Internet Technology as a Tool for Solving Engineering Problems

Aleksander Malinowski
Department of Electrical and Computer Engineering
Bradley University, Peoria, Illinois 61625, USA
olekmali@ieee.org, http://cegt201.bradley.edu/~olekmali/

Bogdan Wilamowski
College of Engineering at Boise
University of Idaho, Boise, Idaho 83712, USA
wilam@ieee.org, http://nn.uidaho.edu/

Abstract - **This tutorial covers all primary technologies that can be used for Web programming with applications to the Internet based data acquisition and system control. The presentation is divided into two parts. The first part discusses several programming languages as programming tools. It provides in depth discussion of the tasks that are best to implement with and on the advantages versus challenges associated with particular cases. The following tools are covered: HTML, JavaScript, Java Applets, Cookies, CGI, PERL, PHP, native languages (C/C++), and Web server configuration for security. Special emphasis is given to PERL and JAVA. Several programming examples for client, server and client-server applications are provided. The fragments of code are selected so that they provide good jumpstart to programming in particular languages for anybody with good programming skills in any programming language (preferably C++ or C). Advantages and disadvantages of different computer languages are discussed so a proper programming platform for different applications and task can be chosen.**

I. SYSTEM ARCHITECTURE CONSIDERATIONS

During software development, it is important to justify which part of the software should run on the client machine and which part should run on the server. Sometimes even the very fundamental client-server architecture must be reconsidered in favor of a peer to peer decentralized structures. The decision about the architecture can be made either based on the process control strategy or based on the information storage.

In case of the process control approach, the first approach is used when there each of the controlled objects can considered to be separate from possible other similar objects. The latter architecture is more beneficial in case of many controlled objects that cooperate with each other. When the information storage is considered then client server is favored over peer to peer communication in cases where information must be centralized, or is easier to manage when it is centralized.

Even between these two models, there may be a hybrid. Consider an instance when one controls a process that is implemented by many objects that cooperate with each other. The controller either deals with each object separately using a client-server approach, or deals only with one object and then relies on the peer to peer architecture to carry out the request. The latter case adds additional complexity of dealing with a distributed server.

II. COMPONENT PARTITIONING AND DATA FLOW

Once a particular architecture is chosen, the component partitioning needs to be considered. Peer to peer architecture usually yields symmetry of all objects. The decisions are made for client-server based on several factors:

- Amount of memory and CPU power available for server and clients. These restrictions may be imposed by technological or cost restrains.
- Available bandwidth of the network connection.
- Connection reliability and latency, especially in case of closing the control loop via network.
- Ease of installation or no need to preinstall any specific component on a client machine.

It is possible to develop two dedicated software components, one for server, and another one for a client and preinstall both. However, other strategies allow for more flexibility such as an automatic installation or update of the client side-software from the server. The latter approach requires storage of the client software components on the server object, possibly increasing the memory requirements and the initial network traffic when a new client must to be installed or updated.

Regardless from the choice of just in time downloaded or preinstalled client the software designer must make choices regarding partitioning the tasks between the server and the client. In case of control, the best results are achieved when the control loop is closed locally on the server that is installed on the controlled. The Internet bandwidth is already adequate for many applications if their data flow is carefully designed. Furthermore, the bandwidth limitation will significantly improve with time. It is therefore important to develop methods, which take advantage of networks and then platform independent browsers. This would require solving several issues, such as:

- Minimization of the amount of data which must be sent by a network
- Task partitioning between the server and client
- Selection of programming tools used for various tasks
- Development of special user interfaces
- Use of multiple servers and job sharing among them
- Security, privacy and, in case of pay per use, account handling
- Portability of software used on servers and clients

- Distributing and installing network packages on several servers

Fig. 1 illustrates an example of software component partitioning for a semiautonomous remote controlled robot. This particular application utilizes several client-server partitions for multiple components. In addition, the network server is at the same time the client in the relation to the thin embedded server that controls the robot movements on the lowest software level.

