CSP2103-4102: Markup Languages

Lecture 9: Introduction to Server-Side Systems

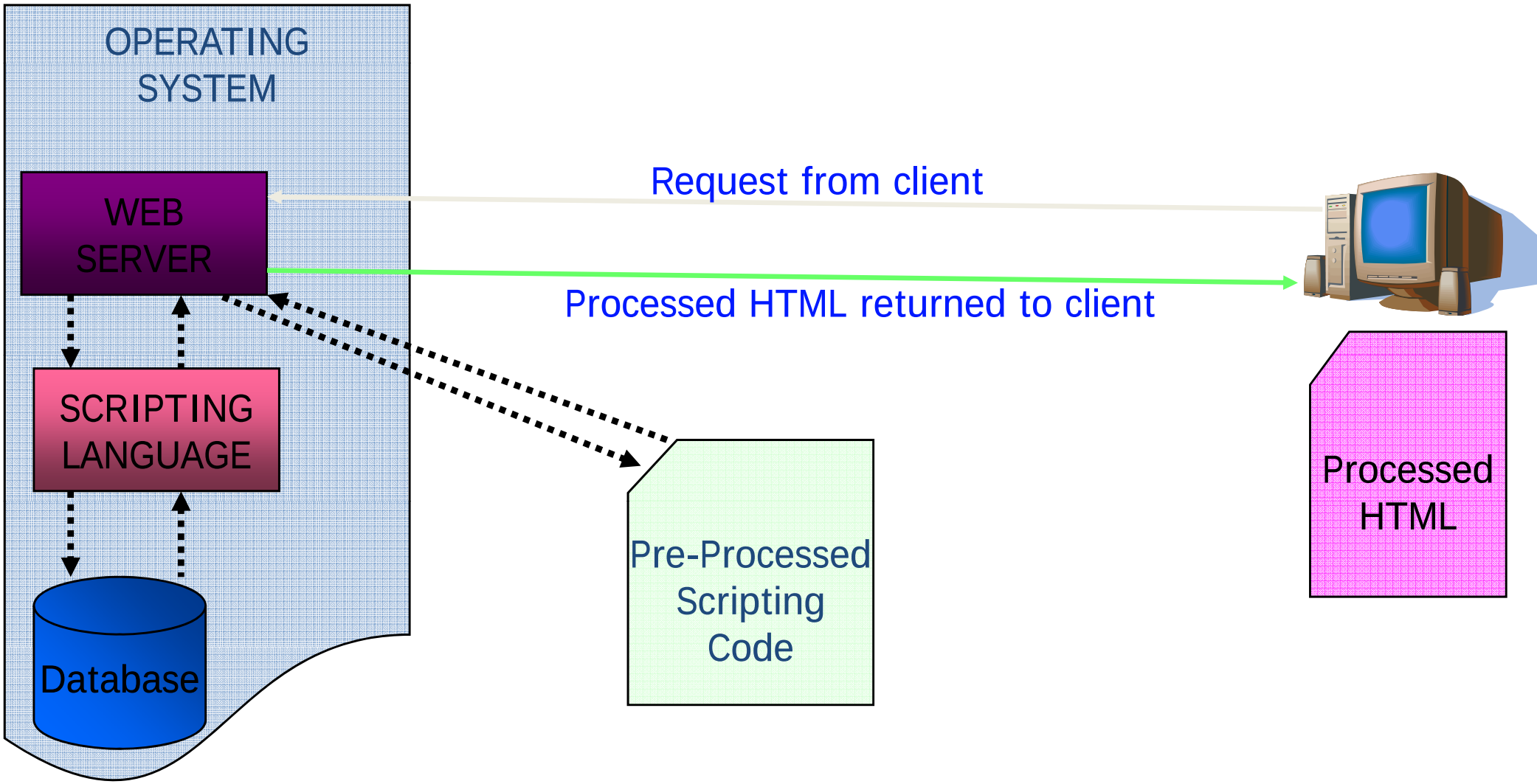
Introduction

- Client vs Server side
- Server-side programming languages
- Features of server-side languages
- Advantages of Server side
- Web server / scripting language interaction
- Scripts / Servers and Databases
- Security

Client / Server

- Client side is the user's side (the web browser on their machine)
- Clients request data from the web server via the web (using HTTP and TCP/IP)
- The web server receives the data request and sends it to the appropriate scripting tool
- Script generates HTML, integrated with data, and returns HTML page to browser

Server Side Elements



Which scripting tool where?

SERVER SIDE

- PHP
- ASP
- Perl
- JSP
- Cold Fusion
- ASP.Net
- C#

CLIENT SIDE

JavaScript

Jscript

VBscript

Flash

XML

XSLT

PHP

- Acronym for **P**HP: **H**ypertext **P**re-processor
- Contains elements of C, Java and Perl
- Designed with web development in mind
- Open source
- Highly integrated with a variety of databases
  - MySQL, Postgres SQL
- Can be embedded in HTML
- Also has modules to integrate with and generate XML

