

# Chapter 5 Documents and Websites – structure, description languages

- 5.1 Web Pages and Websites
- 5.2 Static Architecture - HTML
- 5.3 DHTML Architecture – CSS, DOM, JavaScript, VBScript, Flash, Ajax
- 5.4 High Level Languages – Java, XML
- 5.5 Dynamic Page Architecture – SSI, ASP, PHP
- 5.6 Advanced Management Architecture

■ References



by Professor Vasile AVRAM, PhD



## 5.1 Web pages and Web sites

The documents for World Wide Web (www) are known as Web pages and they are stored on an Internet server and displayed by a Web browser on your computer. The Web page is the basic element of a web site. Web browsers display Web pages by interpreting the special Hyper Text Markup Language (HTM or HTML) tags which are used to encode Web pages with display information. The formatting of Web pages is controlled by a collection of markup codes called HTML tags that marks off parts of Web page to display in certain style.

A Web site is defined as a collection of files that are linked to a central Web page, made available via the Web (the pages forms a cohesive collection of information) as entry point for the website. The Web server is a type of server dedicated to storing, transmitting and receiving the Web pages and Web related files (such GIF and JPEG graphics, AVI sound and images and so on).



## 5.1 Web pages and Web sites



**Web document layers:**

- 1) Content layer;**
- 2) Presentation layer;**
- 3) Behavior layer.**

**Depending on the way a site interacts with the end user and reacts to user actions, how pages delivered as answer to user request realized (existing or generated), and what type of resources the site is able to manipulate the site architectures can be grouped in the following categories of architectures:**

- Static Architecture;**
- DHTML Architecture;**
- High Level Languages based Architecture;**
- Dynamic Pages Architecture;**
- Advanced Management Architecture;**
- Multi-tier (three tiers) Architecture.**



## 5.2 Static Architecture

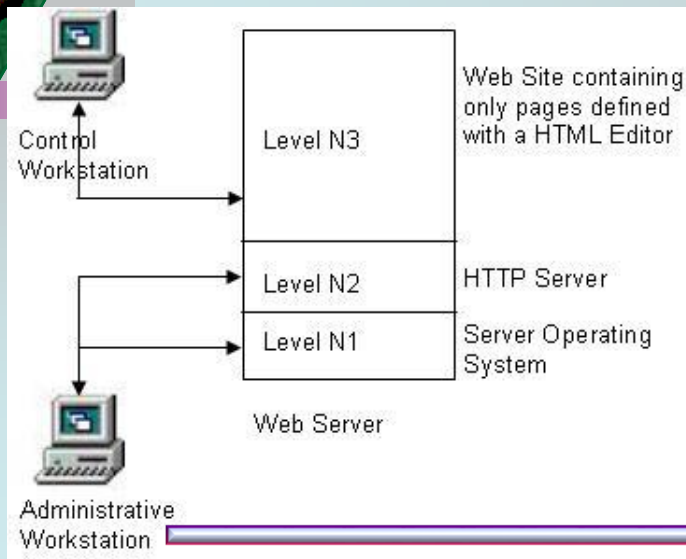


Figure 5.1 The functional Structure of a Site with Static Architecture

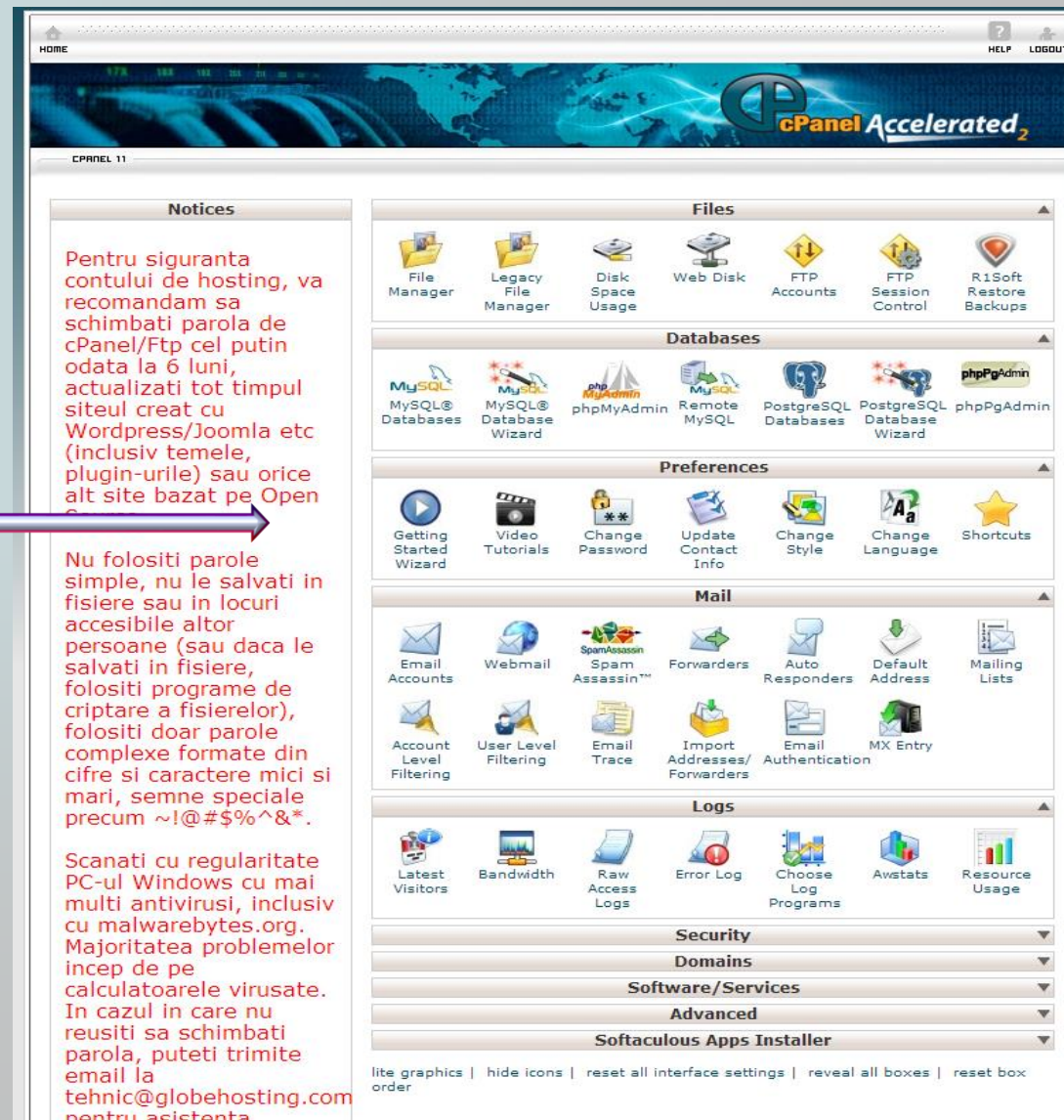


Figure 5.2 An Example of Control Panel for the Administrative Workstation (partly)



## 5.2 HTML Document

The commands for displaying text use their own language called HyperText Markup Language, or HTML.

HTML is nothing more than a coding system that combines formatting information in textual form with the readable text of a document.

In practical terms, HTML is a collection of platform-independent styles (indicated by markup tags) that defines the various components of a World Wide Web document.

HTML was invented by Tim Berners-Lee while at CERN, the European Laboratory for Particle Physics in Geneva (actually at MIT - USA and founder of World Wide Web Consortium, [www.w3C.org](http://www.w3C.org)).

HTML files -> Plain-text.

Editors: text or WYSIWYG editors.



## 5.2 HTML Document

HTML5 has been created on the foundations of HTML 4 with backward compatibilities fully insured and focused on developers. It insures also support to the browser in handling the interpretation of errors caused by incorrect markup implementation.

HTML5, give us a standard for how creating web applications with powerful APIs (Application Programming Interface) for different things such as canvas, drag and drop, offline storage, native video in the browser, and all these are realized having in mind the security concerns





## 5.2 HTML Document

HTML tags (markers), expressed following the next generalized syntax:

***<Tag\_name>*** Text associated /content associated ... [ ***< / Tag\_name>*** ]

Tags can have attributes (or parameters) that can take a finite number of specified values and whose syntax takes the format:

***<Tag\_name attribut1=" value 1" attribut2="value 2" ... >***







## 5.2 HTML Document

**HTML document contains hyperlinks, the embedded links to other documents on the web, that allows defining pathways between documents and surfing on the web.**

**Hyperlinks contains several pieces of vital information, that instruct the Web browser where to go for content, such as:**

- the protocol to use (generally HTTP);
- the server to request the document from;
- the path on the server to the document;
- the document's name (optional).

**The information is assembled together in an URL (Uniform Resource Locator) or URI.**





## 5.2 HTML Document

`<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"  
"http://www.w3.org/TR/html4/loose.dtd">`

This tag specifies the following information:

The document's top tag level is HTML (html);

The document adheres to the formal public identifier (FPI) "W3C HTML 4.01 English" standards (PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN");

The full DTD can be found at the URI <http://www.w3.org/TR/html4/loose.dtd>.



The first tag for a HTML 5 documents is simply:

`<!DOCTYPE html>`

The HTML5 is based on various design principles (addressing compatibility, utility, interoperability, and universal access) defined by the specification of the group WHATWG (Web Hypertext Application Technology Working Group) and is the result of cooperation between three important organizations: WHATWG, W3C, and IETF (Internet Engineering Task Force) [LAS-11]. Some of the new or enhanced functions available HTML5 refer to improved semantics, forms, canvas drawing, drag and drop, limited local storage, page-to-page messaging, desktop notifications, video and audio, web sockets, geolocation, history, and microdata.:



## 5.2 HTML Document Structure - general

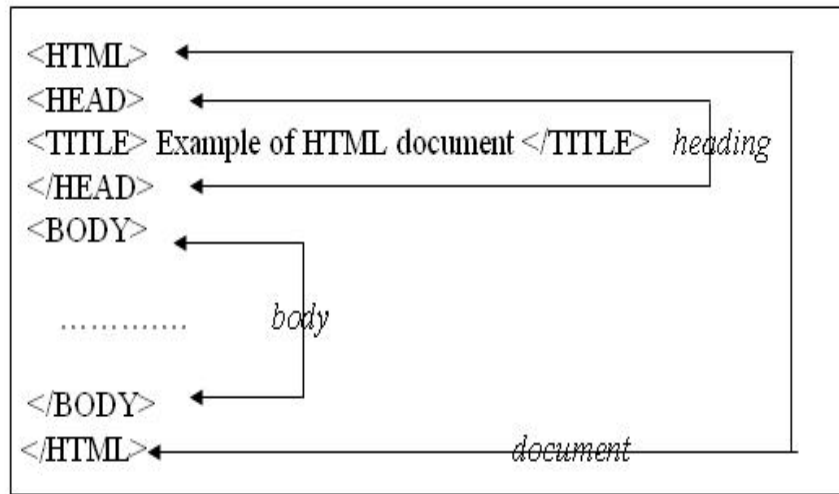


Figure 5.3 The HTML Document Structure

The minimal required elements of a web page delivered to a web browser by the server at user request are the

`<html>`;

`<head>` with `<title>`, meta-tags, style and script sections;

`<body>` tags;

and their corresponding end tags, even included in.

