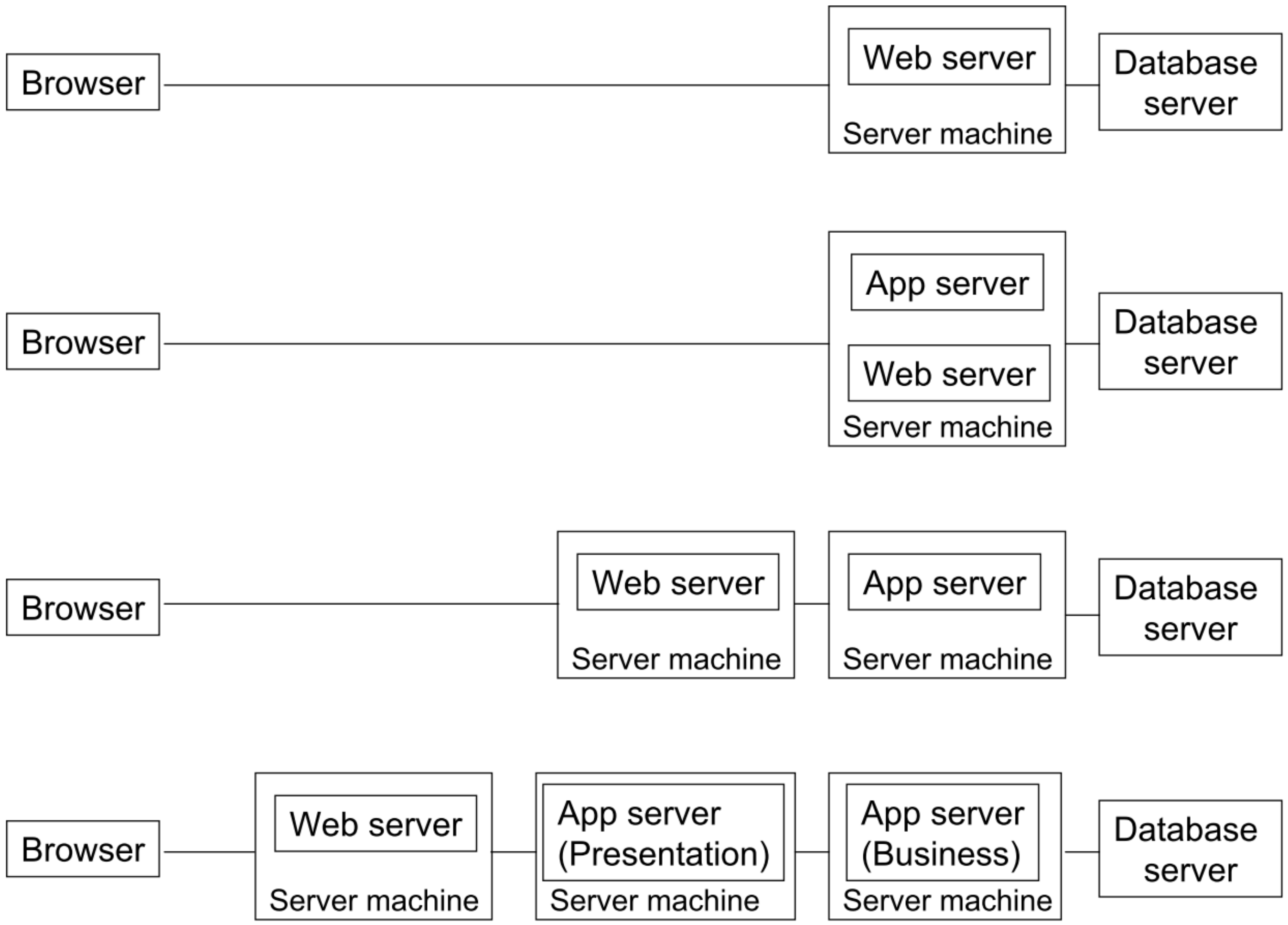
## Scaling Up To The Enterprise

- Instead of being able to focus solely on business logic, Web Application developers have to implement home grown solutions for transaction mgmt, resource pooling, etc.
- For large-scale Web application development, infrastructure concerns have to be separated from business logic and presentation concerns