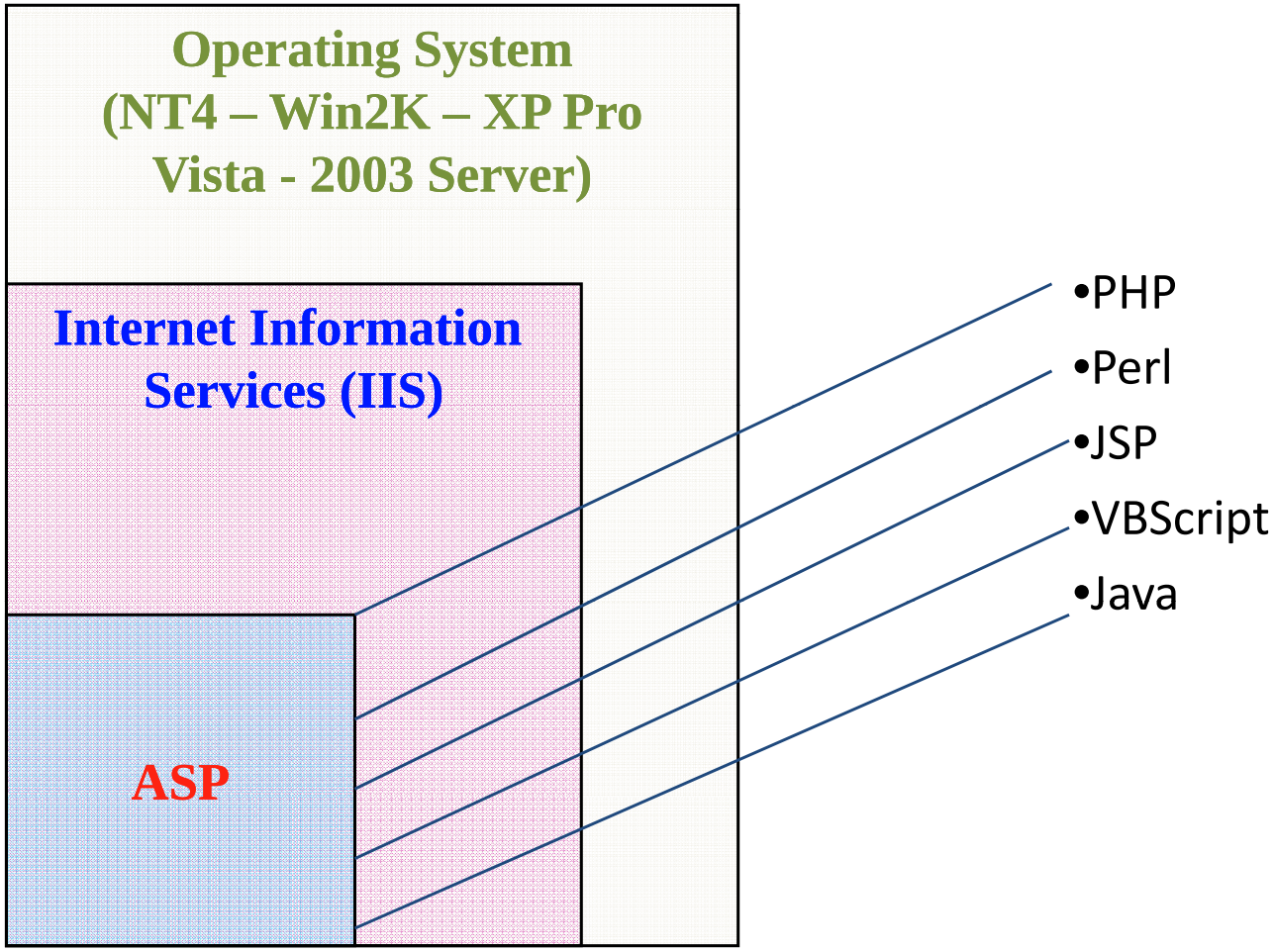
PHP cont…

- Easy to learn
- Heaps of in-built functionality
  - Generate PDF files
  - file uploads
- Works with a large number of web servers
- Highly integrated with Apache server
- Available on nearly all major Operating Systems

ASP

- Active Server Pages
- Proprietary Microsoft Technology
- Is not a language as such, more an encapsulation system
- Allows almost any kind of interpreted or compiled code to run as a web dev tool
- Provides a common interface fore exposing code objects
- Can generate HTML and be embedded in HTML
- Requires Internet Information Services (IIS) with Microsoft OS's

ASP cont…


Operating System
(NT4 – Win2K – XP Pro
Vista - 2003 Server)
Internet Information
Services (IIS)
ASP
•PHP
•Perl
•JSP
•VBScript
•Java

Perl

- Originally, most widely used server-side coding tool
- Extremely powerful, especially for text manipulation
- Cross-platform
- Not specifically designed for web development, simply adapted for the task
- Once again, based around C like syntax
- Origins in unix / linux world

Perl cont…

- Daunting to learn for beginners
- Required precision in logical and syntax
- Used to generate HTML, but cannot be embedded
- Also can be integrated with almost any web server
- Less common in web dev now
- Purists still consider it to be ‘a real language’

Cold Fusion

- ColdFusion is quite different as it is a tag-based coding tool
- Rapid Application Development
- Supports nearly all kinds of databases, scripting languages in addition to own coding tags
- Highly proprietary, needs specialised server
- Expensive
- Highly manageable
- Tag based coding looks very similar to XSLT
- Declined massively in popularity and use in the last few years

ASP.NET

- Built on Microsoft's .Net Framework
- Fully object-oriented
- Allows for web pages and web services to be generated from almost any language
- Web pages are compiled before they can be executed on the server
- Distinction between client-server means less, as client/server side functionality intertwined
- State management almost completely transparent
- Allows for easy learning and RAD (assuming you know OO coding methodology)