Learn HTML (and Web technologies):

<http://www.w3schools.com>

World Wide Web Consortium:

<http://www.w3c.org>



## 5.2 HTML Document Structure – heading

The title element contains the document title and identifies its content in a global context. The title is typically displayed in the title bar at the top of the browser window, but not inside the window itself:

`<title>a given meaningful title for content</title>`

The typical syntax of a metatag is:

`<meta name="meta-name" content="data-content">`

Examples:

`<meta name="Keywords" content="JavaScript, javascript, HTML, function">`

`<meta name="resource-type" content="document">`

`<meta name="revisit-after" content="30 Days">`

`<meta name="Classification" content="Education">`

`<meta name="Robots" content="INDEX, NOFOLLOW">`

`<meta name="distribution" content="Global">`

`<meta name="rating" content="Safe For Kids">`

`<meta name="Author" content="Vasile Avram">`

`<base href="http://www.avrams.ro">`





## 5.2 HTML Document Structure – heading

The metatag syntax was simplified for HTML5 and can be as the previous or of the form:

`<meta data-content>`

Examples (this code is for both mobile and desktop browsers):

```
<!DOCTYPE html><html><head>
```

```
<meta name="viewport" content="width=410, height=400, initial-scale=1">
```

```
<meta content="text/html; charset=utf-8" http-equiv="Content-Type" />
```

```
<title>Body Mass Index (BMI) Computation</title>
```

```
<link rel="stylesheet" type="text/css" href="themes/inspector.min.css" >
```

```
<link rel="stylesheet" type="text/css" href="jquery.mobile.min.css" >
```

```
<script type="text/javascript" src="jquery.min.js"></script>
```

```
<script type="text/javascript " src="jquery.mobile.min.js"></script>
```





## 5.2 HTML Document Structure – body

A vital element to realize a website used to link together the component resources in a cohesive collection, and to define the navigational pathways inside/outside the site, is represented by the anchor tag (hyperlink) whose general syntax is

```
<a href="resource-name" ...>link-name </a>
```

where the *resource-name* is the relative/ absolute pathname (expressed as URL/URI) to the wanted resource and *link-name* is the object displayed on which the user can click to access the resource (can be text, graphic, etc). The obsolete ... denote the common and specific attributes the tag may have.



## 5.2 HTML Document Structure – HTML5

The HTML5 defines new elements for sectioning the content (Figure 5.5) as semantic markups, such as:

- header – header content for a page or a section of the page;
- hgroup – group the headings;
- nav – navigation menu;
- article – independent article content;
- section – a section in a web page;
- aside – vertical panel to left/ right side of the page;
- footer – footer content for a page or a section of the page.

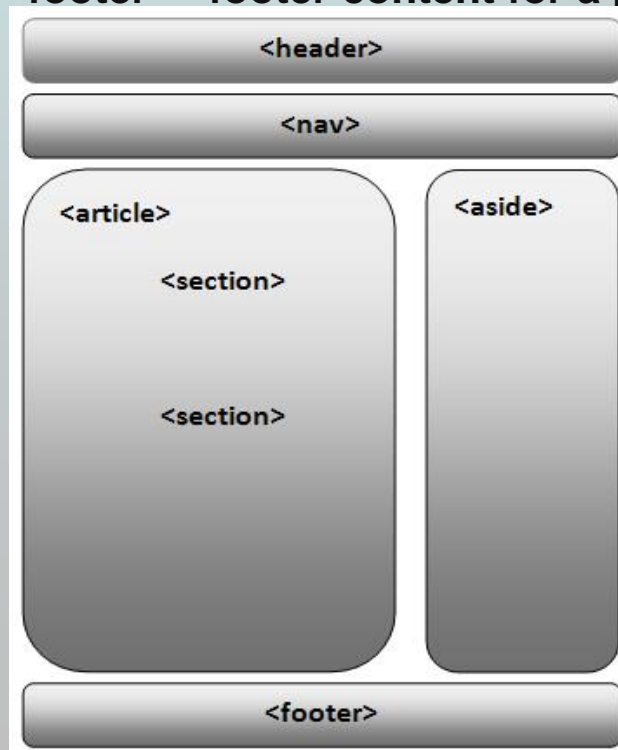


Figure 5.5 HTML5 Enhanced Page Structure

## 5.2 HTML Document Structure – HTML5



Figure 5. 6 Using Semantic Tags to Describe the Web Page Layout

Page can be found at <http://avrams.ro/html5/html5-example-web-page.html>

Use View Page Source from your browser to see the code;

Access associated CSS3 file at <http://avrams.ro/html5/html5-example.css>



## 5.3 DHTML Document Structure

DHTML stands for Dynamic HTML and is not a W3C standard. It was defined by Netscape and Microsoft as a “marketing term” describing a new technology that the generation 5.x (and following) of browser must support. It is a combination of “HTML/XHTML, style sheets and scripts that allows documents to be animated“. This is possible by intermediate of HTML DOM, a W3C standard that defines a standard set of objects for HTML and a standard way to access and manipulate these HTML objects. The HTML DOM specifies the objects together with their associated properties (or attributes, generally the equivalent to tag attributes such as *id*, *name*, *alt*, *title* etc), and where appropriate, methods that can be invoked for the object, such as *blur()*, *focus()*, *click()* etc.

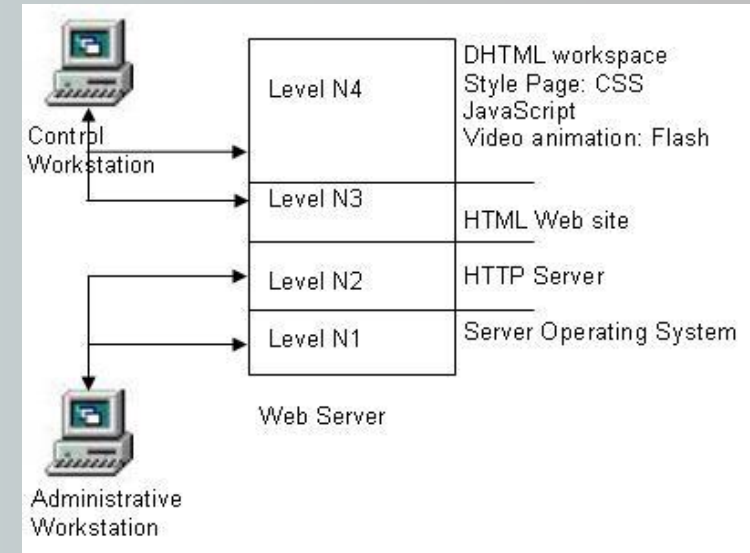


Figure 5.7 The functional structure of a site with DHTML architecture

DHTML represents the usage of a lot of languages that allows realizing animated pages and presentations, computations, and the control and validation of primary fields in fill-in forms, such as:

- HTML/XHTML;
- Style Pages (Cascading Style Sheets);
- Scripts (JavaScript, VBScript etc.).

## 5.3 DHTML Document Structure - CSS - Cascading Style Sheets



CSS is a style language that defines layout of HTML documents. Styles are grouped together in style sheets and are normally stored in external files with a .css extension.

CSS style can be placed (and has the priority, from higher-to-lower):

1. inline;
2. To the heading tag as global;
3. By linking the style page:

```
<LINK REL="stylesheet" TYPE="text/css" HREF="styledocumentname.css">
```

4. Browser default.

Style for avrams.ro <http://avrams.ro/styles/sitestyl.css>

Style for boxes <http://www.avrams.ro/styles/cboxes.css>

Access a HTML5 page associated CSS3 file at <http://avrams.ro/html5/html5-example.css>



## 5.3 DHTML Document Structure - CSS - Cascading Style Sheets

CSS syntax is very simple and is made up of three elements called *selector*, *property*, and *value*:

*selector* {*property*: *value*; *property*: *value* ...}

Selector: id selector (*#name*), class selector (*.name*)

Comments */\** and *\*/*

Example:

```
/* This is a partly content of a stylesheet file */
BODY { BACKGROUND: white; COLOR: black; FONT-FAMILY: sans-serif; }
/* The following are the four link states: unvisited or normal, visited, the mouse is over it, and a link the moment is clicked */
A:link { BACKGROUND: none transparent scroll repeat 0% 0%; COLOR: #00e; }/* unvisited */
A:visited { BACKGROUND: none transparent scroll repeat 0% 0%; COLOR: #529; }/* visited */
A:hover{background:transparent none repeat scroll 0% 0%;color:#999999;}/* mouse over it*/
A:active { background: none transparent scroll repeat 0% 0%; COLOR: #00e;}/*now clicked */
/* Defines style for the element DIV identified by intro */
DIV.intro { MARGIN-LEFT: 5%; MARGIN-RIGHT: 5%; FONT-STYLE: italic; }
PRE { FONT-FAMILY: monospace; }
A:link IMG { BORDER-TOP-STYLE: none; BORDER-RIGHT-STYLE: none; BORDER-LEFT-STYLE: none; BORDER-BOTTOM-STYLE: none; }
A:visited IMG { BORDER-TOP-STYLE: none; BORDER-RIGHT-STYLE: none; BORDER-LEFT-STYLE: none; BORDER-BOTTOM-STYLE: none; }
@media All { A IMG { } }
UL.toc { LIST-STYLE-TYPE: none ;}
.hideme { DISPLAY: none; }
.avbkgtop{border:medium none;background-position:center;BACKGROUND-IMAGE:url('http://www.avrams.ro/imgs/tt077.jpg');FONT-
FAMILY:'Times New Roman'; BACKGROUND-COLOR:transparent;}
#menuv{padding:0;margin:0;height:1em;list-style-type:none;border-left:1px solid #c0c0c0;color: #bbbbbb;font-family:Verdana;font-weight:
normal;font-size: xx-small}
#menuv li{float:left;width:8em;height:1em;line-height:1em;border-right:1px solid #c0c0c0;position:relative;text-
align:center;color:#efbb22;font-family: Verdana;font-weight: normal;font-size:xx-small;}
#menuv li a, #menuv li a:visited{display:block;text-decoration:none;color:#efbb22}
#menuv li a span, #menuv li a:visited span{display:none}
#menuv li a:hover{border:0px none;color:#c0c0c0}
ul, li { margin-left:0px}
```

## 5.3 DHTML Document Structure - Scripts and Document Object Model

Scripts – source programs expressed in a scripting language:

`<script> ... </script>`

Can act: server side, client side (placement), both.

