Integration of Configurable Dynamic Notification System with CSIBER Website

Dr.Poornima G. Naik
Professor
Department of Computer Studies
CSIBER
Kolhapur, India
pgnaik@siberindia.edu.in

Abstract— In this digital era every academic institution and commercial setup investments enormously in hosting and maintenance of the website which plays a critical role in the success of an organization by making it reachable across wide geographical area at any time. A carefully designed website reflects institute's best assets and delivers tremendous first-hand information to any user at any time irrespective of his/her geographical location. To stay in market there is a constant requirement for changing the look and feel and content of the website and incorporating dynamism into the website. It is inevitable to keep the website constantly updated since it is accessible to the public. As the new website data pertaining to event information, notification etc is constantly generated and old data soon becomes obsolete, it demands for continuous manual efforts from the human resource to keep the dynamically changing data current and up-to-date. It can save tremendous amount of human effort and time, if such a task is automated which in turn enables meaningful data to be displayed on the website with very little human intervention. To facilitate this new technologies such as jQuery, JSON, angular JS etc. are emerging continuously to name a few. In the current paper, the author has proposed an algorithm for the integration of dynamic notification system with existing website of CSIBER. The algorithm is implemented in PHP and MySQL and hosted on web server employing the web hosting service availed by the organization. The dynamic module is scheduled to be executed periodically on a daily basic by the Cron utility and server-side include is dynamically created and embedded in home page. Every month's events can be scheduled and stored in the backend database which is parsed by dynamic module and the required data is accordingly generated. As a measure towards efficiency improvement, the tool is executed once per day instead of executing it for every user request. Two options are proposed for integration, one on client-side and the second one on the server-side. The dialog displaying the notification data is rendered mobile friendly and is subject to Google's mobile friendly test.

Keywords- *Client-Side Include, Dynamic Website, Cron, jQuery, MySQL, PHP, Sever-Side Include.*

*****

I. Introduction

It is inevitable for any higher education institution is to have an effective, attractive and consistent web presence. Now-a-days most of the higher education institutions have hosted their websites which provide information to all of its stakeholders such as, current students and their parent, prospective students, faculty, alumni and campus recruiters. The website developers together with the management of the institute need to incorporate huge information in the website making it easy to explore. The effectiveness of the website depends on the website structure – arrangement and grouping of information and is of proper user navigation.There is a rapid technological advancement in design and deployment of web pages owing to the advent of jQuery, JSON, Angular JS and AJAX. jQuery is free, open source software standard JavaScript library which renders the designers easy control of HTML events, animations and other dynamic manipulation of style sheet handlers. jQuery enables Document Object Model (DOM) manipulation on the client-side in contrast to most of the server-side counterparts. With jQuery, web designers can quickly and easily write handlers to the document on specified events, such as a mouse click or a hover, and when we do so, it will change the style and/or position of that element. jQuery further simplifies AJAX--the client-side web development techniques for creating interactive web applications. jQuery outscores other similar technologies in that it is very small, clean code that won't bog down the loading time of web pages like Flash and other applications do. The contemporary technologies are failing to compete with the functionality of jQuery. Five lines of jQuery are equivalent to 25 lines of conventional JavaScript code, which relates to the least amount of "bloat" or extra, unnecessary code. This means smaller files and faster loading web pages. jQuery can perform all of the fancy animations and transitions as Flash, but in a cleaner, simpler, and more SEO friendly way. JSON is a stand-alone JavaScript Object which has replaced XML as object representation by choice owing to its simplicity. JSON notation is however not only readable by JavaScript, but most of the modern languages support JSON library for parsing JSON. Parsing of JSON is easier to implement and therefore less prone to errors than a XML implementation. Angular JS provides structure to JavaScript, two-way data binding, and is tailored for Single-Page Applications (SPA). Angular JS is based on MVC architecture which provides a more declarative (using HTML) way of connecting models to your view. The key to the new technology known as "AJAX" - was to communicate with the server in the background of the web-page, and update the existing web-page with new results.