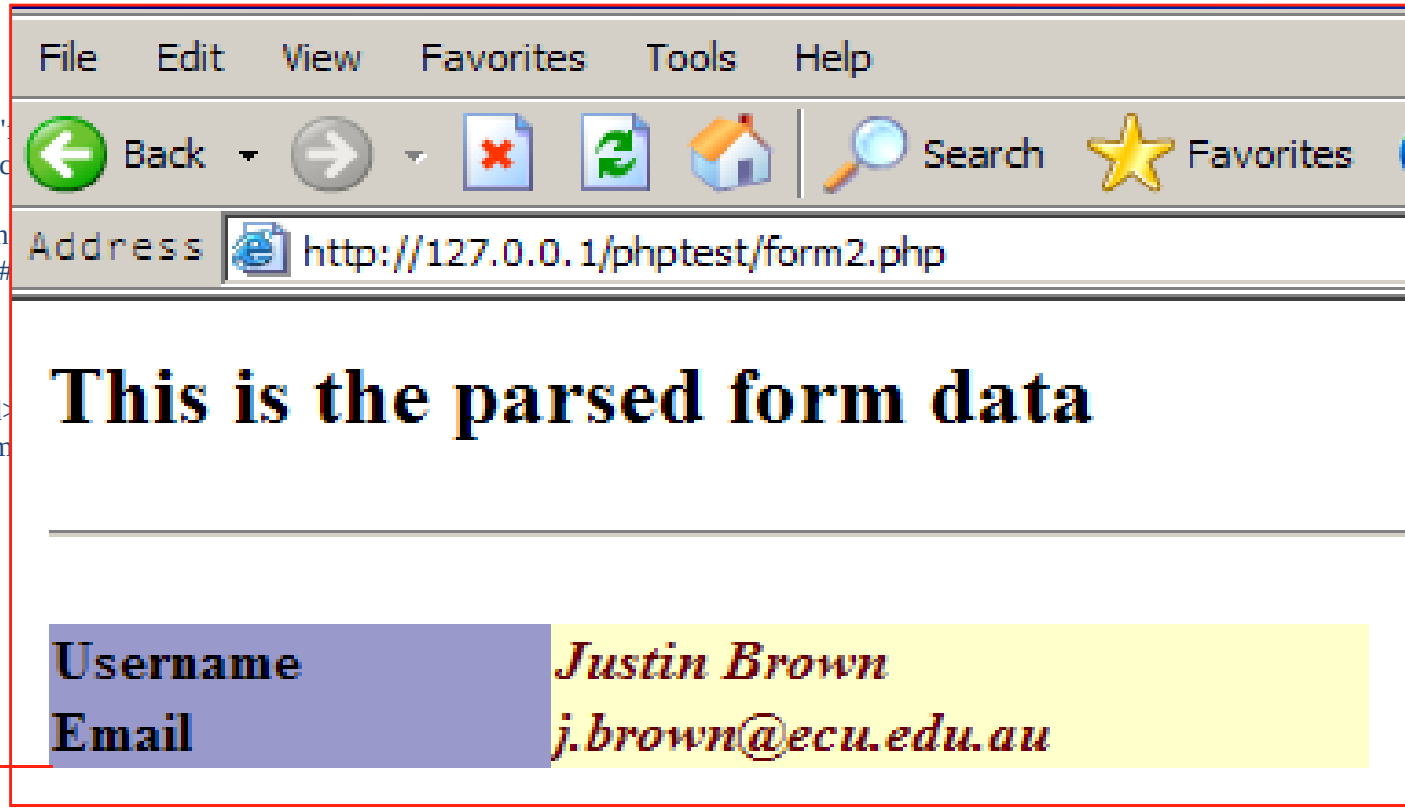
C# (pronounced C Sharp)

- Microsoft's version of Java
- Compiled language that be capable on running on almost any hardware / software platform
- Centrepiece of ASP.NET, replacing VBScript as the primary scripting tool within ASP
- Once again, daunting for beginners
- C# developers in high demand

Script embedded within (X)HTML with PHP

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>Untitled Document</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
</head>

<body>
<h2><strong>This is the parsed form data</strong></h2>
<hr />
<form name="form1" id="form1" method="post" action="
  <table width="29%" border="0" cellpadding="1" cellspac
    <tr>
      <td width="38%" bgcolor="#9999CC"><strong>Usern
      <td width="62%" bgcolor="#FFFFCC"><font color="#
$_POST["UserName"]; ?></strong></em></font></td>
    </tr>
    <tr>
      <td bgcolor="#9999CC"><strong>Email</strong></td>
      <td bgcolor="#FFFFCC"><font color="#660000"><em
?></strong></em></font></td>
    </tr>
  </table>
</form>
</body>
</html>
```



Scripting Example

Pre-Processing : HTML and VBScript