(JavaScript, PHP, VBScript, Python, Perl etc.)

In Document Object Model the HTML/XHTML documents viewed as a collection of objects (having attributes/properties and methods) and provides access to every element in a document. Every element is modeled in a web browser as a DOM *node*, and the nodes make up the DOM *tree* describing the relationships between elements in a *child-parent* fashion. The children of the same node (having the same parent) referred to as *siblings*. A node can have multiple children but only one parent. Because of that access, any element may be modified by a snippet of JavaScript. Elements are easily accessed by use of an *id* attribute (that must be unique within a given document) and a method of the document object. The *document* object is the parent of all the other objects in an HTML/XHTML document (Figure 5.8).

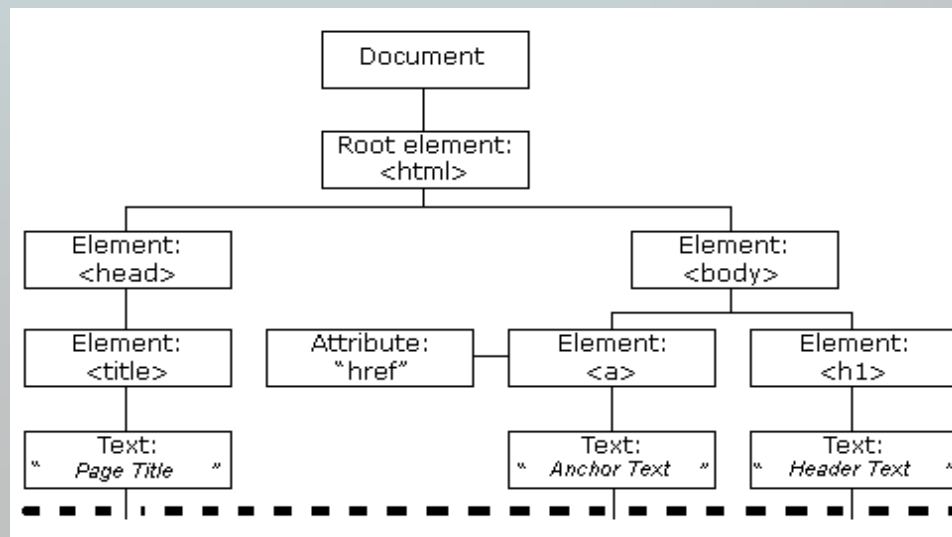



Figure 5.8 The Tree Structure of HTML Documents (Partly; Theoretical)


## 5.3 DHTML Document Structure - DOM

### The Objects in DHTML DOM



Object	Description
<b>window</b>	The top level object in the DHTML DOM. It contains information about the window and the frames. The objects listed below are the children of the window object.
<b>document</b>	Represents the HTML document, and is used to access the HTML elements inside the document
<b>frames</b>	Represents the frameset
<b>history</b>	Keeps track of the sites visited by the browser object.
<b>navigator</b>	Contains information about the user's browser
<b>location</b>	Contains the URL of the rendered document
<b>event</b>	Contains information about events that occurs
<b>screen</b>	Contains information about the computer screen for the computer on which the browser is

### The Methods of Document Objects



Method	Description
<b>close()</b>	Closes an output stream opened with the document.open() method, and displays the collected data
<b>getElementById("id")</b>	Returns a reference to the first object (node) with the specified "id"
<b>getElementsByName("name")</b>	Returns a collection of objects with the specified "name"
<b>getElementsByTagName("tagname")</b>	Returns a collection of objects with the specified "tagname"
<b>open()</b>	Opens a stream to collect the output from any document.write() or document.writeln() methods
<b>write()</b>	Writes HTML expressions or JavaScript code to a document
<b>writeln()</b>	Identical to the write() method, with the addition of writing a new line character after each expression

## 5.3 DHTML Document Structure - DOM

### The Properties of a HTML Object

Property	Description
<b>aLink</b>	Sets the color of hyperlinks as they are clicked.
<b>Background image</b>	Provides the path to a background image for the page.
<b>bgcolor</b>	Sets a background color for the page.
<b>bgProperties</b>	Sets whether the background for the page will be scrolling (default) or fixed (watermark)
<b>bottomMargin</b>	Sets the height of the blank margin at the bottom of the page.
<b>charset</b>	Selects the character set for the page.
<b>defaultClientScript</b>	Sets the default client scripting language.
<b>dir</b>	Sets the reading order of page objects.
<b>keywords</b>	Adds keywords to the META KEYWORDS tag in the <HEAD> of your document
<b>leftMargin</b>	Sets the width of the blank margin on the left side of the page.
<b>link</b>	Sets the default color of hyperlinks before they are clicked.
<b>pageLayout</b>	Selects whether page components added in design view will be positioned in-line as they occur on the page or positioned at specified locations (enables the positioning grid).
<b>rightMargin</b>	Sets the width of the blank margin on the right side of the page.
<b>showGrid</b>	Determines whether the positioning grid will appear in Design View.
<b>targetSchema</b>	Sets the minimum version of HTML required to view this page, and (in some cases) the preferred document object model (DOM) for client scripts on the page.
<b>text</b>	Sets the default color for foreground text on the page.
<b>title</b>	Provides the text string inserted between the <TITLE> and </TITLE> tags in the page HEAD.
<b>topMargin</b>	Sets the height of the blank margin at the top of the page.
<b>vLink</b>	Sets the default color of hyperlinks that have been clicked.







## 5.3 DHTML Document Structure – Scripting

JavaScript and VBScript -> <http://www.avrams.ro/compute-easter-date.html>



```
<script type="text/vbscript" language="vbscript"> <!--
Function Easter_Date(Wanted_Year)
Dim D, E
D = ((Wanted_Year Mod 19) * 19 + 15) Mod 30
E = (((Wanted_Year Mod 4) * 2 + (Wanted_Year Mod 7) * 4) + 6 * D + 6) Mod 7
Easter_Date = DateAdd("d", (D + E+0.), CDate("04/04/" + Trim(Wanted_Year)))
End Function
function vbvalidComp()
if not isnumeric(trim(wantedYear.value)) then msgbox "Err.1. Wrong value for
year: is not a number!", "Easter Date"
wantedYear.focus
exit function
end if
if len(trim(wantedYear.value))<3 or len(trim(wantedYear.value))>4 then msgbox
"Err.2. Wrong value for year: is a to little/large number (min 3 and max 4 digits
allowed)!" + Chr(10) + chr(13) + "[This limitation is given by the way in which
Microsoft implements the functions]" + Chr(10) + chr(13) + "[DateAdd(...) and
CDate(...)]. Eliminate this test and enjoy computing!", "Easter Date"
wantedYear.focus
exit function
end if
easterDate.value=Easter_Date(trim(wantedYear.value))
end function --> </script>
```

```
<script type="text/javascript" language="javascript">
<!--
function easter_datejs(Wanted_Year){
var D; var E;
if (Wanted_Year<0){ alert("The value for Year must be a positive number!");
return -1; }
D = ((Wanted_Year % 19) * 19 + 15) % 30;
E = (((Wanted_Year % 4) * 2 + (Wanted_Year % 7) * 4) + 6 * D + 6) % 7;
D=D+E+4;
if(D>30)
{ easterDate.value='5/'+(D-30.)+'/'+Wanted_Year; }
else
{ easterDate.value='4/'+(D)+'/'+Wanted_Year; }
} --> </script>
```



## 5.3 DHTML Document Structure – Scripting

JavaScript and VBScript -> <http://www.avrams.ro/compute-easter-date.html>

### Determine the Easter Date in a Wanted Year

Pope pleases the great mathematician Gauss to tell him when Easter will be in a wanted year.

Gauss says that Easter will be always on: 4 April+D days+E days where:

D is determined by following the steps

1 - the year is divided by 19;

2 - the remainder is multiplied by 19;

3 - to the result of step two add fix factor 15;

4 - the sum of values obtained in the steps 1 to 3 is divided to 30 and the remainder is D;

E is determined by following the steps:

1 - the year is divided by 4 and the remainder will be multiplied by 2;

2 - the year is divided by 4 and the remainder will be multiplied by 4;

3 - compute the sum of values obtained to step 1 and 2;

4 - to the sum add 6\*D and to product add 6;

5 - the total sum is divided by 7 and the remainder will be E.

Note: This Formulas Compute The Easter Date For Orthodox (even Pope is Catholic!)

A code that implements this algorithm is implemented in VBScript and in JavaScript in that page.

For the VBScript solution the year must be represented on three or four digits only. It is a restriction imposed by the Microsoft implementation of the functions DateAdd() and CDate().

#### VBScript based solution.

The button VBScript Solution from the form do not react if the browser is not a Microsoft Internet Explorer version.

#### JavaScript based solution.

The button JavaScript Solution must work in any circumstances (is a javascript solution).

Type the year you want in the box Wanted Year and then press the button corresponding to the solution you want check.

Wanted Year:

Easter Date:

Figure 5.10 The Algorithm to Determine Easter Date

## 5.3 DHTML Document Structure – Ajax

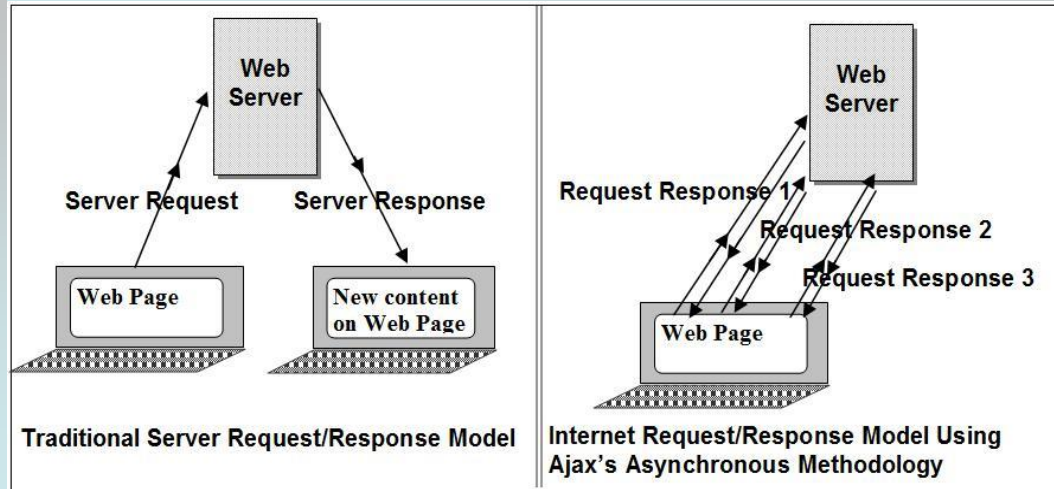



Figure 5. 13 Traditional Server Request/Response Model vs. Ajax Methodology

### Internet Technologies for Business (excerpt from Chapter 3)

[Page 1](#) | [Page 2](#) | [Page 3](#)

(a menu like allowing navigation)

#### Page 1

Asynchronous JavaScript and XML  uses the JavaScript-based XMLHttpRequest object to fire requests to web server asynchronously (or without having to refresh the page). Figure 3.1 shows the usage of traditional server request/response model, the most used technology in Internet, in which the web server responds with a new content for the page at the user request, together with the use of Ajax asynchronous methodology in which the server responds with changes within the web page as answer to the user requests.