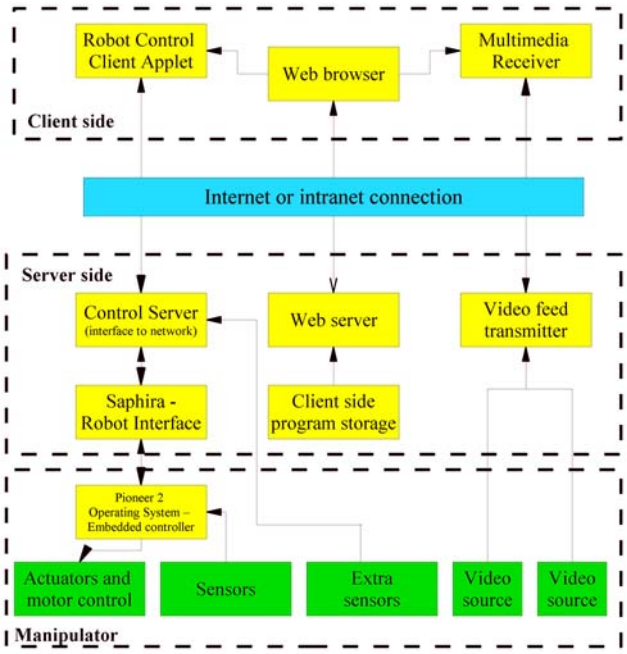


Figure 1. Example of client –server component partition.

Choosing the right set of software tools to implement the components of the system is the next dilemma to be solved after the decisions about the data flow among the software components that are distributed in the network are made. This problem is addressed in the next section.

III. MOST COMMONLY USED NETWORK PROGRAMMING TOOLS

Although it is possible to develop network applications using solely C++, or other compiled languages, it is much easier to develop networked applications using dedicated software tool for each component. There are several well-developed network-programming tools available today [1]. These tools include HTML, JavaScript, VBScript, Java, ActiveX, Common Gateway Interface (CGI) and PERL or C++, Active Server pages (ASP) and PHP. It is essential to make a correct decision which programming language should be used for which part of the software package. Short characterizations of different network programming tools are given below.

A. Hypertext Markup Language

Hypertext Markup Language (HTML) was originally designed to describe a document layout regardless of the displaying device, its size, and other properties [2]. It can be incorporated into networked application front-end development either to create form-based dialog boxes or as a tool for defining the layout of an interface, or wraparound for Java applets or ActiveX components. In a way, HTML can be classified as a programming language because the document is displayed as a result of the execution of its code. In addition scripting language can be used to define simple interactions between a user and HTML components [3][4]. Several improvements to the standard language are available: Cascading style sheets (CSS) allow very precise description of the graphical view of the user interface; Compressed HTML allows bandwidth conservation but can only be used by Microsoft Internet Explorer. HTML is also used directly as it was originally intended – as a publishing tool for instruction and help files that are boundled with the software.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2//EN">
<HTML>
<HEAD>
<TITLE>This is the title for this Web
Page</TITLE>
<META HTTP-EQUIV="Content-Type"
CONTENT="text/html; charset=iso-8859-1">
<META HTTP-EQUIV="Pragma" CONTENT="no-cache">
<META NAME="ROBOTS" CONTENT="INDEX,NOFOLLOW">
<META NAME="Author" CONTENT="BMW & AM">
<META NAME="Description" CONTENT="This is
displayed by the search engine">
<META NAME="KeyWords"
CONTENT="search engine kewoards, html">
</HEAD>
<BODY BGCOLOR="white" TEXT="black">
<H1>This is the title</H1>
<P ALIGN=”left”>This is the body of
this Web page.
<A HREF=”http://nn.uidaho.edu”>go</A>
<P ALIGN=”center”>Another paragraph and
<FONT COLOR="red" FACE="Ariel,helvetica"
SIZE=”+1”>a different font</FONT>
<!-- this is a coment -->
</BODY>
</HTML>
```

Figure 2. Typical HTML source code.

The HTML code shown in Fig.2 illustrates the nature of this language. The control structures are called tags. A tag is identified by `<` and `>` and us used to control the meaning and format that is used to display the information. Most of the tags are used in pairs, for example `<BODY>` and `</BODY>` marks the beginning and the end of the section that should be displayed as a Web page. Each tag may have several attributes. For example the two of many

attributes of `<BODY …>` are color setting `BGCOLOR` and `TEXT`. Inside the body of the page tags are used to provide text formatting. <P…> denotes a new paragraph, and is one of only a few tags that do not need the complementing and end tag `</P>`. `<Hn>` indicates the n-th header or section title of the n-th level. In addition to those and many other logical information tags, there are several tags that porvide only instruction regarding the way the text is to be displayed, for example `<FONT…>` tag. Although it is possible to set a particular font size in points, it is strongly recommended to alter the readers preference using relative sizes like +1 in the example above. The reader should be able to adjust the display to her preferences so that it is easy to read.

The anchor tag `<A …>` is the most important feature of the HTML. This implements the very idea of hypertext – the links. The `HREF` attribute instructs the Web browser about the location of another page that must be loaded in case the reader clicks on the text enclosed until `</A>`.

The header portion of the Web page that is marked by `<HEAD>` and `</HEAD>` may seam not to be that important. Information enclosed there may be very important for Web browsers, proxy systems or search engines. The example in Fig. 2 instructs the Web browser always to check for the new version of the Web page (`no-cache`), and defines the font set (`8859-1`) that is very important when the Web page displays any non-English characters. The other tags (robots, author, keywords and description) are sued by search engines to enhance the automatic classification of the Web page.

B. JavaScript

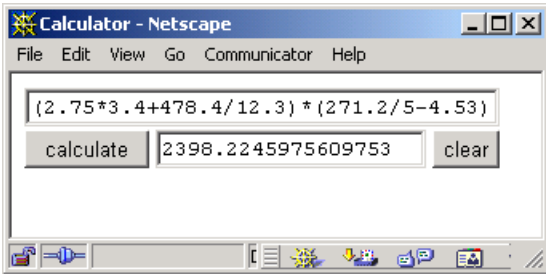
HTML itself lacks even basic programming constructs such as conditional statements or loops. A few scripting interpretive languages were developed to allow for use of programming in HTML [2]. They can be classified as extensions of HTML and are used to manipulate or dynamically create portions of HTML code. One of the most popular among them is JavaScript. The only drawback is that although JavaScript is already well developed still there is no one uniform standard. Different Web browsers may vary a little in the available functions [4]. JavaScript is an interpretative language and the scripts are run as the Web page is downloaded and displayed. There is no strong data typing or function prototyping. Yet the language includes support for object oriented programming with dynamically changing member functions. JavaScript programs can also communicate with Java applets that are embedded into an HTML page.