```
1  <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
2  <HTML>
3  <HEAD>
4  <TITLE> My ASP Page </TITLE>
5  </HEAD>
6  <BODY>
7  <H1>WELCOME TO THIS PAGE</H1>
8  <B>The Current time is:</b> <% = Now() %>
9  </BODY>
10 </HTML>
11
```

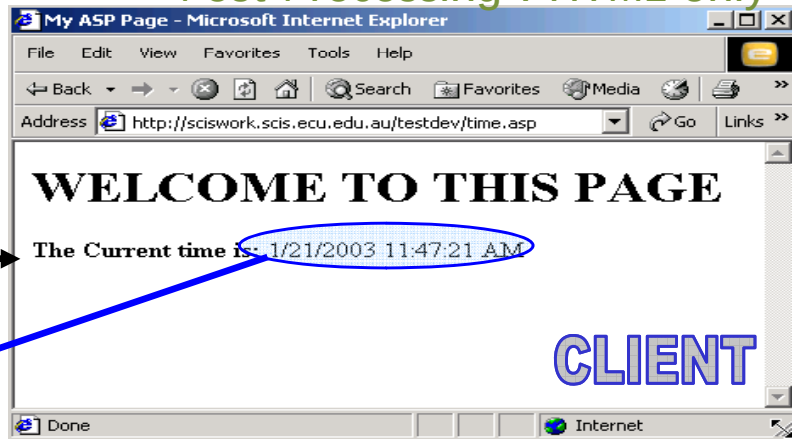
SERVER

VBScript pre-defined Function (gets the current system time)

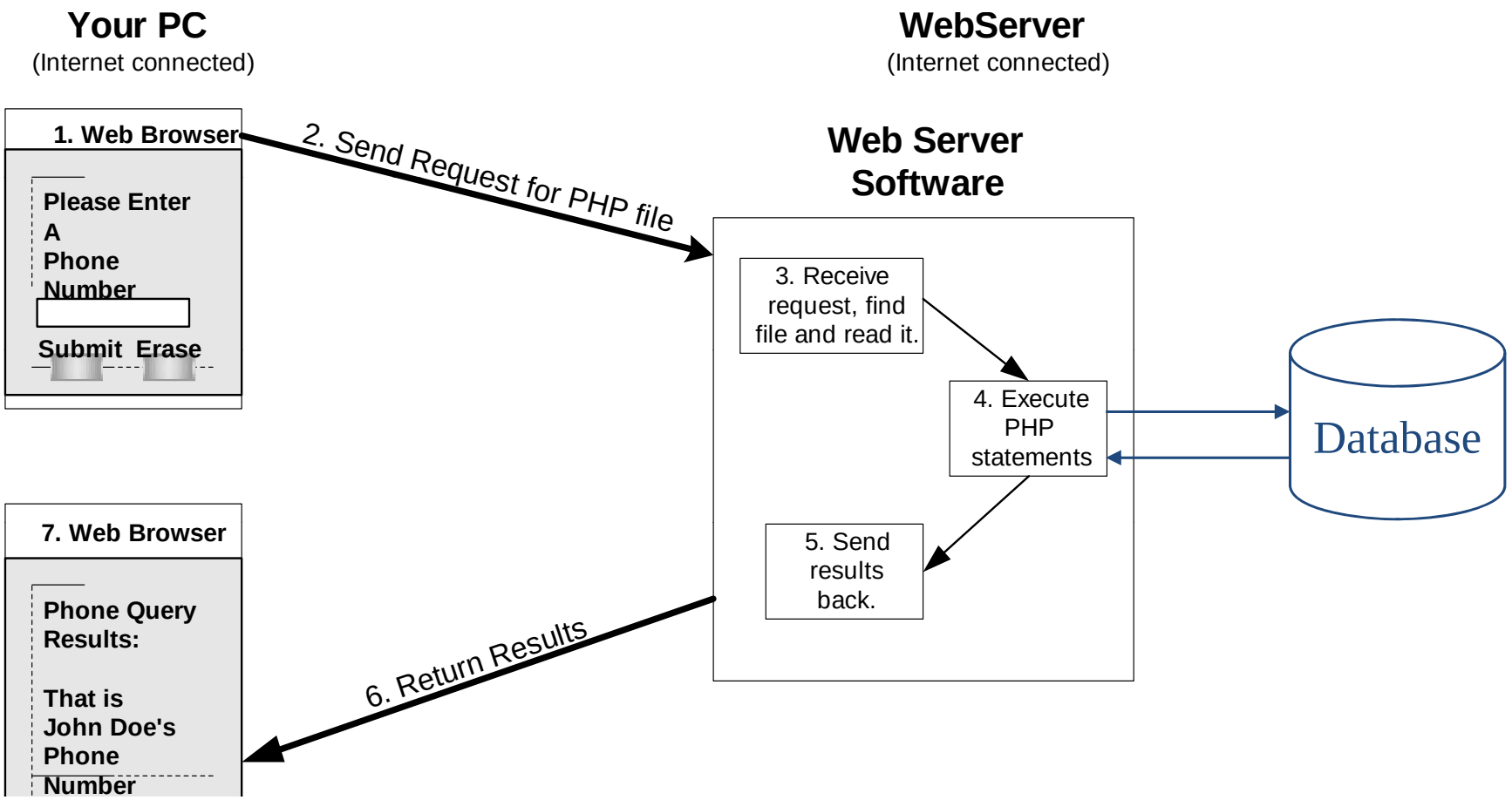
CGI/ISAPI

Processing replaces VBScript with function value as HTML

Post-Processing : HTML only



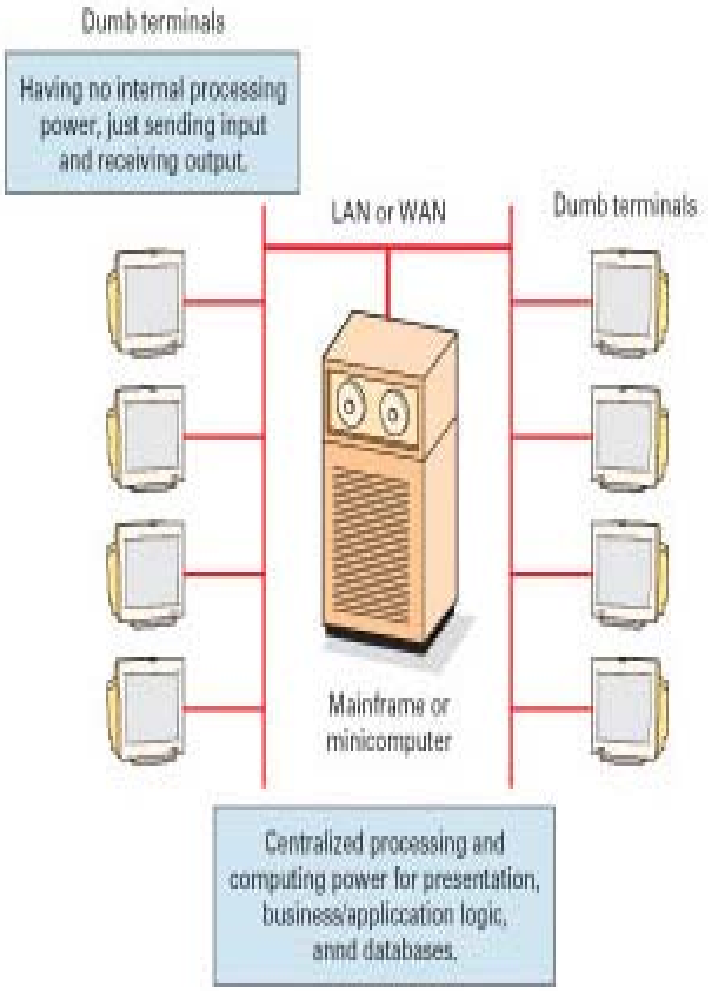
Script Execution Process



One-Tiered Architecture

Traditional Architectures

- Server mainframe or minicomputer
- Dumb terminals
  - No internal processing power
  - No storage capacity
- All processing, calculation, storage done on mainframe
- Disadvantages
  - Single centralized processor
  - Lacks flexibility and scalability

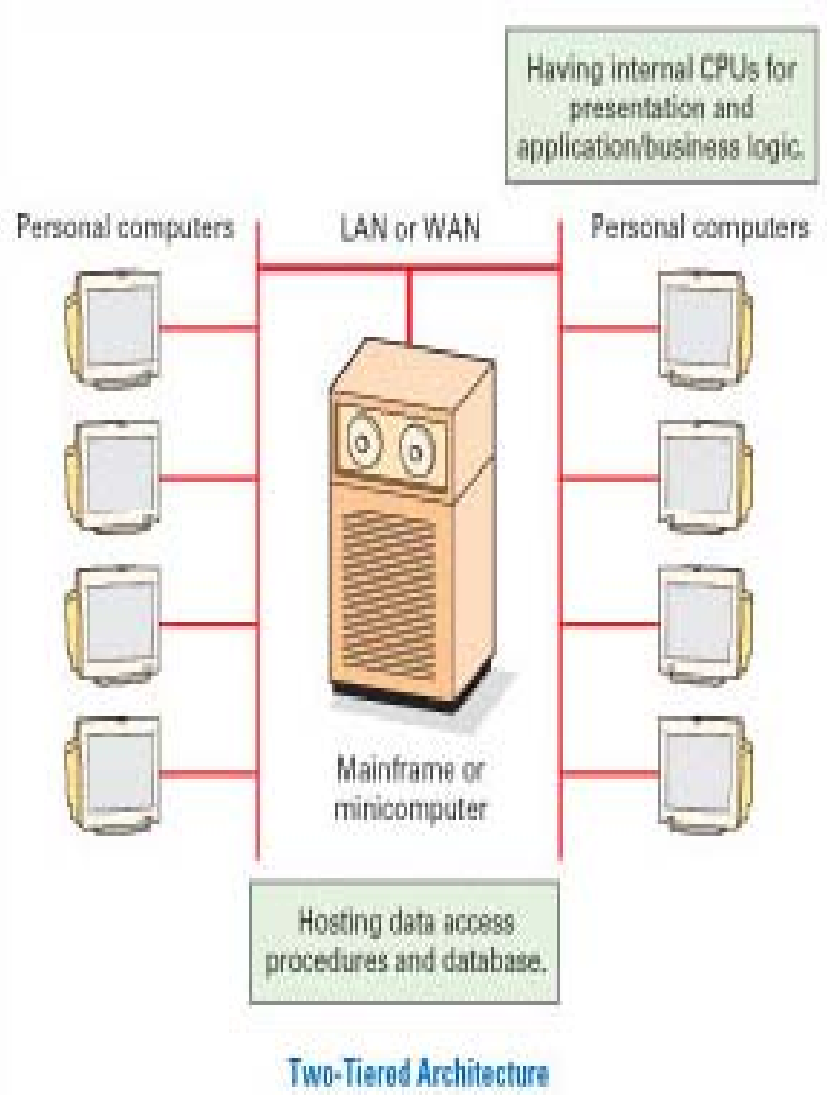


One-Tiered Architecture

Two-Tiered Architecture

Traditional Architectures

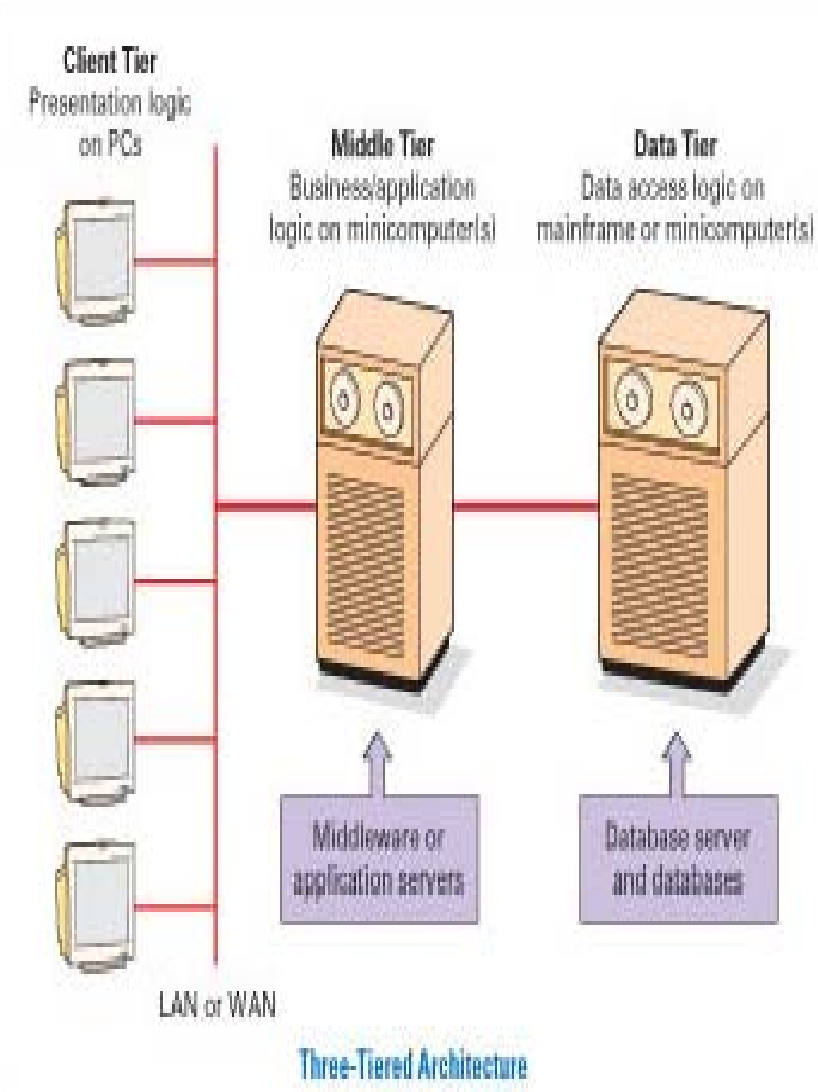
- Server mainframe or minicomputer
- Personal computers
- Processing tasks divided
  - Applications run on PCs
  - Data stored on mainframes
- Disadvantages:
  - Complex applications are large
  - Client PCs become “fat”



Three-Tiered Architecture

Traditional Architectures

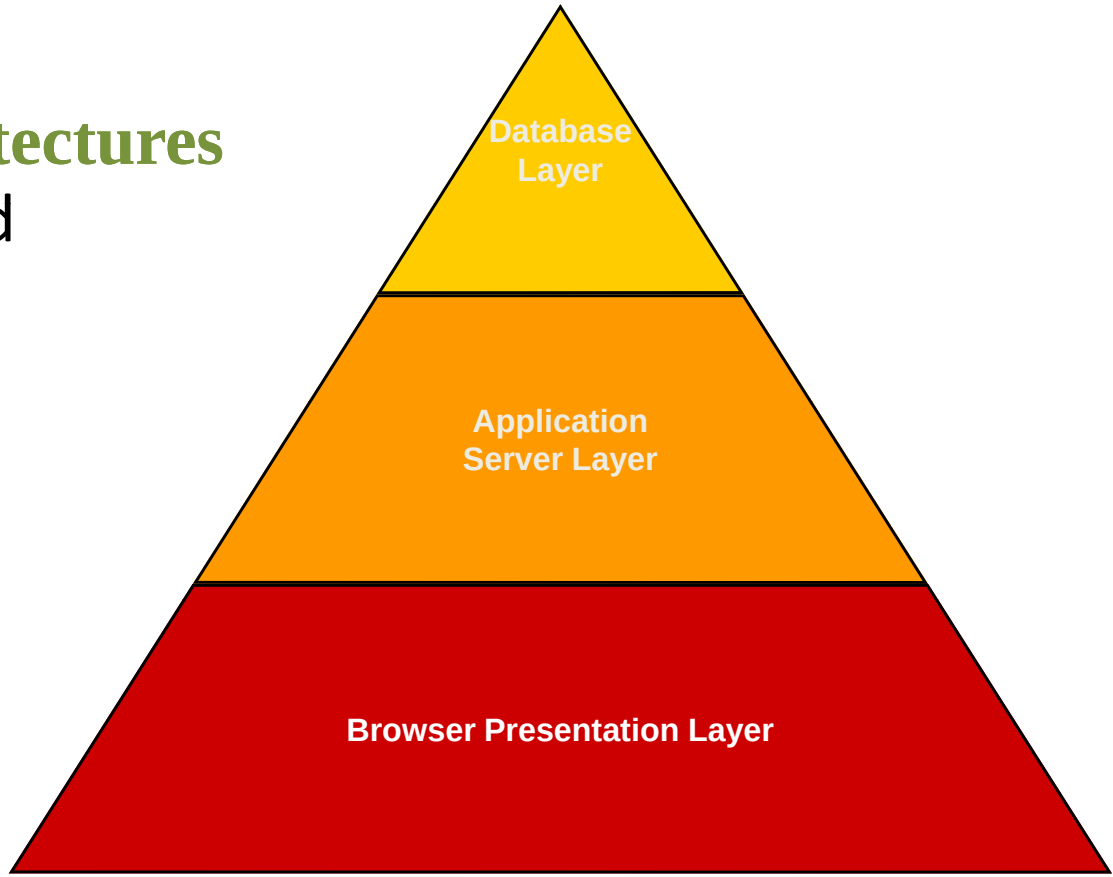
- Server mainframe or minicomputer
- Personal computers
- Application server
  - Middleware
  - Provides applications
  - Acts as transaction monitor
  - Stores data
- Advantages:
  - “Thin” client (such as web browser)



Web-Based Client/Server