A. Problem Definition

10% of the data on a typical website is constantly chaning pertaining to the current news, events and other notification messages. It is highly desirable to keep the contents of the public website up-to-date as large number of visitors across the globe will be constantly viweing website information. Replacement of a stale and oboslete data with the current one is a continuous process which demands constant updation and regular involvement of a human resource assigned particulaly for this task. In order to get rid of this manual task what is desirable is design of an automated tool which will interact with the backend database management system storing information about upcoming events along with the period for which the notification should appear dynamically on the home page of the website employing cutting edge technologies in web design.

To exactly address this issue and cater the earlier specified needs, the author has proposed an algorithm for the integration of dynamic notification system with existing website of CSIBER. The algorithm is implemented in PHP and MySQL and hosted on web server employing the web hosting service availed by the organization. The dynamic module is scheduled to be executed periodically on a daily basic by the cron utility and server-side include is dynamically created and embedded in home page. Every month's events can be scheduled and stored in the backend database which is parsed by dynamic module and the required data is accordingly generated. As a measure towards efficiency improvement, the tool is executed once per day instead of executing it for every user request.

II. LITERATURE REVIEW

There are very few papers in literature which deal with integration of notification system with the corporate's website.However, publish/subscribe infrastructures, specifically notification servers, are used in a large spectrum of distributed applications as their basic communication and integration infrastructure. Owing to their popularization, notification servers are being extensively developed to support specific application domains. Meanwhile, generalpurpose notification servers provide a large set of functionality for a broad set of applications. With so many options in place, developers face the dilemma of choosing between application-specific or general purpose notification servers. In both cases, however, the set of features provided by the servers are usually neither extensible nor configurable, making their customization to specific application domains a difficult task. In this regard, a more flexible approach is proposed – a customizable, extensible and dynamic architecture for notification services – which allows the customization of the notification service to different application domains is presented [1]. The extensibility model is presented according to the design framework proposed by Rosemblum and Wolf. The authors also have discussed a preliminary implementation of the prototype, as well as configuration examples. Over the last few decades, a large number of publish/subscribe services,

especially in the form of notification servers, have been used as the basic infrastructure for them implementation of many distributed applications such as user and software monitoring [2], groupware applications [3], awareness tools [4], workflow management systems and mobile applications [5]. Hence, due to its increased popularity, event notification services need to cope with new requirements, coming from different application domains. In fact, a broad spectrum of research and commercial tools are available nowadays. At one extreme, "one-size-fits-all" approaches, such as adopted by CORBA Notification Service (CORBA-NS) [6] or READY

Mobile push notifications are an important feature in mobile computing services which have been widely implemented in mobile systems. However, it also brings the vulnerability of security and reliability to the system. Formal specification and verification is an effective approach for understanding the properties of mobile push notification services and ensuring quality of the system development. Due to the dynamic interaction, security and mobility properties in mobile computing services, formally modeling and analyzing them is a grand challenge. In their paper, Ding et al [7] have proposed an approach to model mobile computing services using a high level Petri nets and analyze them through combining formal verification and testing techniques. The dynamic interaction between users and service providers is modeled with the publisher subscriber architecture. The mobility is modeled with a connector that dynamically connects a user to its service provider based on user's runtime environment, and the security is modeled with a threat model. The effectiveness of the approach is demonstrated with a case study of modeling and analyzing a mobile searching engine that is implemented with mobile push notifications services.

As wireless sensor networks gain in popularity, many deployments are posing new challenges due to their diverse topologies and resource constraints. Previous work has shown the advantage of adapting protocols based on current network conditions (e.g., link status, neighbor

status), in order to provide the best service in data transport. Protocols can similarly benefit from adaptation based on current application conditions. In particular, if proactively informed of the status of active queries in the network, protocols can adjust their behavior accordingly. Merlin et al [8] have proposed a novel approach to provide such proactive application event notification to all interested protocols in the stack. Specifically, we use the existing interfaces and event signaling structure provided by the X-Lisa (Cross-layer Information Sharing Architecture) protocol architecture, augmenting this architecture with a Middleware Interpreter for managing application queries and performing event notification. Using this approach, the authors observe gains in Quality of Service of up to 40% in packet delivery ratios and a 75% decrease in packet delivery delay for the tested scenario. The authors propose the notion to define an architecture that realizes interaction and integrated notification between mobile devices users and merchants [9]. This architecture allows user to specify a wide variety of interest through various gateways. It aggregates data from numerous sources and makes different type of interaction possible in single framework. The proposed architecture allows notification between different systems running on different platforms. It also accesses information in different modes from various mobile devices users. Moreover, it brings up the need of publishing and subscribing for information of interest. Furthermore, this system integrated an intelligent recommender system and designed to deliver the information of interest by user with imminent or immediately forthcoming events with applicable recommended notification. This model is suitable for environments that need strong personalization of services available in the mobile market place, where the convenience of data delivering request has fixed dead-line. This architecture not only helps in getting the items having high degree of interest, but also plays important role in revenue increase, cost reduction and enhancing the ability of electronic finance. More effectively framework has the potential to increase consumer satisfaction, enhance consumer/ company loyalty, and boost overall sales by targeting huge number of customers based on their interest.