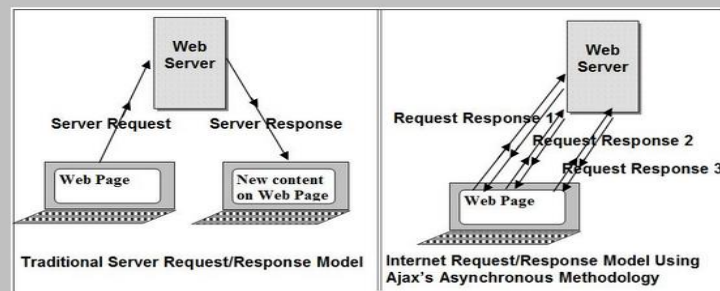


Figure 3.1 Ajax asynchronous methodology

## 5.3 DHTML Document Structure – Ajax

```
<!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>
<meta content="text/html; charset=windows-1252" http-equiv="Content-Type" />
<title>Ajax - Change Page Content</title>
<script type="text/javascript"><!--
var xmlhttp; xmlhttp=null; xmlhttp=new XMLHttpRequest();
function makerequest(serverPage,objID) {
  var obj = document.getElementById(objID);
  xmlhttp.open("GET", serverPage);
  xmlhttp.onreadystatechange = function() {
    if (xmlhttp.readyState == 4 && (xmlhttp.status == 0 || xmlhttp.status == 200)) { obj.innerHTML = xmlhttp.responseText; }
  }
  xmlhttp.send(null);} →</script>
</head>
<body onload="makerequest('http://www.avrams.ro/pgs/ajax-1.html','repme')">
<a name="none"></a>
<div align="center" style=" background-color: silver; border: 6px blue groove">
  <h1 style=" color:green">Internet Technologies for Business (excerpt from Chapter 3)</h1>
  <a href="#none" onclick="makerequest('http://www.avrams.ro/pgs/ajax-1.html','repme');return false;" style=" color: red; font-size:x-large">Page 1</a>
  | <a href="#none" onclick="makerequest('http://www.avrams.ro/pgs/ajax-2.html','repme');return false;" style=" color: yellow; font-size:x-large">Page 2</a>
  | <a href="#none" onclick="makerequest('http://www.avrams.ro/pgs/ajax-3.html','repme');return false;" style=" color: blue; font-size:x-large">Page 3</a>
  <p style="text-align:center; font-size:x-small; font-style:italic;">(a menu like allowing navigation)</p>
  <hr style="height:6px; background:blue"/>
  <div id="repme" name="repme" style="width: 99.9%">This content will be replaced by the requested page</div>
</div>
<script src="http://www.google-analytics.com/urchin.js" type="text/javascript">
<script type="text/javascript">
  _uacct = "UA-1653633-1";
  urchinTracker();
</script>
</body>
</html>
```

## 5.4 High Level Languages based Architecture

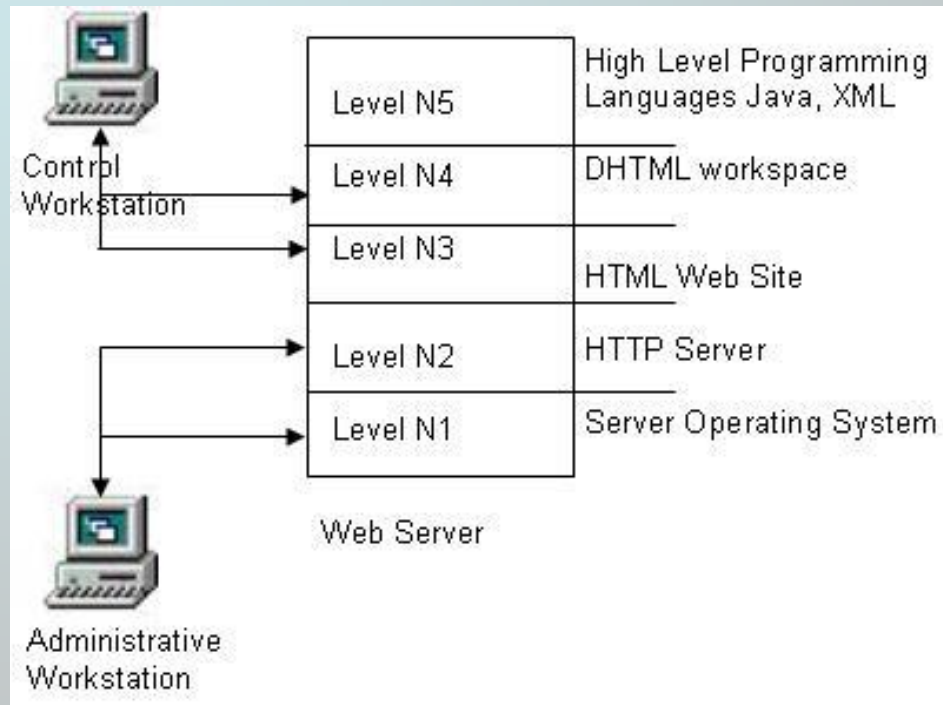


Figure 5.16 High Level Languages Based Architecture

## 5.4 High Level Languages based Architecture - Java

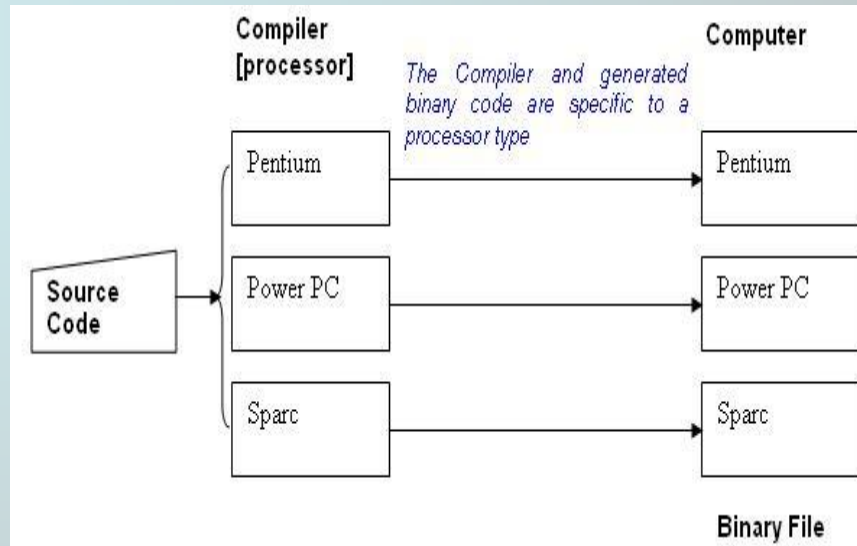


Figure 5.17 Programs “classic” Compiled

-Bytes codes

-Applets

```
<Applet Code="anapplet.class" width="200" Height="100">
</Applet>
```

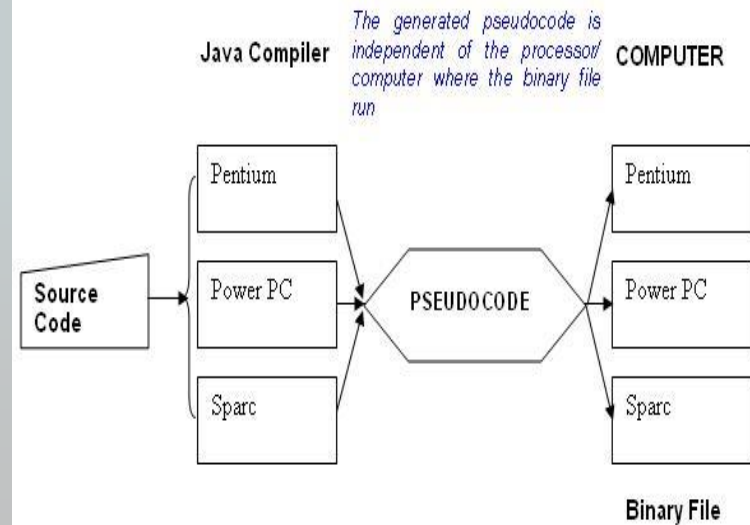


Figure 5.18 Compiled Java Programs





## 5.4 High Level Languages based Architecture - Applet

Attribute	Explanation	Example
<b>Code</b>	Name of class file	Code="anapplet.class"
<b>Width=n</b>	n=Width of applet	Width=200
<b>Height=n</b>	n=Height of applet	Height=100
<b>Codebase</b>	Library where the applet is stored. If the applet is in same directory as your page this can be omitted.	Codebase="applets/"
<b>Alt="Text"</b>	Text that will be shown in browsers where the ability to show applets has been turned off.	alt="Menu Applet"
<b>Name=Name</b>	Assigning a name to an applet can be used when applets should communicate with each other.	Name="starter"
<b>Align=</b> Left Right Top Texttop Middle Absmiddle Baseline Bottom Absbottom	Justifies the applet according to the text and images surrounding it.	Align=Right
<b>Vspace=n</b>	Space over and under the applet.	Vspace=20
<b>Hspace=n</b>	Space to the left and right of applet.	Hspace=40





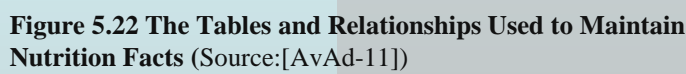


## 5.4 High Level Languages based Architecture – XML (eXtensible Markup Language)

XML was developed by the XML Working Group (initially called SGML Editorial Review Board) from W3C (World Wide Web Consortium) in 1996. The initial specification of the language establishes the following objectives for this:

- XML must be directly usable in Internet;
- XML must support a variety of applications;
- XML must be compatible with SGML;
- The XML pages creation must be as simple as possible and rapidly done;
- XML may not contains facultative functions;
- The XML must have a high readable degree;
- The syntax must be formal and concise;
- The code concision is an element of little importance.