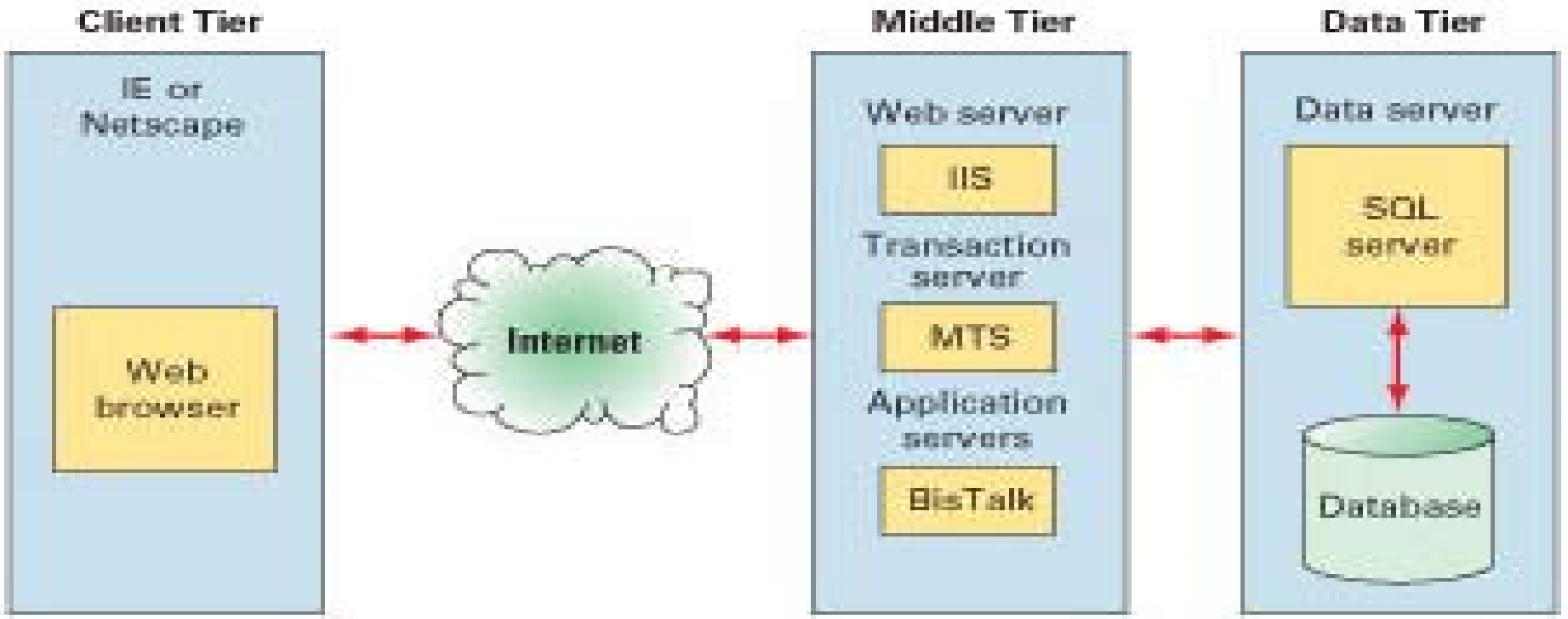
Modern, Distributed Architectures

- Advantages over LANs and WANs:
  - No geographic boundaries
  - Accessible wherever Internet connection available



Web-Based Architecture

Modern, Distributed Architectures



Web-Based Client/Server Architecture

Browser Presentation Layer

Modern, Distributed Architectures

- Human-to-computer interface
- Possible elements:
  - The visual page that users interact with
  - Forms
  - Clickable buttons
- Essentially a logical representation of the data stored in the data tier and allowable functions as dictated by the application tier

Application Server Layer

Modern, Distributed Architectures

- Middle layer
  - Web server takes requests from client browser
  - Forwards requests to the database
  - Server collects responses from database
  - Passes responses to client browsers
- Building an application server layer:
  - Install a Web server
  - Connect the server to the Internet
  - Create a virtual directory for the Web server
  - Write a Web client/server database application with HTML and DHTML

Database Layer

Modern, Distributed Architectures

- Stores and retrieves data
  - SQL requests received from Web server
  - Database app stores data, processes results
  - Sends results to server
  - Server forwards results to client browsers
- Building a database layer:
  - Develop database app on the server computer
  - Create DSNs for the user and system
  - Under some circumstances an XML document could theoretically be used as the data repository within the database layer

Impact of Applications

Modern, Distributed Architectures