## App Servers, Components, Middleware

- Large-scale web applications
  - Separate web server, presentation logic, business logic, and data access into distinct tiers.
  - Enterprise Web applications have reliability and infrastructural requirements
    - Middleware services are required to support these requirements.
    - Application servers aim to meet middleware requirements

## Scaling Up


Browser
Web server
Server machine
Database server
Browser
App server
Web server
Server machine
Database server
Browser
Web server
Server machine
App server
Server machine
Database server
Browser
Web server
Server machine
App server (Presentation)
Server machine
App server (Business)
Server machine
Database server

## Middleware

- Middleware
  - Reliability
  - Throughput
  - Integration
  - Security
  - Development
- Application servers and components
- Java
  - Servlets
  - JSP
  - J2EE
- .NET

## Requirements for Enterprise Business Systems

| Requirement | Approach/Sol | Description |
|---|---|---|
| Reliability | Transparent fail-over | Route requests for failed services to another server. |
| | Transaction support | Units of work completely succeed or are rolled back. |
| | System management | Provide system monitoring and control capabilities. |
| | High availability | Minimize time that a system is not available due to failures. |
| | Replication | Duplicate resources for load balancing or recovery. |
| | Failure recovery | Detect service failures, divert requests to another instance. |
| | | |
| | | |
| | | |

## Requirements for Enterprise Business Systems

| Requirement | Approach/Sol | Description |
|---|---|---|
| Throughput | Load balancing | Alternate requests between servers to equalize utilization. |
| | Clustering | Interconnect multiple servers to share a processing workload. |
| | Threading | Execute requests concurrently within a process. |
| | Efficiency | Process requests with minimal resources and latency. |
| | Scalability | Maintain stable performance as the rate of requests increases. |
| | Resource pooling | Share resources between multiple users in an optimized way. |
| | Caching | Save computations for later compatible requests. |
| | | |
| | | |