```
<SCRIPT language="JavaScript">
// this comment goes to the end of the line
alert("hello world!");
// end hiding comment
</SCRIPT>
<NOSCRIPT>No script support found</NOSCRIPT>
```

Figure 3. A simple example of JavaScript code.

JavaScript is part of the HTML code. It can be placed in both header and body of a Web page. The script starts with `<script language="JavaScript">` line. This example generates an alert dialog box shown above the code.

One of the most useful applications of JavaScript is verification of the filled form before it is submitted on-line. That allows for immediate feedback and preserves the Internet bandwidth as well as lowers the Web server load. Fig. 4 shows a sample code of an HTML form and its interaction with JavaScript that responds immediately.

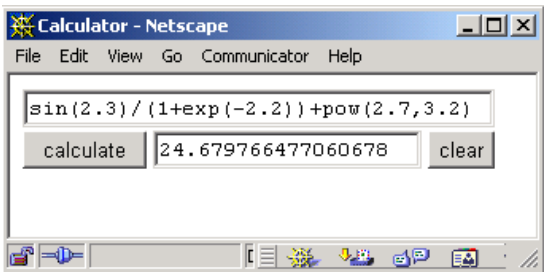



```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0//EN">
<html>
<head>
<TITLE>Calculator</TITLE>
</head>
<body>
<form name="form1">
<input type="text" name="text1" size="36"><BR>
<input type="button" name="button1"
    value="calculate"
    onclick=document.form1.text2.value
        =eval(document.form1.text1.value)>
<input type="text" name="text2">
<input type="reset" value="clear">
</form>
</body>
</html>
```

Figure 4. A Web page with JavaScript based calculator.

The next example shows a more powerful calculator, which is capable to compute even complicated functions.

Note that all computations are done not on the server but on the client computer. The web page generated is similar to this shown in the previous example but it is much more powerful. Its view and source code is shown in Fig. 5.