A selective dissemination of information (SDI) system attempts to facilitate users' information retrieval and information ltering needs [10]. With the rise of the internet as an information source, the volume of information available ranging across all interests has exploded, anddifficulties in surveying, querying, and ltering information pertaining to individuals' interests increase with this explosion. The goal of an SDI system is to deliver new information arriving at an SDI-aware information provider to users who express their interests via user proles. Mechanisms used to implement such SDI systems vary; one such option is a persistent query mechanism. Users create and pose queries to an SDI system; queries remain resident in the system which works to somehow match documents and users. Successful matches are delivered from the SDI system back to users. In the thesis [11], the author addresses the dimensions of support necessary to implement a persistent query system; these are considered to be applicable at two locations within such a system at the information provider and in any system-wide infrastructure that may exist to support persistent querying. For our own protocol, we select a set of these dimensions, compose them, and present our reasons for such decisions. Specically, we are concerned with handling internet-based information providers, the scalability of the protocol, and not unduly burdening the information providers participating in persistent querying. The solution the authors propose uses variable rate transmission of change notications from information providers; authors believe these characteristics to be key to the development of a persistent query system. Their protocol is implemented in the existing Personalized Information Environments (PIE) system.

III. Application Architecture

The event and notification information is stored in a MySQL table on the web server. The cron utility on the web server is scheduled to execute the notification tool on a daily basis which retrieves the event information stored in the database, checks for the events of the current day and dynamically generates notification.html file on the web server which is embedded in the home page. The control flow logic is depicted in Figure 1.

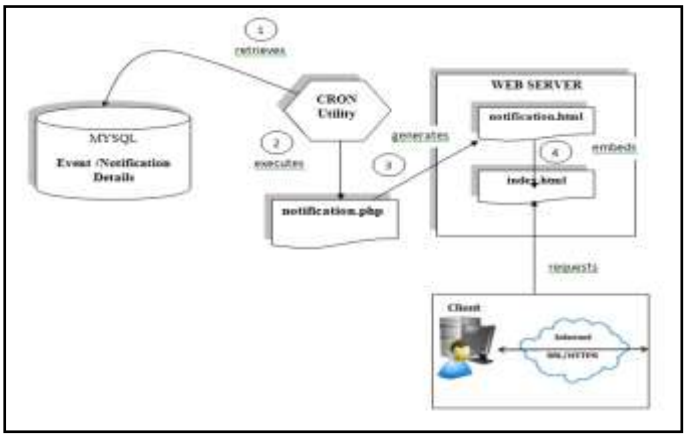


Figure 1. Interaction Between Different Components of Automated Notification System

The partial deployment folder structure on the web server for the implementation of automated notification system is depicted in Figure 2.

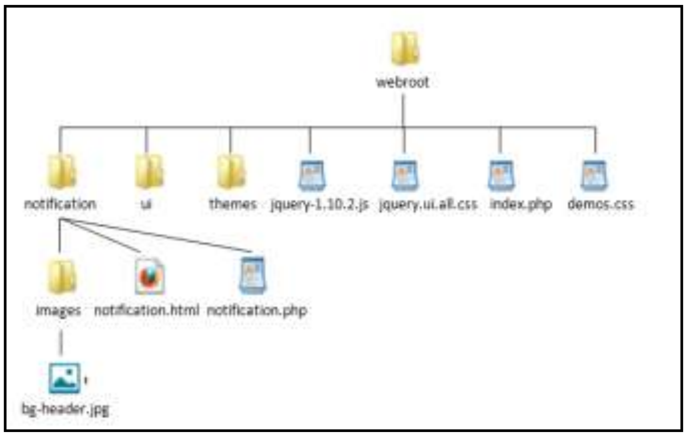


Figure 2. Partial Deployment Folder Structure for Automated Notification System

The author has designed and developed DFA parser for parsing the tokens [12]. The notification modal dialog is implemented in jQuery and request is processed using AJAX.