**Figure 5.23 The DTD Definition for Diet Database (Source: [avrams.ro]/diet-01.dtd)**

## 5.4 High Level Languages based Architecture – XML



```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE Diet SYSTEM "http://www.avrams.ro/diet-01.dtd">
<Diet>
  <Categories><catid>1</catid><catdesc>Milk and Milk products</catdesc></Categories>
  <Categories><catid>2</catid><catdesc>Nuts and Seeds</catdesc></Categories>
  <Categories><catid>3</catid><catdesc>Grains, Breads, and Pasta</catdesc></Categories>
  <Categories><catid>4</catid><catdesc>Soups</catdesc></Categories>
  <Categories><catid>5</catid><catdesc>Condiments, Sauces, and Herbs</catdesc></Categories>
  <Categories><catid>6</catid><catdesc>Vegetables</catdesc></Categories>
  <Categories><catid>7</catid><catdesc>Meat, Poultry, and Fish</catdesc></Categories>
  <Categories><catid>8</catid><catdesc>Seafood</catdesc></Categories>
  <Categories><catid>9</catid><catdesc>Oils and Dressings</catdesc></Categories>
  <Categories><catid>10</catid><catdesc>Beans</catdesc></Categories>
  <Categories><catid>11</catid><catdesc>Fruit and Fruit Juices</catdesc></Categories>
  <Subcategories><catid>1</catid><subcatid>1</subcatid><subcatdesc>Milk</subcatdesc>
</Subcategories>
  <Subcategories><catid>1</catid><subcatid>2</subcatid><subcatdesc>Cream</subcatdesc>
</Subcategories>
  <Subcategories><catid>1</catid><subcatid>3</subcatid><subcatdesc>Butter</subcatdesc>
</Subcategories>
  <Subcategories><catid>2</catid><subcatid>4</subcatid><subcatdesc>Cheese</subcatdesc>
</Subcategories>
  <Products><catid>1</catid><subcatid>1</subcatid>
    <prodid>1</prodid><proddesc>Milk 1.5%</proddesc>
    <energyvalue>90</energyvalue>
    <proteins>6.8</proteins><lipids>3</lipids><carbohydrates>9</carbohydrates>
    <extraattribute>0</extraattribute>
  </Products>
  <Products><catid>1</catid><subcatid>1</subcatid>
    <prodid>2</prodid><proddesc>Milk whole</proddesc>
    <energyvalue>0</energyvalue>
    <proteins>0</proteins><lipids>0</lipids><carbohydrates>11.4</carbohydrates>
    <extraattribute>0</extraattribute>
  </Products>
    <Products><catid>1</catid><subcatid>2</subcatid>
      <prodid>1</prodid><proddesc>Cream 3%</proddesc>
      <energyvalue>10</energyvalue>
      <proteins>0</proteins><lipids>0</lipids><carbohydrates>11.7</carbohydrates>
      <extraattribute>0</extraattribute>
    </Products>
  <Products><catid>1</catid><subcatid>4</subcatid>
    <prodid>1</prodid><proddesc>Telemea de vaca</proddesc>
    <energyvalue>10</energyvalue>
    <proteins>0</proteins><lipids>0</lipids><carbohydrates>.2</carbohydrates>
    <extraattribute>0</extraattribute>
  </Products>
</Diet>
```



**Figure 5.24** The Content of Data Associated to the Structure from Figure 5.23 (Source: [avrams.ro]/diet-03.xml)

```

<!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>
<style type="text/css"> .style1 {margin-left: 2px;}</style>
<script type="text/javascript">
if (window.XMLHttpRequest)
    { // code for IE7+, Firefox, Chrome, Opera, Safari
    xmlhttp=new XMLHttpRequest();
    }else{ // code for IE6, IE5
    xmlhttp=new ActiveXObject("Microsoft.XMLHTTP");
    }
xmlhttp.open("GET","http://www.avrams.ro/dietdb-03.xml",false);
xmlhttp.send();
xmlDoc=xmlhttp.responseXML;
ctname = ""; sctname="";
xc=xmlDoc.getElementsByTagName("Categories");
xs=xmlDoc.getElementsByTagName("Subcategories");
x=xmlDoc.getElementsByTagName("Products");
i=0;
document.write("<p style='text-size: 2em; margin-left: 60px;'><strong>Show Products from
Catalog</strong><br/><hr style='margin-left: 60px;text-align:left; width: 470px;'><br/></p>");
function displayProduct(){
codpr=(x[i].getElementsByTagName("prodid")[0].childNodes[0].nodeValue);
title=(x[i].getElementsByTagName("proddesc")[0].childNodes[0].nodeValue);
ctid = (x[i].getElementsByTagName("catid")[0].childNodes[0].nodeValue);
sctid = (x[i].getElementsByTagName("subcatid")[0].childNodes[0].nodeValue);
cid = ctid; sid = sctid;
if (ctid>0) cid = ctid - 1;
if (sctid>0) sid = sctid - 1;
ctname = (xc[cid].getElementsByTagName("catdesc")[0].childNodes[0].nodeValue);
sctname = (xs[sid].getElementsByTagName("subcatdesc")[0].childNodes[0].nodeValue);
energ=(x[i].getElementsByTagName("energyvalue")[0].childNodes[0].nodeValue);
lipid=(x[i].getElementsByTagName("lipids")[0].childNodes[0].nodeValue);
prot=(x[i].getElementsByTagName("proteins")[0].childNodes[0].nodeValue);
carb=(x[i].getElementsByTagName("carbohydrates")[0].childNodes[0].nodeValue);
txt="<b>CatID:</b>" + ctid + " | " + ctname + "<br/><b>SubcatID:</b>" + sctid + " |
" + sctname + "<br/><b>ProdID:</b>" + codpr + "<br/><b>Title:</b>" + title + "<br/><b>Energy:</b>"
+ energ + "<br/><b>Lipids:</b>" + lipid + "<br/><b>Proteins:</b>" + prot + "<br/>
<b>Carbohydrates</b>" + carb;
document.getElementById("showProduct").innerHTML=txt;}
function next(){if(i<x.length-1){i++;displayProduct();}}
function previous(){if (i>0){i--;displayProduct();}}
</script></head><body onload="displayProduct()">
<div align="center" class="style1" style="width: 593px">
<div id="showProduct" style="border-style: solid; border-color: inherit; border-width: 1px; text-align:left; width: 472px;"><br/>
<input type="button" title="Previous Product" onclick="previous()" value="&larr;" />
<input type="button" title="Next Product" onclick="next()" value="&rarr;" /></div>
<script src="http://www.google-analytics.com/urchin.js" type="text/javascript"></script>
<script type="text/javascript"> _uacct = "UA-1653633-1"; urchinTracker(); </script>
</body></html>

```

### Show Products from Catalog

**CatID:**1 | Milk and Milk products  
**SubcatID:**2 | Cream  
**ProdID:** 1  
**Title:** Cream 3%  
**Energy:** 10  
**Lipids:** 0  
**Proteins:** 0  
**Carbohydrates:** 11.7



**Figure 5.25** Navigation in Data from Figure 5.24 (Source: [avrams.ro] /showproducts.html)

## 5.4 High Level Languages based Architecture – XML

- XSL (eXtensible Stylesheets Language);
- XQL (eXtended Query Language)