```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0//EN">
<html>
<head>
<TITLE>Calculator</TITLE>
</head>
<body>
<script language="JavaScript">
function math() {
var s=document.form1.text1.value;
s1=s.replace("abs", "Math.abs");
s2=s1.replace("sin", "Math.sin");
s3=s2.replace("cos", "Math.cos");
s4=s3.replace("sqrt", "Math.sqrt");
s5=s4.replace("tan", "Math.tan");
s6=s5.replace("atan", "Math.atan");
s7=s6.replace("asin", "Math.asin");
s8=s7.replace("acos", "Math.acos");
s9=s8.replace("exp", "Math.exp");
s10=s9.replace("floor", "Math.floor");
s11=s10.replace("log", "Math.log");
s12=s11.replace("max", "Math.max");
s13=s12.replace("min", "Math.min");
s14=s13.replace("pow", "Math.pow");
s15=s14.replace("random", "Math.random");
s16=s15.replace("round", "Math.round");
document.form1.text2.value=eval(s16);
}
</script>
<form name="form1">
<input type="text" name="text1" size=36>
<input type="button" name="button1"
value="calculate" onclick="math()">
<input type="text" name="text2"><BR>
<input type="reset" value="clear">
</form>
</body>
</html>
```

Figure 5. A Web page with advanced calculator.

C. Visual Basic Script

If the client-side software development is limited to Microsoft Windows and Microsoft Internet Explorer then VBScript may be used instead of JavaScript. The disadvantage is the lack of portability that is offered by this tool. However, that downside is compensated by ease of communicating with ActiveX components and possibility to use programs and libraries available to the operating system [2].

D. Java

Java is an object oriented programming language compiled in two stages. The first stage of compilation, to so-called byte-code, is performed during the code development. Byte-code can be compared to machine code instructions for a microprocessor [5]. Because no processor understands directly byte-code instructions, interpreters, called Java Virtual Machines (JVM) were developed for various microprocessors and operating systems. At some point JVM were improved so that instead of interpreting the code they do perform the second stage of compilation, directly to the machine language. However, to cut down the initial time to run the program the compilation is done only as necessary (just in time (JIT)) and there is no time for extensive code optimization [6]. At current state of art of JIT technology, programs written in Java run about two to five times slower than their C++ counterparts. Adding a JVM to a Web browser allowed embedding software components that could be run on different platforms [5][7].

Several features ensured success and increasing importance of this programming tool:

- similarity to C and C++ - a lot of existing programmers can switch relatively easily [5][7][8][9]
- support of C++ objects – suitability for large projects [5][8][9]
- simplified features – less complex than C++, easier to learn and utilize correctly [7][8][9]
- large standard set of libraries, including graphical libraries that can be used on multiple OS platforms [7][9]
- built in network libraries and some IP protocols [7][9][10]
- simple, platform independent multithreading – not as powerful as in C or C++ but much simpler [7][9]
- availability of fast JVM that use JIT compiler technology – only two times slower than C++ [6][7]
- ability to control the level of security buy enabling or disabling certain libraries that come with JVM
- availability of non-portable features by linking functions in machine language of a particular system [9]
- Smaller requirements for flash memory in the embedded systems due to compactness of byte-code (but more volatile memory is required)

Despite all those great advantages, there are a few problems of implementation that prevents Java from being used everywhere.

- it is still at least two to five times slower than C++ [6]
- multithreading does not have all features available to C or C++ programs [7][8]
- Most of implementations of JVM do not allow real time running due to garbage collector type of the memory management [6]
- Much higher memory requirements for JVM and running program (but smaller footprint of applications stored in flash memory versus C/C++)

D. ActiveX

Microsoft developed ActiveX as another technology allowing for the automatic transfer of software over the network [2][11]. ActiveX, however, can be executed presently only on a PC with a Windows operating system, thus making the application platform dependent. Although this technology is very popular already, it does not allow for the development of applications running on multiple platforms. ActiveX components can be developed in Microsoft Visual Basic or Microsoft Visual C++. There are the only choice in cases when Java is too slow, or when some access to the operating system functionality or devices supported only by Windows OS is necessary. The easy access to the operating system form an ActiveX component makes it impossible to provide additional security by limiting the features or resources available to the components.

Fig. 6 shows one of the simplest possible programs written in Java that also demonstrates use of functions. Since the language is strongly object oriented, all functionsmust be embedded in a class.

```
public class Test {
    public static void main(String args[]) {
        // a comment
        procedure("Hello programmer!");
    }
    private static void procedure(String s) {
        System.out.println(s);
    }
}
```

Figure 6. Code for a simple application written in Java.

Fig. 7 and Fig. 8 show a template for an applet written in Java. Applets are run embedded in Web pages. Fig. 7 shows how to embed the applet in HTML.

```
<APPLET CODEBASE="." CODE="Test.class"
    WIDTH="200" HEIGHT="100">
</APPLET>
```

Figure 7. Embedding an applet in a Web page.

Function `paint()` is called from the operating system environment whenever the graphics needs to be redrawn. Functions `init()` and `start()` are called when the applet is initialized. All computations should be initialized there and then carried on in a separate thread. Function `stop()` is called when the applet need to be stopped. All computations that were initialized in `start()` and carried on in another threads must be stopped then. This simple applet does not do anything besides painting a text and drawing two horizontal lines.