A. Proposed Algorithm

The algorithm developed for generation of notification tool in C++ style is presented below:

/*Any high level language interfacing with backend database management system provides high level API for primitive functions such as creating a database connection, executing the query, manipulating the data, creating a recordset object and fetching the data etc. Hence this algorithm assumes some standard functions as shown below:

Standard Functions of language L used in the Algorithm

getConnection() - is a built-in function in a language L returning Connection object and accepting servername, username and password as arguments.

getResultSet() - is function in a language L for returning a ResultSet object on execution of a query and using the Connection object specified as arguments to the function.

getColumn() - is function in a language L for returning number of the value of the column containing data in a table where row number and name are passed as arguments.

isEOF(ResultSet) - is a function in a language L which returns a boolean value, indicating whether the resultset pointer is at the beginning of or at the end of resultset. */

open() – is a function in a language L for opening a file given as a first parameter in a specified mode given as a second parameter to the function.

write() - – is a function in a language L for writing a string to the file specified as arguments to the function.

close() – is a function in a laguage L for closing a file whose name is passed as an argument to the function.

erase() - is a function in a language L for deleting the contents of a file whose name is passed as an argument to the function.

```
/* Global Variable Declaration */
char servername[50];
char user[20];
char pwd[20];
int flag;

function generateNotificationFile()
{
/* Read Current Date */
Read cDate;
/* Create a connection to MySQL database */
con=getConnection(servername, user,pwd);
query=”SELECT * FROM News”
resultSet=getResultSet(con,query);
flag=0;
/* Open the file in Write Mode */
open(“notification.html”,”w”);
while((isEOF(resultSet) == false)
 {
Read row;
startDate=getColumn(row, “Publishing_StartDate”);
endDate=getColumn(row, “Publishing_EndDate”);
hypertext=getColumn(row, “Text”);
hyperlink=getColumn(row, “URL”);
if (startDate <= cDate && endDate >= cDate)
{
        line=”<a href=’”+hyperlink+”’>”+hypertext+”</a>”;
        write(“notification.html”,line);
}
}
close(“notification.html”);
if (falg==0)
  erase(“notification.html”);
}
```

The algorithm is implemented in PHP and deployed on the web server. The PHP code along with the dynamically generated notification.html file is listed in Appendix A.

IV. RESULTS AND ANALYSIS

The algorithm proposed in Section III is implemented in PHP with MySQL as back end for storing notification specific information. CSIBER institute’s website is hosted on GoDaddy server on Linux platform. The utilities made available at the web hosting server is depicted in Figure 3.

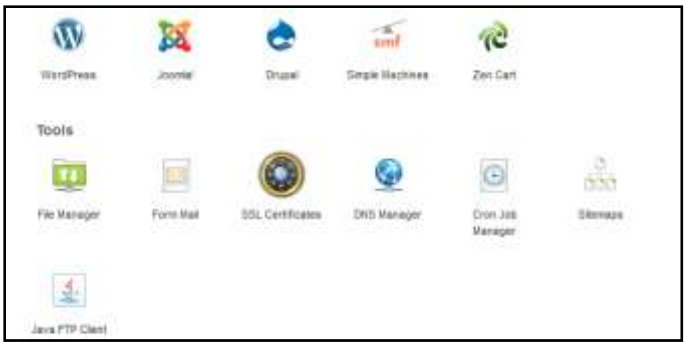
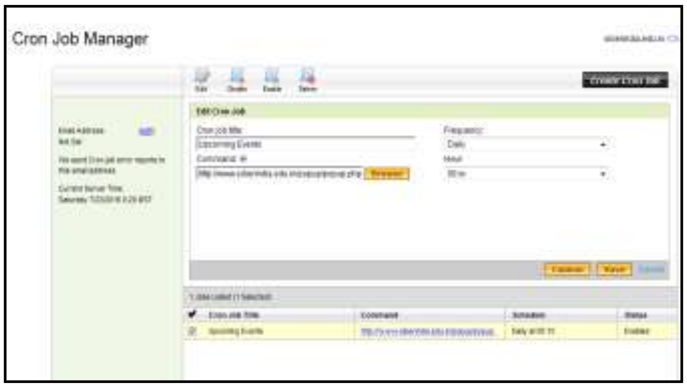