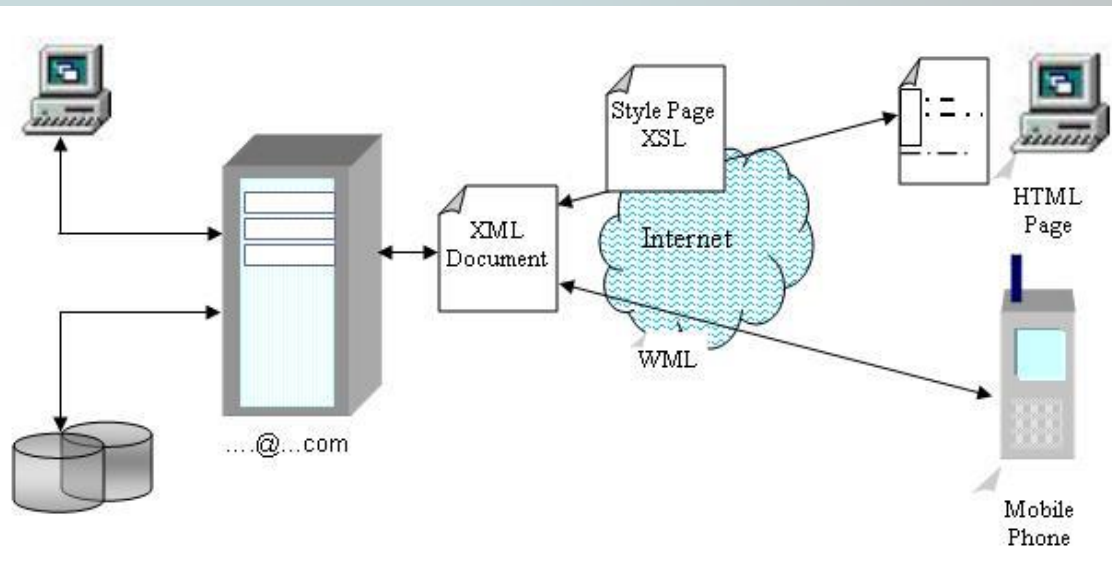


Figure 5.27 The steps of an XSL processing



## 5.5 Dynamic Pages Architecture

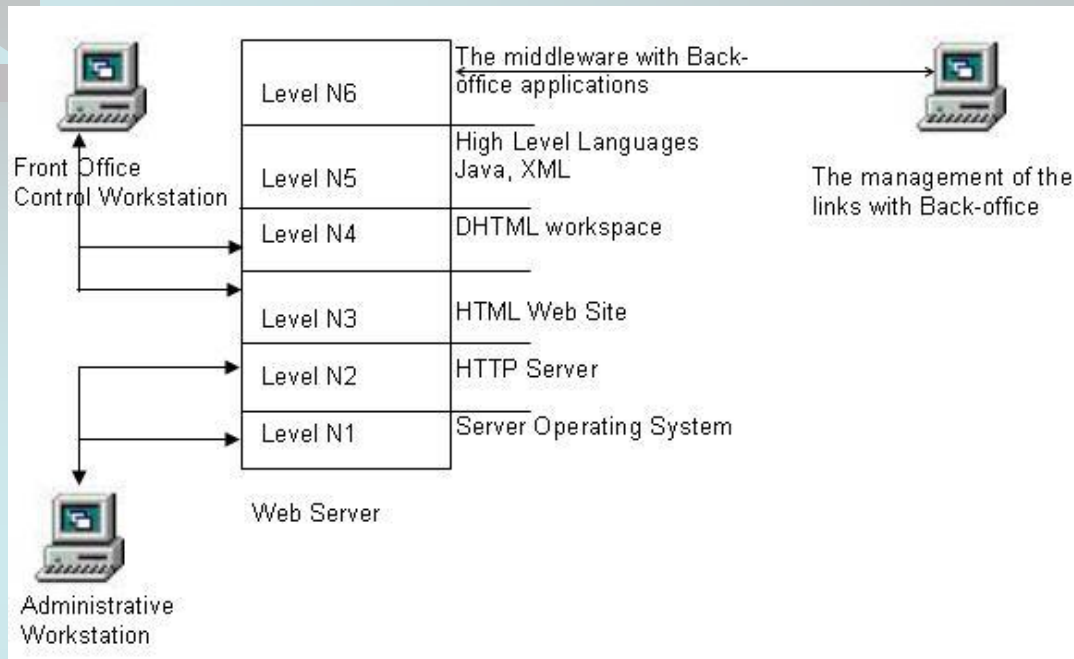


Figure 5.33 Functional Architecture for Dynamic pages

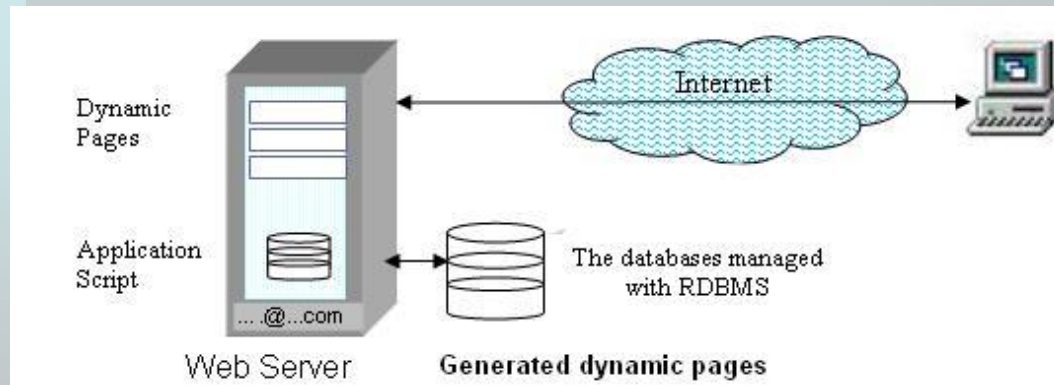


Figure 5.34 The generation of a dynamic page

**SSI- Server Side Include**

## 5.5 Dynamic Pages Architecture

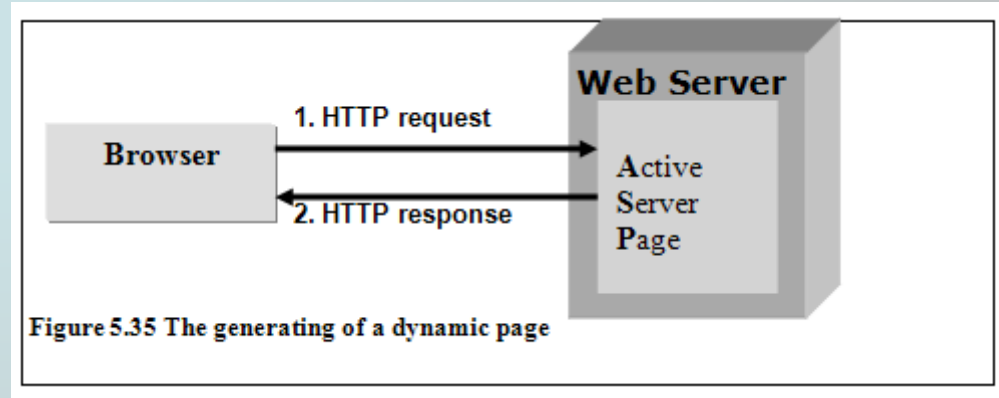


Figure 5.35 The generating of a dynamic page

**ASP – Active Server Page**





## 5.6 Advanced Management Architecture

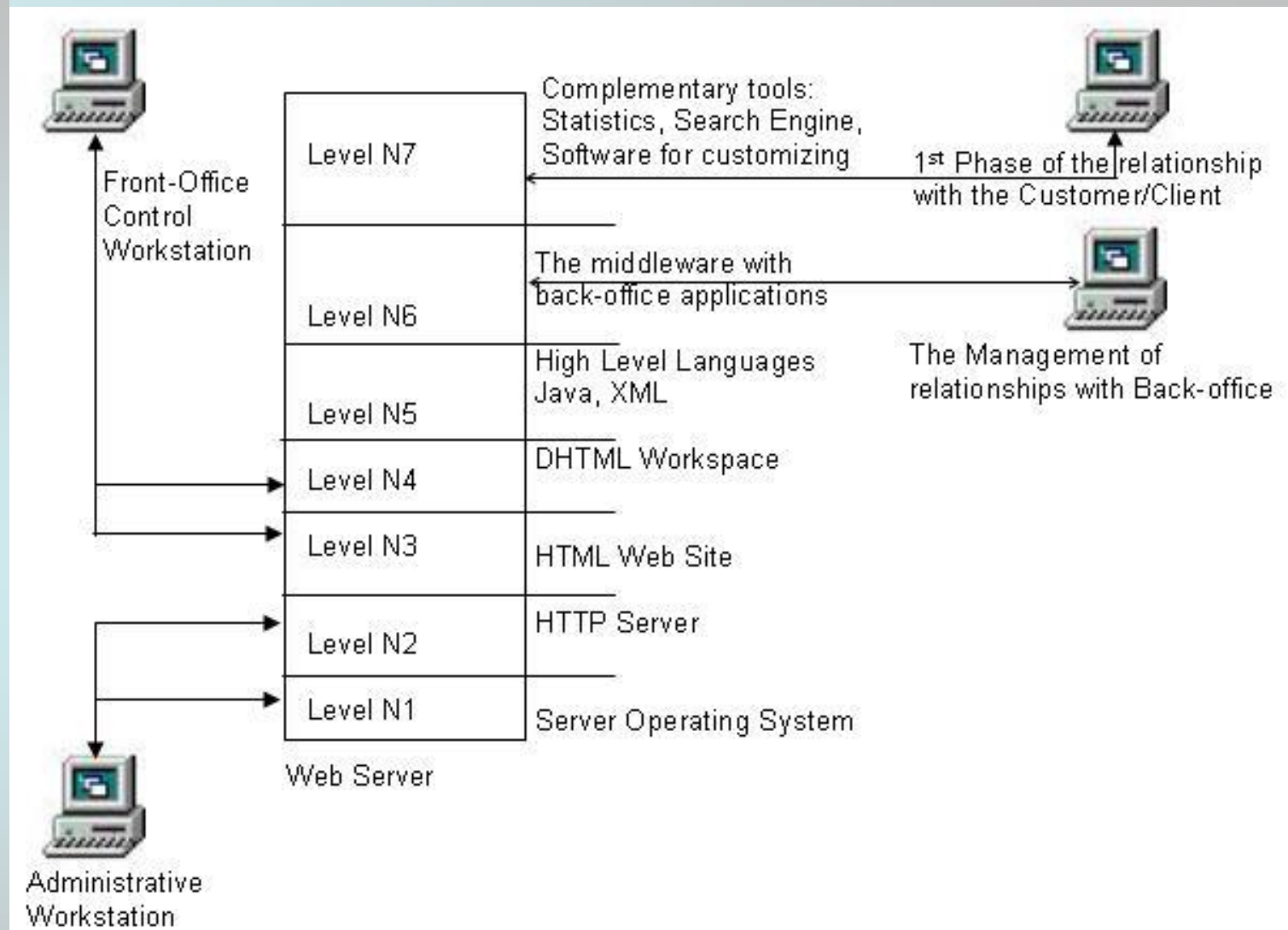


Figure 5.36 The Advanced Management Functional Architecture

## 5.6 Advanced Management Architecture

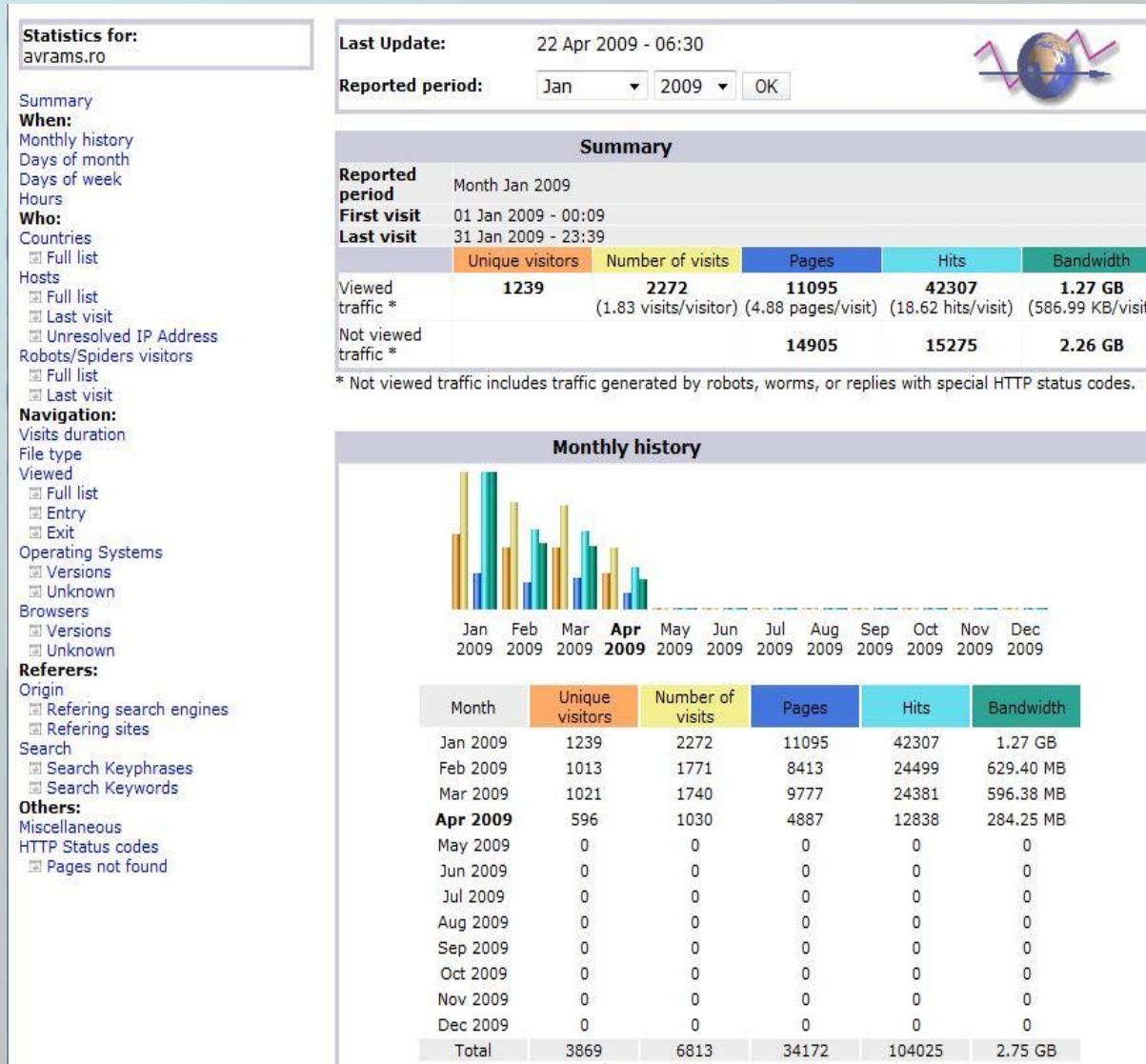


Figure 5. 37 AWStats reports (main screen)