```
// a sample applet template
import java.applet.Applet;
import java.awt.Graphics;
public class Test extends Applet {
    public void init() {
    }
    public void start() {
    }
    public void paint(Graphics g) {
      g.drawLine(10,30,  120, 30);
      g.drawLine(10,60,  120, 60);
      g.drawString("Hello Programmer!", 10, 50);
    }
    public void stop() {
    }
}
```

Figure 8. A template for an applet written in Java.

E. CORBA and DCOM

CORBA (Common Object Request Broker Architecture) is a technology developed in the early 90's for network distributed applications. It is a protocol for handling distributed data, which has to be exchanged among multiple platforms [12][13]. A CORBA server or servers must be installed to access distributed data. CORBA in a way can be considered as a very high-level application programming interface (API). It allows sending data over the network, sharing local data that are registered with the CORBA server among multiple programs. Microsoft developed its own proprietary API that works only in Windows operating system. It is called DCOM and can be used only in ActiveX technology [11][14].

F. Common Gateway Interface

CGI, which stands for Common Gateway Interface, can be used for the dynamic creation of web pages. Such dynamically created pages are an excellent interface between a user and an application run on the server [2][9][15]. CGI program is executed when a form embedded in HTML is submitted or when a program is referred directly via a Web page link. The Web server that receives a request is capable of distinguishing whether it should return a Web page that is already provided on the hard

drive or run a program that creates one. Any such program can be called a CGI script. CGI describes a variety of programming tools and strategies. All data processing can be done by one program, or one or more other programs can be called from a CGI script. The name CGI script doe not denote that a scripting language must be used. However, developers in fact prefer scripting languages, and PERL is the most popular one.

Because of the nature of the protocol that allows for transfer of Web pages and execution of CGI scripts there is a unique challenge that must be faced by a software developer. Although users working with CGI-based programs have the same expectations as in case of local user interface, the interface must be designed internally in entirely different way. The Web transfer is a stateless process. That means, that no information is sent by Web browsers to the Web servers that identify each user. Each time the new user interface is sent as a Web page, it must contain all information about the current state of the program. That state is recreated each time a new CGI script is sent and increases the network traffic and time latency caused by limited bandwidth and time necessary to process data once again.

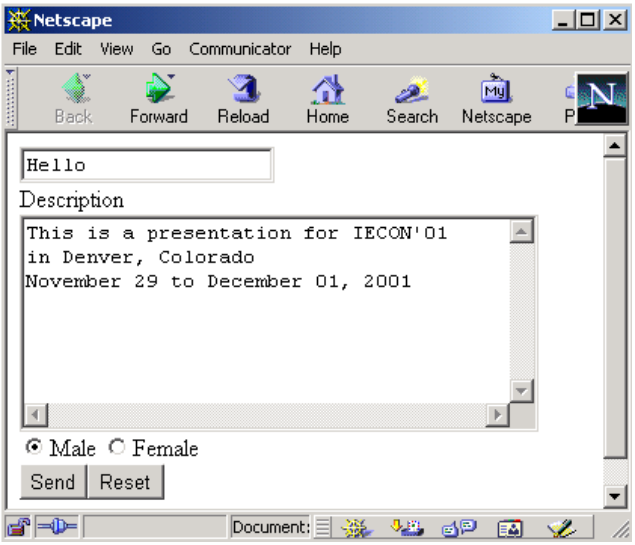
In addition, the server side software must be prepared for inconsistent data streams. For example, a user can back off through one or more Web pages give a different response to a particular dialog box and execute the same CGI script. At the time of the second execution of the same script, the data sent back with the request may already be out of synchronization from the data kept on server. Therefore, additional validation mechanisms must be implemented in the software that are not necessary in case of a single program.

G. PERL

PERL is an interpretive language dedicated for text processing. It is primarily used as a very advanced scripting language for batch programming and for text data processing [2][16][17]. PERL interpreters have been developed for most of existing computer platforms and operating systems. Modern PERL interpreters are in fact not interpreters but compilers that precompile the whole script before running it.

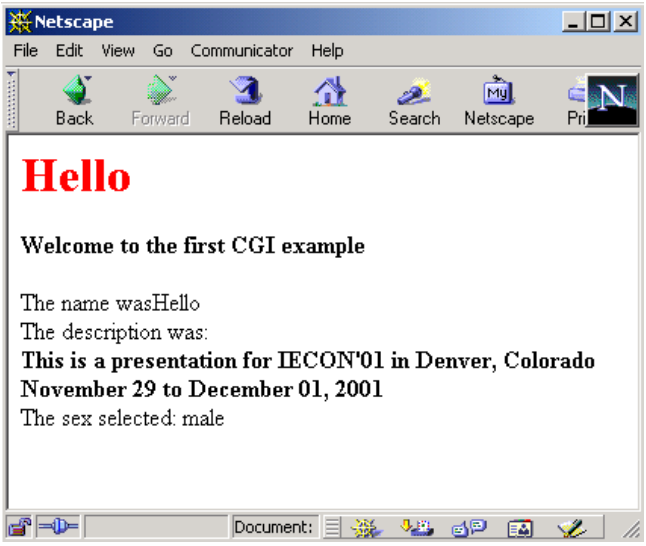
PERL was originally developed for Unix as a scripting language that would allow for automation of administrative tasks. It has many very efficient string, data stream and file processing functions. Those functions make it especially attractive for CGI processing that deals with reading data from the networked streams, executing external programs, organizing data, and in the end producing the feedback to the user in the form of a text based HTML document that is sent back as an update of the user interface [2][15]. Support of almost any possible computing platform and OS and existence of many program libraries makes it a platform independent tool.

Fig. 9 and Fig. 10 show an example of a data form that is filled in by a user on a remote computer (client). After the form shown in Fig. 9 is completed, the user clicks the "SEND" `submit` button. All data is transferred to the server and forwarded to the CGI script that is specified in the `form` tag in the `action` attribute. The source code of the CGI program is shown in Fig. 10. The program reads the data, processes them, and generates a Web page that is a feedback to the user.