Figure 3. Utilities Provided by Web Hosting Service Provider

The Cron job shceduler is utilized for executing the notification PHP script on daily basic. The Cron configuration details are depiected in Figure 4(a) and 4(b).



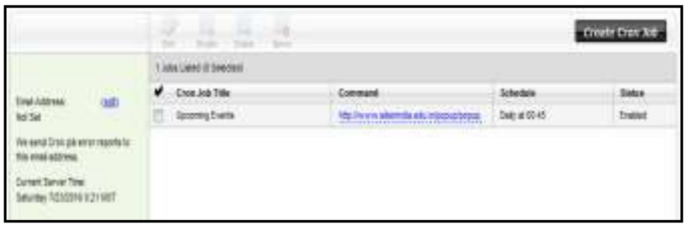


Figure 4(a)-4(b). Cron Job Scheduler for Scheduling Notification PHP Script

Structure of the EventNotification table is depicted in Figure 5.

Text	URL	Publishing_StartDate	Publishing_EndDate

Figure 5. Structure of Event_Notification Table

The description of different columns in the table in depicted in Table I.

TABLE I. DESCRIPTION OF COLUMNS IN EVENTNOTIFICATION TABLE.

Column Name	Description
Text	Employed in renders hypertext dynamically
URL	Employed in generation of href attribute
Publishing_StartDate	If the current date is between Publishing_StartDate and Publishing_EndDate the hyperlink is created and rendered on the notification dialog.
Publishing_EndDate	

The notification dialog is created in jQuery and displayed on the website employing server-side include as depicted in Figure 6.

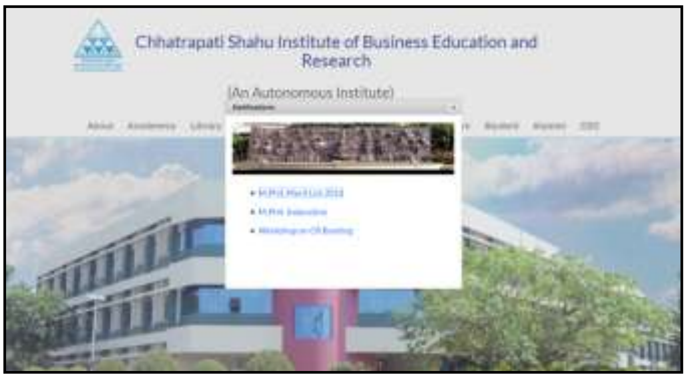

Chhatrapati Shahu Institute of Business Education and Research
(An Autonomous Institute)

Figure 6. Integration of Notification system with CSIBER Website

A. Testing the Application on Multiple Devices.

Responsive Web Design

Responsive Web design targets at the design and development responding to the user's behavior and environment based on screen size, platform and orientation. For rendering the notification dialog user friendly, the in-built screen object is queried for height and width of the target device and the layout is accordingly adjusted as depicted in the following code.

```
<script>
x=screen.width;
y=screen.height;
var w = x - 200;
var h = y - 200;
        $(function() {
                    $( "#dialog-modal" ).dialog({
                                height: h,
                                 width: w,
                                modal: true
                    });
        });
</script>
```

The better method is to use HTML5 and CSS3. Bootstrap makes use of certain HTML elements and CSS properties that require the use of the HTML5 doctype.

The website is then subject to Google’s mobile friendly test by visiting the URL https://www.google.com/webmasters/tools/mobile-friendly/?url=www.siberindia.edu.in.

The output generated is depicted in Figure 7.