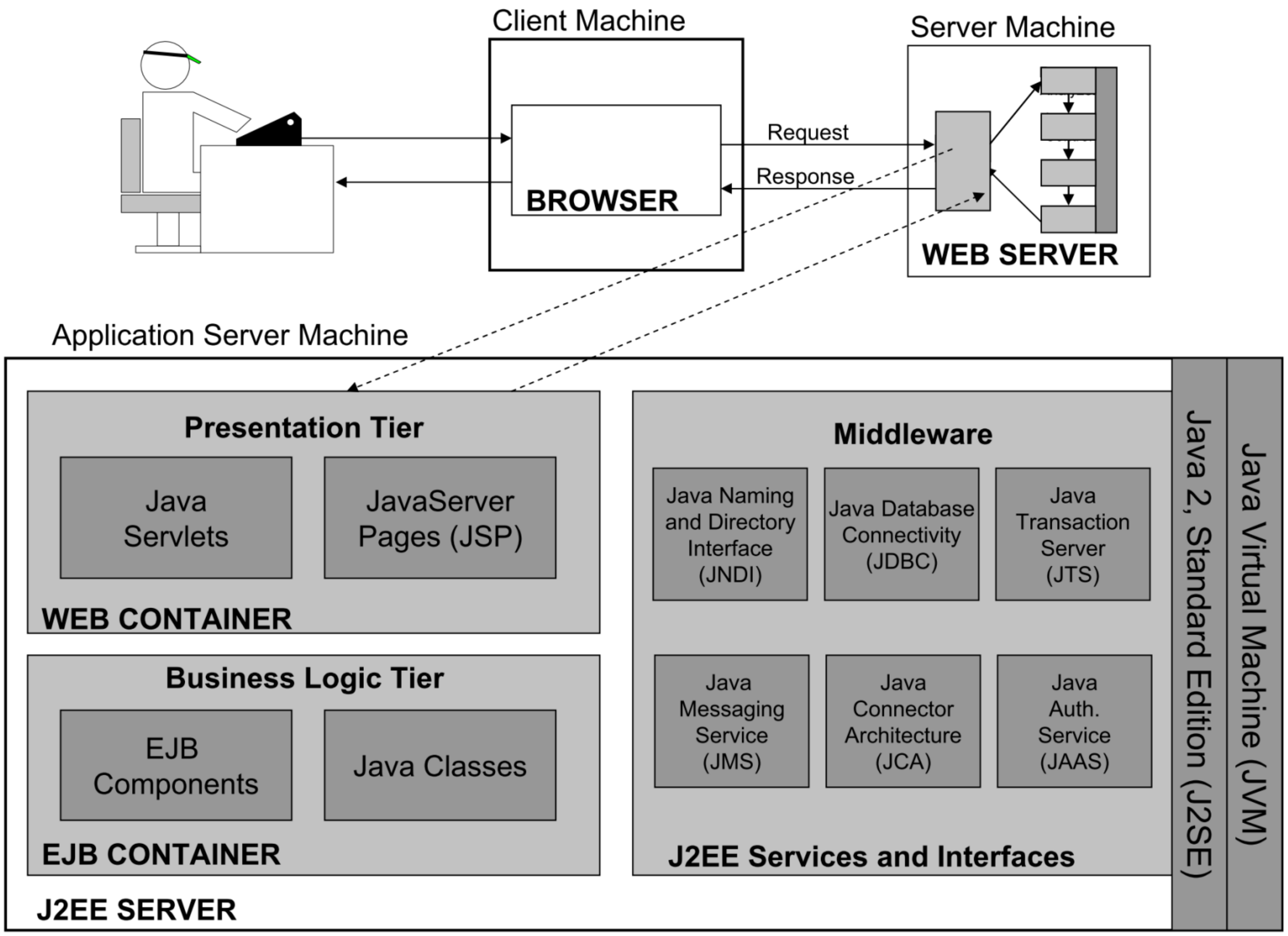
## Requirements for Enterprise Business Systems

| Requirement | Approach/Sol | Description |
|---|---|---|
| Integration | Remote method invocation | Interfaces for synchronous method invocation. |
| | Back-end integration | Interfaces to external information systems. |
| | Database access | Store and retrieve database information. |
| | Location transparency | Allow services to be requested by directory name. |
| | Multi-protocol support | Integrate multiple protocols into a uniform interface. |
| | Message-passing | Asynchronous communications though message-passing. |
| | Legacy connection | Interfaces to obsolete or surpassed technologies. |
| Security | Logging and auditing | Record significant activities that occur within the system. |
| | Permission checking | Verify identity and protect resources. |