```
<FORM action="http://nn.uidaho.edu/csp/cgi1.pl"
method="get">
<INPUT TYPE="text" name="name"><BR>
Description<BR>
<TEXTAREA name="description" rows=5 cols=40>
</TEXTAREA><BR>
<INPUT type="radio" name="sex" value="male">Male
<INPUT type="radio" name="sex"
value="female">Female
<BR>
<INPUT type="submit" value="Send"><INPUT
type="reset">
</FORM>
```

Figure 9. Data form implemented in HTML.




```
#!c:\progra~1\perl\bin\perl.exe
use strict;
use CGI qw(:standard);
print header;
print <<label1;
<H1><FONT COLOR="#FF0000"> Hello </FONT></H1>
<B>Welcome to the first CGI example </B><P>
label1
print "The name was", param('name'), "<BR>";
print "The description was:<B><BR>",
param('description'),"<BR></B>";
print "The sex selected: ",
param('sex'), "<P>";
```

Figure 10. A CGI-script written in PERL that handles data received from the form shown in Fig. 9.

The PERL code above uses CGI library with `param` function and this way reading data from the form is very simple. For example `param('name')` returns a string that was typed in the text field named `name` (see the HTML code above). `param('sex')` returns the name of the radio button pressed. `param('description')` returns a string that was typed in the text area named `description`. The PERL code generates a new screen on the client computer as shown above the code.

Please note that there are two ways of displaying messages of the client computer. The first

```
print <<label1;
<H1><FONT COLOR="#FF0000"> Hello </FONT></H1>
<B>Welcome to the first CGI example </B><P>
label1
```

sends entire HTML code between lines `print <<label1;` and `label1`. The other way is to use print statement and send HTML code line by line using `print` statements.

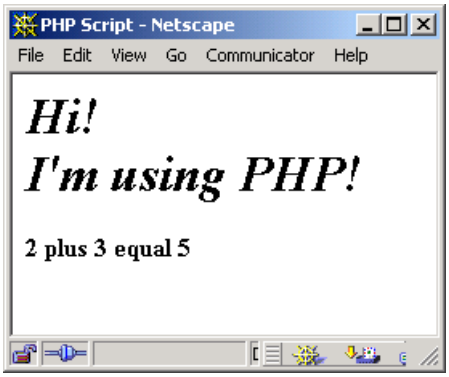
H. Active Server Pages

The concept of CGI scripts is centered on the idea that a program that is external to the Web server is run on the request made by a client. Then an HTML based reply is generated and sent back as the part of the outcome of the execution. Active Server Pages (ASP) provide the same functionality with the exception that the external program or programs are embedded into the skeletons of Web pages [18]. Those pages are preprocessed by the Web server before they are forwarded to the client, and the outcome of the embedded scripts is included.

In case of a CGI script, a reply to the user by sending an HTML based Web page is its significant portion. It makes sense then to provide also tools for embedding the scripts inside HTML instead of embedding HTML inside print statements in the CGI script. ASP technology is nothing else but shifting the way the server side programs are organized.

I. PHP

PHP is a scripting language like PERL. In fact, its syntax resembles PERL. The main difference lays in the set of standard built in libraries that support generation of HTML code, processing data from and to the Web server, and handling cookies. The same functionality can be accessed in PERL by inclusion of one or more libraries. PHP can be sued either as a classical CGI scripting language or as an implementation of ASP technology [18]. Since certain frequently used functionality is built in directly into the language, it is more efficient to use. In general any specialized tool will be somewhat more efficient for one particular task it was designed for, instead of other powerful but general purpose tools. PHP has been very popular for the last three years.

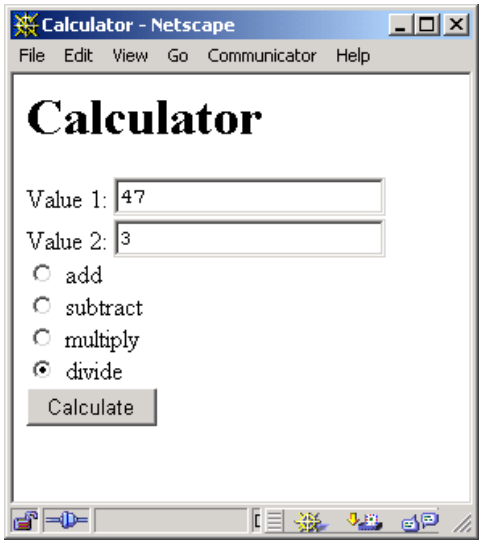