- Communication
- Education
- Research
- Software / driver downloads
- Essentially, everything you do on the web from day to day
- Imagine your university / home life without web applications
- In this unit, we have looked at web applications developed from the client-side only (XML and XSLT)
- However, these are analogues of the client/server environments discusses previously

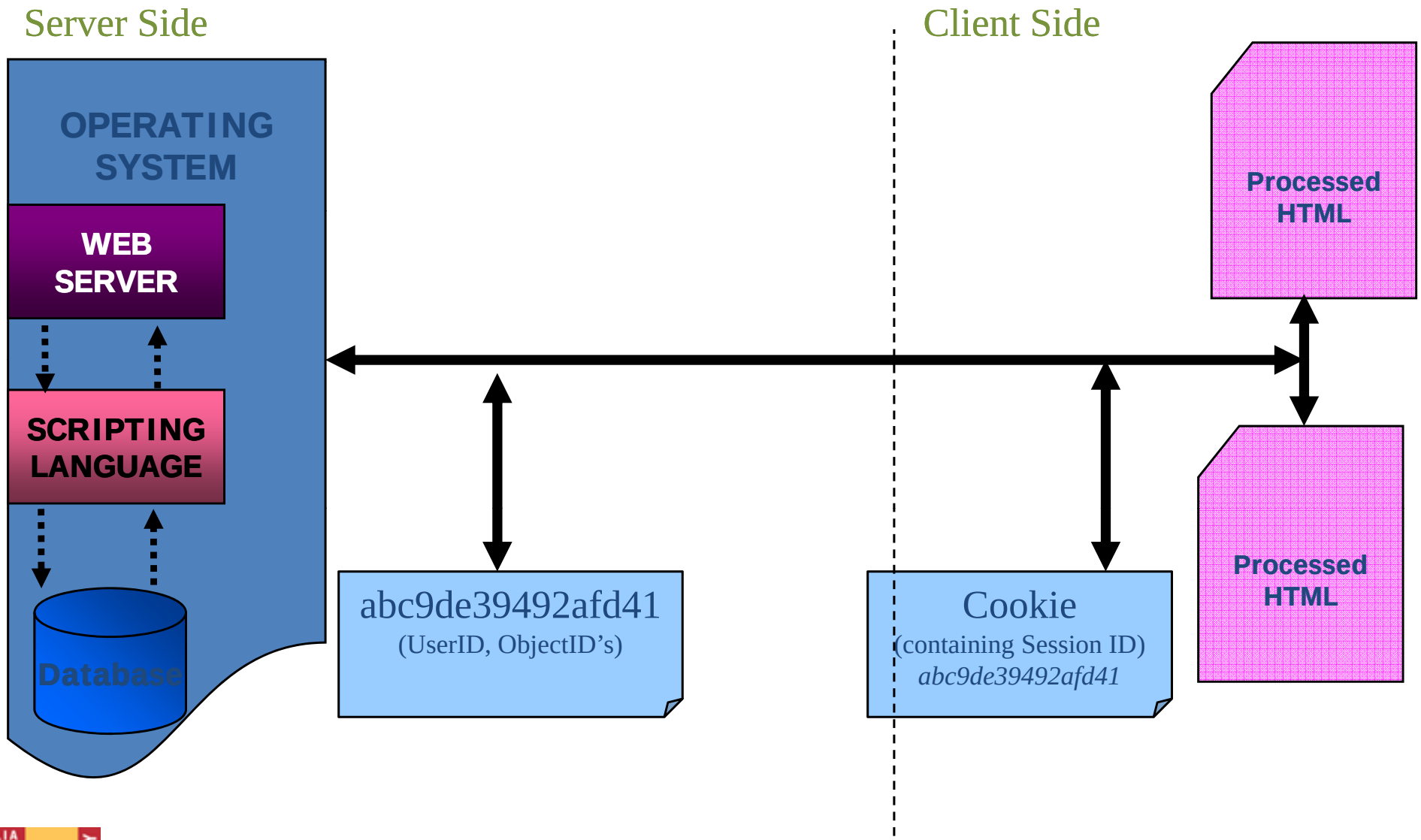
Web servers

- Web servers allow sites and pages to be accessed via the web
- They receive http requests and send formatted html documents in return
- IIS (ISAPI)
- Apache
- IBM WebSphere
- Xitami
- There are any number of free web servers, and most of them do the same kind of job, provide access to server-side content/functionality

Sessions and Cookies

- Web pages are ‘stateless’
- When moving from one page to another, the state of the previous page is lost
- Can be managed by constantly transferring form values from page to page (time consuming, error prone and non dynamic)
- Sessions are written to server memory and / or hard disk
- Used to store key data, such as record id’s for application users
- Controlled by the scripting language / server
- Creates an inherent link between each browser window and the web-application state for a given user
- Think of it like calling your phone company with a query – you have to tell them who they are so that they can deal with your specific client data amongst the thousands of queries they get per day

Sessions and Cookies cont…



Development of Applications

- Browser presentation layer
  - XHTML