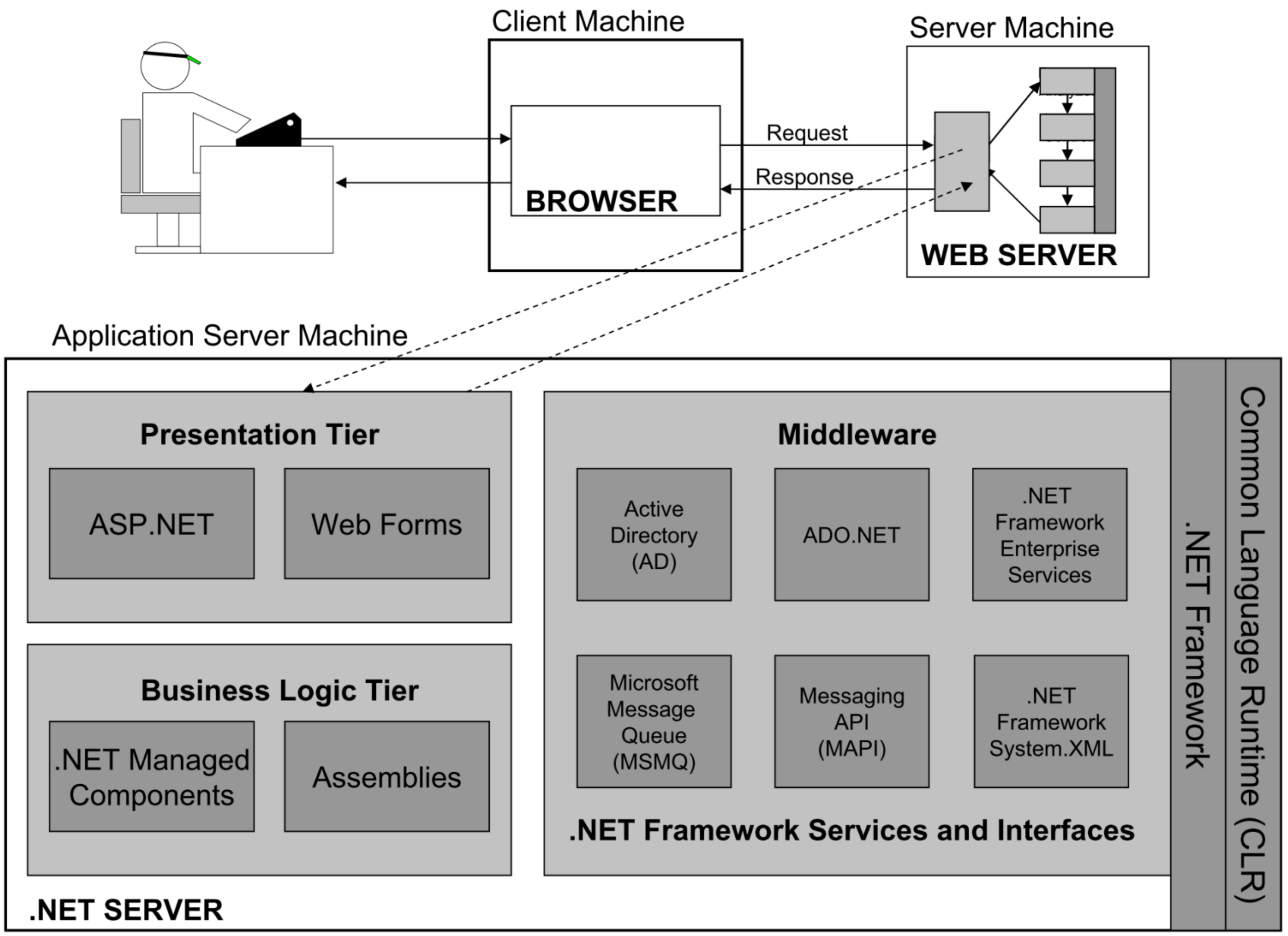
## Requirements for Enterprise Business Systems

| Requirement | Approach/Sol | Description |
|---|---|---|
| Integration | Rapid development | Reliable applications can be developed quickly. |
| | Dynamic redeployment | Running applications can be updated without interruption. |
| | Separation of concerns | Role-specific contributions are separate in code modules. |
| | Modularity | Isolated changes minimally impact other parts of a system. |
| | Reusability | Modules can be used for multiple applications. |
| | Software scalability | Software remains manageable as system size increases. |
| | Portable languages | Modules can run without changes on multiple platforms. |
| | Standardization | The technology has multiple compliant providers. |
| | | |

# J2EE


Client Machine
Server Machine
BROWSER
Request
Response
WEB SERVER
Application Server Machine
Presentation Tier
Java Servlets
JavaServer Pages (JSP)
WEB CONTAINER
Business Logic Tier
EJB Components
Java Classes
EJB CONTAINER
J2EE SERVER
Middleware
Java Naming and Directory Interface (JNDI)
Java Database Connectivity (JDBC)
Java Transaction Server (JTS)
Java Messaging Service (JMS)
Java Connector Architecture (JCA)
Java Auth. Service (JAAS)
J2EE Services and Interfaces
Java 2, Standard Edition (J2SE)
Java Virtual Machine (JVM)