```
<HTML>
<HEAD><TITLE>PHP Script</TITLE></HEAD>
<BODY>
<?
echo "<H1><em>Hi! <BR>
I'm using PHP!</em></H1>";
$a = 2; $b = 3; $c=$a+$b;
echo "<B>$a plus $b equal $c </B></p>";
?>
</BODY>
</HTML>
```

Figure 11. A simple server side script in PHP.

PHP script (between `<?` and `?>`) can be easily incorporated into HTML code as illustrated in Fig. 11. Instead of `<?` and `?>` one can use `<?php` and `?>`, or `<script language="php">` and `</script>`. The script is run by the Web server on the server side, before the Web page is transferred through the Internet to the browser.

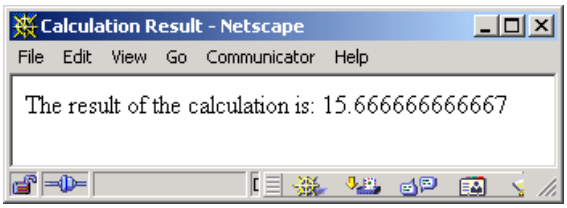
Fig. 12 shows another example of PHP programming. This time PHP is used to generate an HTML form. It is more concise than HTML and thus faster to develop and less likely to contain errors, but add to the load of the server computer. The resulting form is shown above the source code.




```
<HTML>
<HEAD><TITLE>Calculator</TITLE></HEAD>
<BODY>
<h1>Calculator</h1>
<FORM METHOD="post" ACTION="calculator.php">
Value 1: <INPUT TYPE="text" NAME="val1"></br>
Value 2: <INPUT TYPE="text" NAME="val2"></br>
<INPUT TYPE="radio" NAME="calc" VALUE="add">
add<br>
<INPUT TYPE="radio" NAME="calc" VALUE="sub">
subtract<br>
<INPUT TYPE="radio" NAME="calc" VALUE="mul">
multiply<br>
<INPUT TYPE="radio" NAME="calc" VALUE="div">
divide</br>
<INPUT TYPE="submit" NAME="submit"
VALUE="Calculate">
</FORM>
</BODY>
</HTML>
```

Figure 12. PHP utilized to generate a form in HTML.

When the form is submitted, the Web server needs to run a CGI script. Since a Web page merged with PHP can function as a program the PHP-based page can be used for the form processing as shown in Fig. 13. The Web page that is generated in the reply is shown above the source code.




```
<?
if (($val1 == "") || ($val2 == "") || ($calc
==="")) {
        header("Location:
http://nn2/cal_fm.htm");
        exit;
}
if ($calc == "add") {$r = $val1 + $val2;}
else if ($calc == "sub") {$r= $val1-$val2;}
else if ($calc == "mul") {$r = $val1*$val2;}
else if ($calc == "div") {$r = $val1/$val2;}
?>
<HTML><HEAD>
<TITLE>Calculation Result</TITLE> </HEAD>
<BODY>
The result of the calculation is: <? echo "$r";
?>
</BODY>
</HTML>
```

Figure 13. PHP utilized for Cgi scripting.

One of the principles of the correct coding is enclosing all source code that implements a particular functionality in one place. This can be applied to PHP. The code shown in Fig. 14 works both as HTML form generator and as the data processor in case it is called back by the generated form.

```
<HTML> <HEAD> <TITLE>AIO Form</TITLE> </HEAD>
<BODY>
<?
$formstring = "
<FORM METHOD=\"post\" ACTION=\"$PHP_SELF\">
Value 1: <INPUT TYPE=\"text\" NAME=\"val1\"></br>
Value 2: <INPUT TYPE=\"text\" NAME=\"val2\"></br>
<INPUT TYPE=\"radio\" NAME=\"calc\"
VALUE=\"add\"> add<br>
<INPUT TYPE=\"radio\" NAME=\"calc\"
VALUE=\"sub\"> subtract<br>
<INPUT TYPE=\"radio\" NAME=\"calc\"
VALUE=\"mul\"> multiply<br>
<INPUT TYPE=\"radio\" NAME=\"calc\"
VALUE=\"div\"> divide</br>
<INPUT TYPE=\"submit\" NAME=\"submit\"
VALUE=\"Calculate\">
</FORM>
";
if ($submit) {
if ($calc == "add") {$r = $val1 + $val2;}
else if ($calc == "sub") {$r= $val1-$val2;}
else if ($calc == "mul") {$r = $val1*$val2;}
```