## 5.6 Advanced Management Architecture

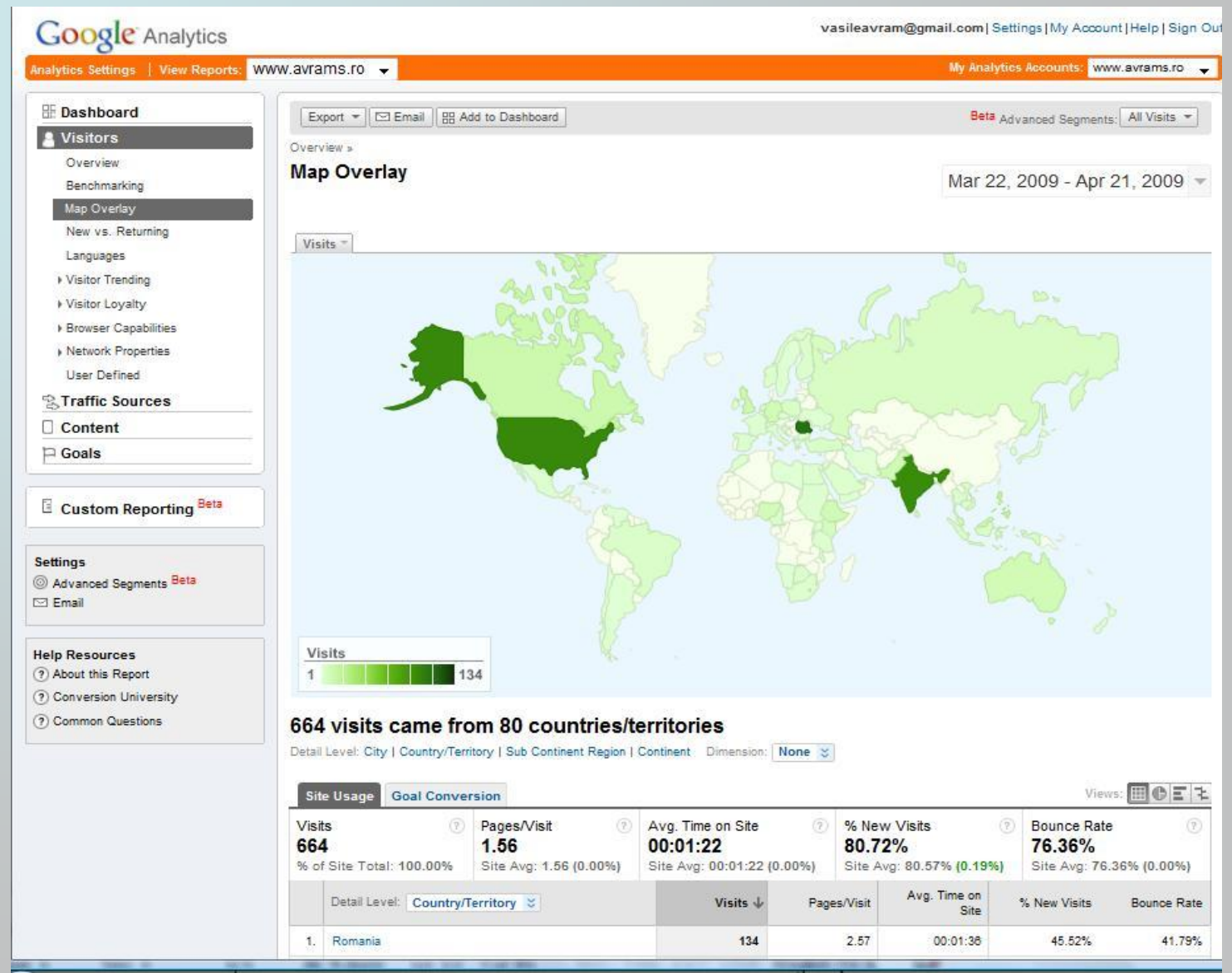
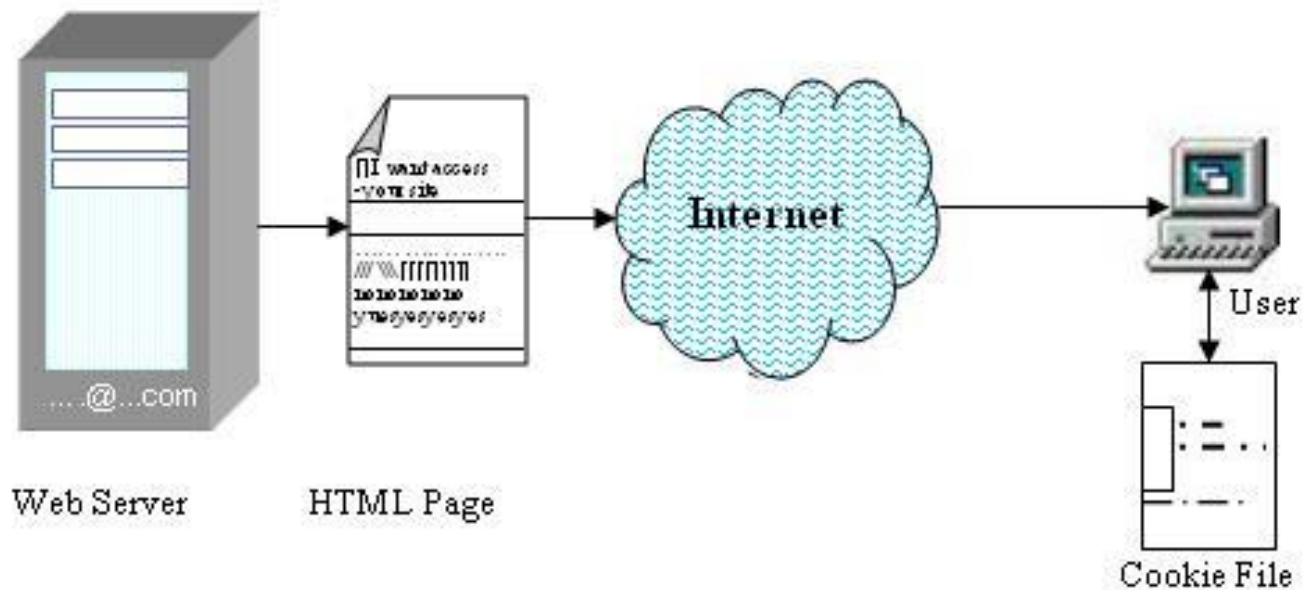


Figure 5. 38 The Google Analytics reports

## 5.6 Advanced Management Architecture

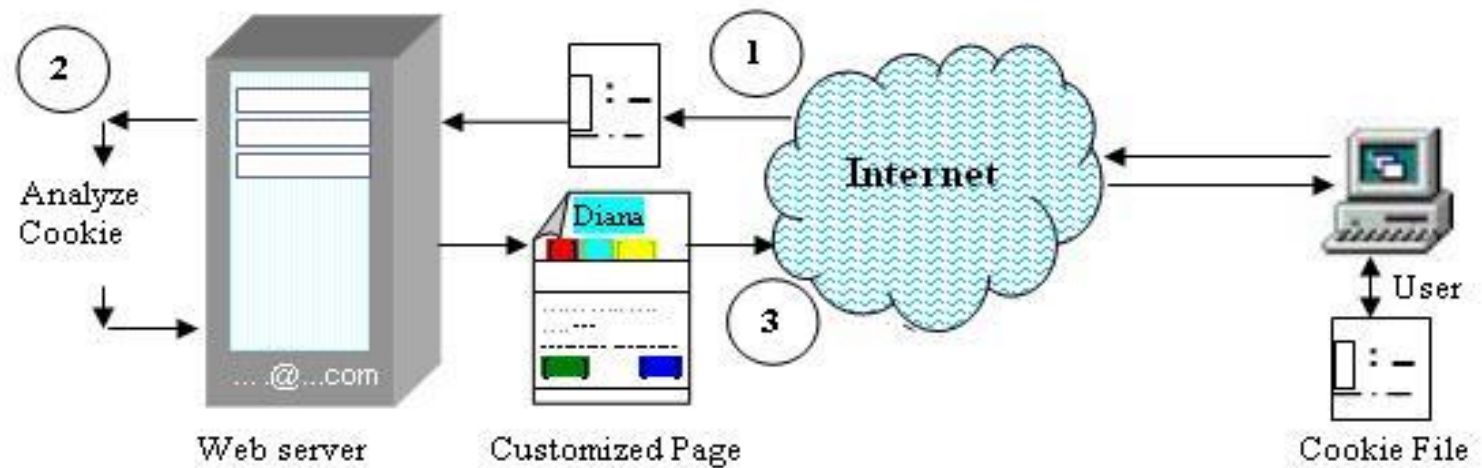


At the first contact of a new client with the web site the server sends to this one a HTML page defining the syntax for creating a cookie

When the script interpreted by the browser this one creates a cookie file containing some pre-established information (including the address and visit date).