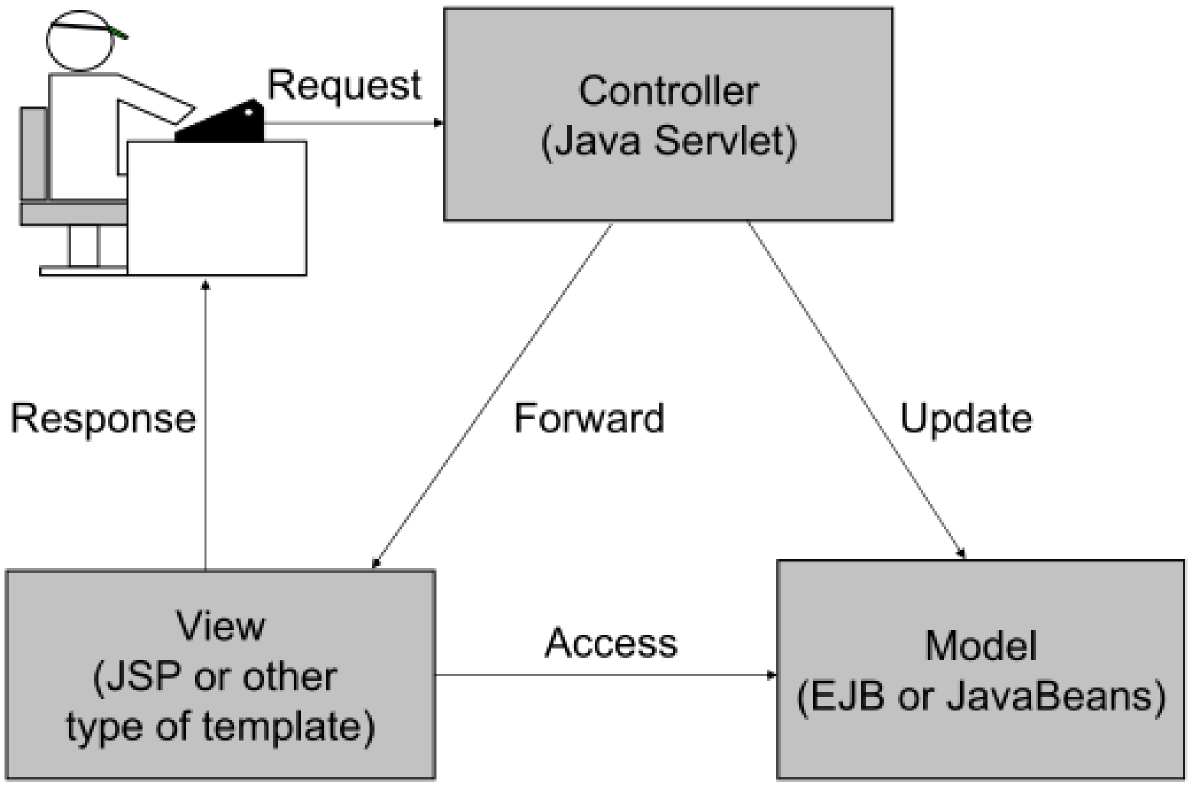
# .NET


Client Machine
Server Machine
BROWSER
Request
Response
WEB SERVER
Application Server Machine
Presentation Tier
ASP.NET
Web Forms
Business Logic Tier
.NET Managed Components
Assemblies
Middleware
Active Directory (AD)
ADO.NET
.NET Framework Enterprise Services
Microsoft Message Queue (MSMQ)
Messaging API (MAPI)
.NET Framework System.XML
.NET Framework Services and Interfaces
.NET Framework
Common Language Runtime (CLR)
.NET SERVER

## Web programming vs. regular programming

- Web development traditionally lagged state-of-the-art, until J2EE.
- Approaches carried forward to the Web
  - Patterns
  - Tiered architectures
  - Frameworks
    - Persistence
    - Lightweight containers
    - WebMVC

# MVC


Request
Controller
(Java Servlet)
Response
Forward
Update
View
(JSP or other
type of template)
Access
Model
(EJB or JavaBeans)

## Acknowledgements

## Q&A

innovate achieve lead

BITS Pilani
Pilani Campus

## Thank You