- Application server layer
  - ASP, Visual Basic, VBScript, Java, JavaScript, ColdFusion, CGI, Perl
- Database layer
  - Web-enabled database software
  - MySQL, Access, SQL Server, Oracle. Postgres
  - DSNs
  - Enable database access through application layer (ODBC)

Advantages of Server-side Web Development

- Have access to databases
- Have access to scripting languages
- The infrastructure for data services and code libraries in existing applications can be largely re-used for web dev
- Web-sites with large amounts of information turn over can have form-driven management interfaces
- People need only write content, then click submit
- While client-side systems have solid data output capabilities (XML and XSLT), they do not have the same data input and dynamic querying functionality

Issues with server-side web development

- Configuring *some* scripting languages to work with *some* web servers can take some work
- Scripts must reside inside the web-server folder
- Scripts must be processed through the web-server – cannot be launched directly from the file system
- In other words, more tools and infrastructure required in order for systems to work

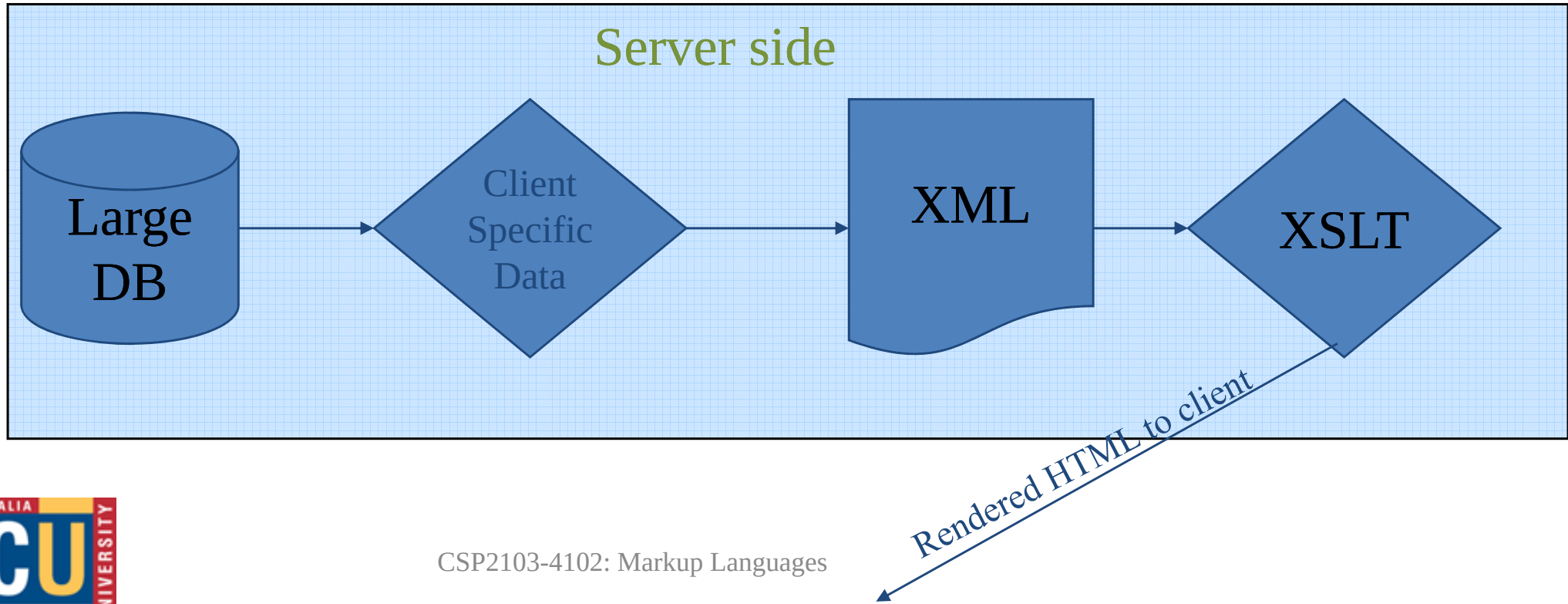
Security concerns

- Due the integration of server-side tools with Operating System, server / scripting tools present a point of attack
- More and more features creeping into web servers / scripting tools
  - Email
  - File upload
  - Remote admin
  - GUI into database servers
  - SQL Injection
- If server / scripting tools compromised, OS may be exposed
- Some servers plagued by security concerns