Figure 5.39 The implementation of a cookie

## 5.6 Advanced Management Architecture



**1** When a client connects to a previously visited site this sends a cookie together with the request to process this to the web server

**2** The web server analyzes the cookie and takes an action

**3** If the cookie is accepted then a customized page is sent back to the user

Figure 5.40 The evolution of a cookie



## 5.7 Multi-tier Architecture

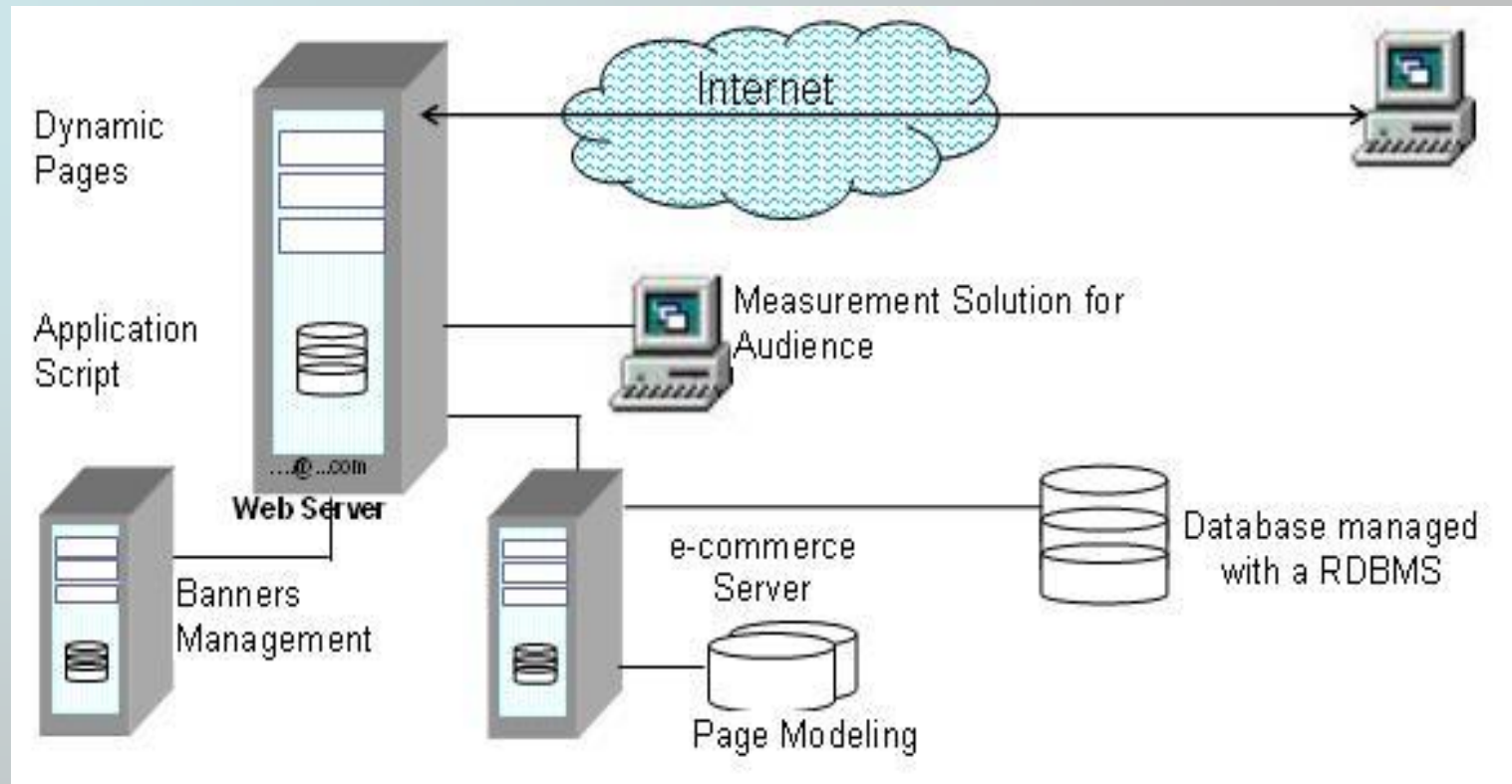


Figure 5.41 Network traffic analysis tools



## 5.7 Multi-tier Architecture

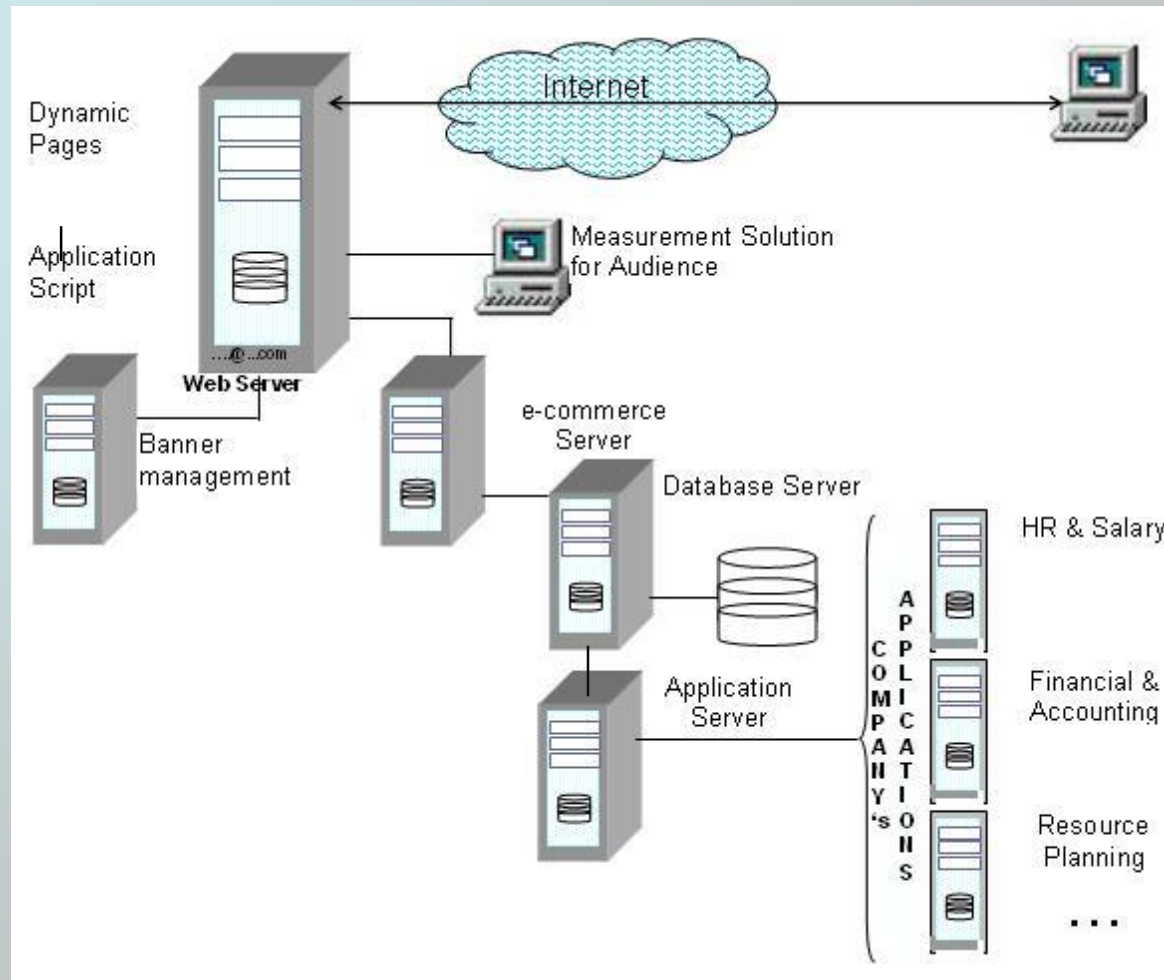


Figure 5.42 Three-tier Architecture

# References

1. [AVAI-09] Vasile Avram, Dragos-Marcel Vespan, Diana Avram, Alina Ion Internet Technologies for Business, Editura ASE, 2009
2. [AvDg03] Vasile Avram, Gheorghe Dodescu Informatics: Computer Hardware and Programming in Visual Basic, Ed. Economică, București, 2003 (Chp. 1.6, 1.7, 1.8, 7.11.3 and 7.11.4)
3. [AvAd-11] Vasile Avram, Diana Avram *An Architectural Solution of Assistance e-Services for Diabetes Diet*, Informatica Economica, Volume 15, Issue 2, January 2011, ISSN 1453-1305, EISSN 1842-8088
4. [BIS-TDM] Dave Chaffey, Paul Bocij, Andrew Greasley, Simon Hickie Business Information Systems-Technology, Development and Management for the e-business, Prentice Hall, London, second edition, 2003
5. [BF01] Benjamin Faraggi Architectures marcondes et portails B to B, Ed. DUNOD, Paris, 2001
6. [DgAv05] Gheorghe Dodescu, Vasile Avram Informatics: Operating Systems and Application Software, Ed. Economică, București, 2005 (Chp. 10.1, 10.2 and 10.3)
7. [DOS\_03] Daconta, Michael C., Leo J. Obrst, and Kevin T. Smith. The Semantic Web: A Guide to the Future of XML, Web Services, and Knowledge Management. John Wiley & Sons. © 2003. Books24x7. <[http://common.books24x7.com/book/id\\_6073](http://common.books24x7.com/book/id_6073)
8. [RFC 1630] T. Berners-Lee RFC 1630 - Universal Resource Identifiers in WWW, Network Working Group, CERN, June 1994
9. [RFC3986] T. Berners-Lee W3C/MIT, R. Fielding Day Software, L. Masinter Adobe Systems Uniform Resource Identifier (URI): Generic Syntax, January 2005
10. [HL-11] Chuck Hudson, Tom Leadbetter HTML5 Developer's Cookbook, 2011, Addison-Wesley, ISBN 978-0-321-76938-1
11. [KLJL] Kenneth C. Laudon, Jane P. Laudon Essentials of Management Information Systems – Managing the Digital Firm, Prentice Hall, fifth edition, 2003
12. [LAS-11] Peter Lubbers, Brian Albers, and Frank Salim Pro HTML5 Programming, Second Edition, Appress, 2011, ISBN-13 (pbk): 978-1-4302-3864-5, ISBN-13 (electronic): 978-1-4302-3865-2

# References

13. [OLS07] Phillip Olson et al PHP manual, <http://www.php.net/docs.php>, 2007
14. [W3C] www.w3c.org World Wide Web Consortium, Web standards collection
15. [MNSS] Todd Miller, Matthew L. Nelson, e-Business Management Models: A Services Perspective and Case Studies, Stella Ying Shen and Michael J. Revere Group Shaw
16. [SS05] Steve Schafer Web Standards Programmer's Reference: HTML, CSS, JavaScript, Perl, Python, and PHP Wrox Press © 2005
17. [RRSD] Robert Reinhardt, Snow Dowd Macromedia Flash 8 Bible, John Wiley & Sons © 2006
18. [JLMT] Jerri Ledford, Mary E. Tyler Google™ Analytics 2.0, John Wiley & Sons, August 27, 2007, ISBN-13: 978-0-47017501-9
19. E-commerce business models <http://www.iusmentis.com>  
<http://www.iusmentis.com/business/ecommerce/businessmodels/>
20. <http://digitalenterprise.org/models/models.html> Professor Michael Rappa, North Carolina State University
21. <http://reference.sitepoint.com/css/css> SitePoint CSS reference
22. [W3schools] <http://www.w3schools.com>
23. [W3C] <http://www.w3c.org>
24. [avrams.ro] Vasile Avram, <http://www.avrams.ro>
25. <http://www.google.com/analytics>
26. - David Powers, Creating your first website – Part 1: Set up your site and project files,  
[http://www.adobe.com/devnet/dreamweaver/articles/first\\_website\\_pt1.html](http://www.adobe.com/devnet/dreamweaver/articles/first_website_pt1.html), retrieved on 10 Nov 2013  
- David Powers, Creating your first website – Part 2: Creating the page structure,  
[http://www.adobe.com/devnet/dreamweaver/articles/first\\_website\\_pt2.html](http://www.adobe.com/devnet/dreamweaver/articles/first_website_pt2.html), retrieved on 10 Nov 2013