```
else if ($calc == "div") {$r = $val1/$val2;}
echo "The result of the calculation is: $r";
} else {
echo "$formstring";
}
?>
```

Figure 14 Utilizing the sdame code both for HTML-form generation and data processing in CGI-script mode.

J. Cookies

A cookie is a piece of data stored in the client computer. When a request is sent to a server to get an HTML file, some cookies may be transmitted with that request. The server may send different data depending on the information retrieved from the user. Furthermore, JavaScript is also capable of browsing through all the cookies stored by the user machine [19]. This information may be used to enhance performance, for example by remembering the user's preferences. This very useful feature, however, is sometimes abused by some Internet providers, who can spy on the user by analyzing what kinds of web pages are being used.

IV. CONCLUSION

Given limited time and space that was allocated to this tutorial most of the important programming tools that can be applied to solving engineering problems were discussed. Client-server architecture and the system partitioning that were discussed in the introductory sections must be applied to a particular problem. Then based on need one or more tools has to be selected to implement client and server. HTML and JavaScript is generated on the server but utilized on the client side. CGI and ASP with PERL and PHP are stored and utilized on the server. Java can be used on the client side as well as on the server side. It allows implementing a complex functionality of a larger program by using object oriented and well-structured language.

If you are interested in more detailed examples or would like to participate in a 45 hour course offered by Bradley University as a long distance course please visit the Web site that is located at:

http://cegt201.bradley.edu/~olekmali/courses/

and follow the EE-WEB-2000 link to the course materials.

V. REFERENCES

For more information on particular topics discussed in this tutorial please refer to the following source materials that are recommended by the authors:

[1] Kaplan, G., "Ethernet's winning ways," *IEEE Spectrum,* January 2001, pp. 113-115.

[2] Jamsa K., Lalani S., Weakley S., *Web Programming,* Jamsa Press, Las Vegas, NV, 1996.

[3] Goodman, D., *Dynamic HTML, The Definitive Reference,* O'Reilly & Associates, Sebastopol, CA, 1997.

[4] Flanagan D., *JavaScript, The Definitive Guide,* O'Reilly & Associates, Sebastopol, CA, 1997.

[5] Van der Linden P., *Not Just Java,* Prentice Hall and Sun Microsystems, Palo Alto, CA, 1998.

[6] Web Page: Hank Shiffman, Boosting Java Performance: Native Code and JIT Compilers, `http://www.disordered.org/Java-JIT.html`, posted in 1998, last time visited in 2001.

[7] Van der Linden P., *Just Java 2,* Prentice Hall and Sun Microsystems, Palo Alto, CA, 1998.

[8] Web Page: Hank Shiffman, Making Sense of Java, `http://www.disordered.org/Java-QA.html`, posted in 1998, last time visited in 2001.

[9] Hall, M., Brown, L., *Core Web Programming 2nd ed.,* Prentice Hall, Upper Saddle River, NJ, 2001.

[10] Harold, E. R., *Java Network Programming,* O'Reilly, Sebastopol, CA, 1997.

[11] Roff, J.T., ADO: *ActiveX Data Objects,* O'Reilly & Associates, Sebastopol, CA, 2001.

[12] Object management Group, The Common Object Request Broker: Architecture and Specification, v. 2.2, published by Object Management Group, February 1998.

[13] Object management Group Web Site `http://www.corba.org/`, posted in 1997, visited in 2001.

[14] Thai, T.L., Oram, A., *Learning Dcom,* O'Reilly & Associates, Sebastopol, CA, 1999.

[15] Guelich, S., *CGI Programming with PERL, 2nd ed.,* O'Reilly & Associates, Sebastopol, CA, 2000.

[16] Wall L., Christiansen, T., Orwant, J., *Programming PERL, 3rd ed.,* O'Reilly & Associates, 1996.

[17] Holzner, S., *PERL Black Book,* Coriolis Group, 1999.

[18] Atkinson, L., Core PHP Programming: Using PHP to Build Dynamic Web Sites, 2nd ed., Prentice Hall, Upper Saddle River, NJ, 2000.

[19] CookieCentral.Com, Cookie Central, URL: `http://www.cookiecentral.com/`, posted in 1996, visited in 2001.