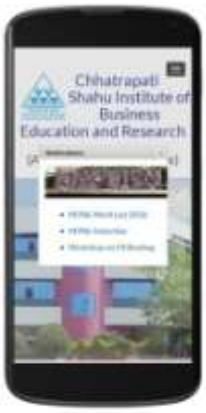

Chhatrapati Shahu Institute of Business Education and Research

Figure 7. Mobile Friendly Test

There are two options for integrating the notification system with the existing CSIBER website.

i. Using client-side include in HTML using JavaScript function.

ii. Using server-side include in PHP using include_once() function.

The partial PHP code for server-side include is listed below:

B. Server-Side Include

```
<script>
    x=screen.width;
    if ( x > 1000)
    {
       $(function() {
                  $( "#dialog-modal" ).dialog({
                              height: 400,
                               width: 500,
                              modal: true
                  });
       });
    }
    else
    {
        $(function() {
                  $( "#dialog-modal" ).dialog({
                              height: 380,
                               width: 400,
                              modal: true
                  });
       });

    }
  </script>
</head>
<body>
<?php
session_start();
if (!isset($_SESSION['popup']))
{
  $_SESSION['popup']=1;
  echo $_SESSION['popup'];
  include_once('popup/notification.html');
}
?>
```

C. Client-Side Include

For facilitating client-side include, the JavaScript function displayNotification()is dynamically generated and stored in a file with the extension .js, which is embedded in HTML file and the function is invoked at runtime.

```
    function displayNotification()
     {
    document.write("           <div           id="dialog-modal"
title="Notifications">
    document.write("<p><div class="popwrapper">
    .
    .
    .
    .
    .
    document.write("</div>
    document.write("</div>
    document.write("</div>
     }
<script src= 'notification.js'>displayNotification();</script>
```

V. Conclusion and Scope for Future Work

A typical website data is open to public and as such demands for constant updation so that only the current and meaningful data is made available to the public. To facilitate this, in the current work, the author has proposed an algorithm for the integration of dynamic notification system with existing website of CSIBER. The algorithm is implemented in PHP and MySQL and hosted on web server employing the web hosting service availed by the organization. The dynamic module is scheduled to be executed periodically on a daily basic by the Cron utility and server-side include is dynamically created and embedded in home page. Every month's events can be scheduled and stored in the backend database which is parsed by dynamic module and the required data is accordingly generated. As a measure towards efficiency improvement, the tool is executed once per day instead of executing it for every user request. Two options are proposed for integration, one on client-side and the second one on the server-side. The dialog displaying the notification data is rendered mobile friendly and is subject to Google's mobile friendly test.

References

[1] Roberto S. Silva Filho, Cleidson R. B. de Souza, David F. Redmiles , The Design of a Configurable, Extensible and Dynamic Notification Service,

[2] D. Hilbert and D. Redmiles, "An Approach to Large-scale Collection of Application Usage Data over the Internet," presented at 20th International Conference on Software Engineering (ICSE '98), Kyoto, Japan, 1998.

[3] P. Dourish and S. Bly, "Portholes: Supporting Distributed Awareness in a Collaborative Work Group," presented at ACM Conference on Human Factors in Computing Systems (CHI '92), Monterey, California, USA, 1992.

[4] Anita Sarma and A. v. d. Hoek, "Palantír: Increasing Awareness in Distributed Software Development," presented at International Workshop in Global Software Development at ICSE'2002, Orlando, Florida, 2002.

[5] G. Cugola, E. D. Nitto, and A. Fuggeta, "The Jedi Event-Based Infrastructure and Its Application on the Development of the OPSS WFMS," IEEE Transactions on Software Engineering, vol. 27, pp. 827-849, 2001.

[6] OMG, "Notification Service Specification v1.0.1," Object Management Group, 2002.

[7] R. E. Gruber, B. Krishnamurthy, and E. Panagos, "The Architecture of the READY Event Notification Service," presented at In Proceedings of the 1999 ICDCS Workshop on Electronic Commerce and Web-Based Applications, Austin, TX, USA, 1999.

[8] Junhua Ding, Wei Song and Dongmei Zhang, International Journal of Services Computing (ISSN 2330-4472) Vol. 2, No. 4, Oct.-Dec. 2014

[9] Supporting Proactive Application Event Notification to Improve Sensor Network Performance, Christophe J. Merlin and Wendi B. Heinzelman, ADHOCNETS 2009, LNICST 28, pp. 3–18, 2010.