Server-side systems and XML

Scenario 1

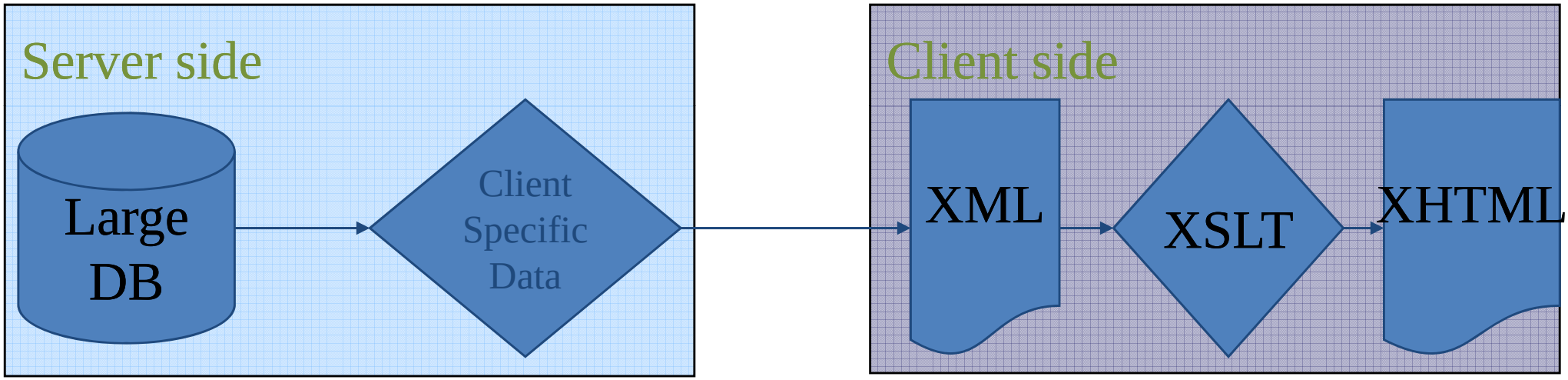
- Large database provides client-specific records
- Data could be exported to XML document on server-side
- Custom XSL and DTD for client's recordset could exist on server



Server-side systems and XML

Scenario 2

- Large database provides client-specific records
- Data could be exported to XML document over the web to client-side
- Custom XSLT and DTD for client's recordset could exist on client system (ready to deliver to CD or other client-side technology)



When to Use Server-side and XML technologies

- Good for storing client-specific settings
- Delivering customised data to a client in a useful, easy to manipulate form
- Delivering specific sub-sets of data from much larger data sets (such as delivery country-specific stories from an international news service)
- Essentially, uses and implementation only limited to creativity of developers and support tools (such as XML and XSLT parsers)
- Server-side scripting gives the management and control functionality, XML gives the delivery and end-user customizability aspects

Conclusion

- Large number of language options
- Large number of server options
- Large number of database options
- Developers do not need to be experts at individual language, but must understand logic of script execution / server interaction
- Allows developers to becoming familiar with new languages rapidly
- Security concerns paramount
- Know the right environment for the right job
- Flexibility and the ability to learn new technology rapidly is the key to employment in the field