[10] International Journal of Distributed and Parallel Systems (IJDPS) Vol.2, No.1, January 2011 DOI : 10.5121/ijdps.2011.2110 116, Taxonomy for personalized Notification System for Improving the Quality Service of Mobile Marketplace, Amer Ali Sallam and Siba K. Udgata

[11] Selective Dissemination of Information in the Dynamic Web Environment A Thesis Presented to the Faculty of the School of Engineering and Applied Science University of Virginia Edward K. O'Neil, May 2001

[12] Girish R. Naik, Dr. V.A. Raikar, Dr. Poornima G. Naik, Multi Objective Criteria for Selection of Manufacturing Method, International Journal of Advanced Research in Computer Science and Software Engineering, Volume 4, Issue 7, July 2014

Appendix A
Source Code of notification.php

```
<?php
$dt= date('Y-m-d');
$servername = "xxxx";
$usr = "root";
$pwd = "xxxx";
/* Creating Database Connection */
$conn = mysql_connect($servername, $usr, $pwd);
  if(! $conn )
  {
    die('Could not connect: ' . mysql_error());
  }
mysql_select_db( 'popup' );
$sql = "select * from News";
$retval = mysql_query( $sql, $conn );
if(! $retval )
{
  die('Could not get data: ' . mysql_error());
}
/* Create file storing notification information */
$fp = fopen('notification.html', 'w');
fwrite($fp,   '<div   id="dialog-modal"   title="Up   Coming
Events">');
fwrite($fp, '<p><div class="popwrapper">');
fwrite($fp, '<div class="popimage-outer">');
fwrite($fp, '<img src="webroot/images/bg_header.jpg"');
fwrite($fp,   'class="popimage"   alt="Image"   width=675
height=200>');
fwrite($fp, '</div>');
fwrite($fp,   '<div   class="popcontent"><h4>Up   Coming
Events</h4><br>');
fwrite($fp, '<ul>');
/* Generation of dynamic content starts */
$flag=0;
fwrite($fp, '<h4>');
while($row = mysql_fetch_array($retval, MYSQL_ASSOC))
```

```
{
   if( $dt>=$row['StartDate'] && $dt<=$row['EndDate'])
    {
          $flag=1;
          fwrite($fp, '<li><a href=');
          fwrite($fp, $row['URL']);
          fwrite($fp,'>');
          fwrite($fp, $row['Text']);
          fwrite($fp, '</a></li><br>');
    }
}
fwrite($fp, '</h4></ul>');
fwrite($fp, '<div class="td-center"><a
href="javascript:self.close();"><img ');
fwrite($fp, 'src="webroot/images/close-');
fwrite($fp, 'popup.png"class="pop-close" alt="Close"></a>');
fwrite($fp, '</div>');
fwrite($fp, '</div>');
fwrite($fp, '</div>');
fwrite($fp, '</div>');
/* Generation of dynamic content ends here */
fclose($fp);
mysql_close($conn);
/* flag value equal to 0 indicates no matching records found */
if ($flag == 0)
{
  $fp = fopen('notification.html', 'w');
  fclose($fp);
}
?>
```

Dynamically Generated menu.html file

```
<div id="dialog-modal" title="Notifications">
<p><div class="popwrapper">
  <div class="popimage-outer">
  <img
src="webroot/notification/images/bg_header.jpg"class="popim
age" alt="Image" width=675 height=200>
  </div>
  <div class="popcontent"><h4>Up Coming
Events</h4><br>
  <ul>
   <h4><li><a
href='webroot/admission/MPhil_MeritList.pdf'>M.Phil Merit
List 2016</a></li><br></h4>
   <h4><li><a
href='webroot/admission/MPhil_Induction.pdf'>M.Phil
Induction</a></li><br></h4>
   <h4><li><a
href='webroot/workshops/booting_workshop.pdf'>Workshop
on OS Booting</a></li><br></h4>
  </ul>
  <div class="td-center"><a href="javascript:self.close();">
  <img src="http://localhost/popup/picts-popup/close-
popup.png"class="pop-close" alt="Close"></a>
 </div>
</div>
</div>
</div>
```