Analyzing Performance of Web-based Metrics for Evaluating Reliability and Maintainability of Hypermedia Applications

Sanjeev Dhawan[1], **Rakesh Kumar**[2]
[1]Faculty of Computer Engineering, University Institute of Engineering & Technology (U.I.E.T), Kurukshetra University, Kurukshetra (K.U.K)- 136 119, Haryana, India.
[2]Faculty of Computer Science, Department of Computer Science and Applications (D.C.S.A), Kurukshetra University, Kurukshetra (K.U.K)- 136 119, Haryana, India.
E-mail: {rsdhawan, rsagwal}@rediffmail.com

Abstract - *This paper has been designed to identify the Web metrics for evaluating the reliability and maintainability of hypermedia applications. In the age of Information and Communication Technology (ICT), Web and the Internet, have brought significant changes in Information Technology (IT) and their related scenarios. Therefore in this paper an attempt has been made to trace out the Web-based measurements towards the creation of efficient Web centric applications. The dramatic increase in Web site development and their relative usage has led to the need of Web-based metrics. These metrics will accurately assess the efforts in the Web-based applications. Here we promote the simple, but elegant approaches to estimate the efforts needed for designing Web-based applications with the help of User Behavior Model Graph (UBMG), Web page replacement algorithms, and RS Web Application Effort Assessment (RSWAEA) method. Effort assessment of hyperdocuments is crucial for Web-based systems, where outages can result in loss of revenue and dissatisfied customers. Here we advocate a simple, but elegant approach for effort estimation for Web applications from an empirical point of view. The proposed methods and models have been designed after carrying out an empirical study with the students of an advanced university class and Web designers that used various client-server based Web technologies. Our first aim was to compare the relative importance of each Web-based metric and method. Second, we also implemented the quality of the designs obtained based by constructing the User Behavior Model Graphs (UBMGs) to capture the reliability of Web-based applications. Thirdly, we use Web page replacement algorithms for increasing the Web site usability index, maintainability, reliability, and ranking. The results obtained from the above Web-based metrics can help us to analytically identify the effort assessment and failure points in Web-based systems and makes the evaluation of reliability of these systems simple.*

Keywords: Web metrics, Web page replacement algorithms, Web usability, RS Web application effort assessment (RSWAEA), User behaviour model graph, method (UBMG).

1. Introduction

Reliable and precise effort estimation of high volume Web-based hyper-documents is critical for project selection, planning and control. Over the past thirty years, various estimation models have been developed to help the managers to perform estimation tasks, and this has led to a market offering a number of estimation tools. For organizations interested in using such estimation tools, it should be crucial to know about the predictive performance of the estimates such tools produce. The construction of an estimation model usually requires a set of completed projects from which an arithmetic model is derived and which is used subsequently as the basis for the estimation of future projects. So, there is a need for an estimation model to identify efforts of these Web projects. Web hypermedia applications have great potential in areas such as software engineering, education, and training to mention but a few. This paper looks at the relative importance of Web-based metric, their design, reliability, Web site usability and maintainability index, and ranking. The results obtained from the above techniques can easily evaluate the size and other important attributes related to Web-based hypermedia applications. The Web-based hypermedia applications are the non-conventional applications characterized by the authoring of information using nodes (chunks of information), links (relationship between nodes), access structures (for navigation), anchors, and its delivery over the Web. Web technologies commonly used for developing such applications are HTML, JavaScript, PHP and multimedia. The World Wide Web has created a standardized communications infrastructure that has enabled a wide range of applications, especially for business-to-business eCommerce, customer support, and entertainment. The rapid design and deployment of Web applications has been done largely in the absence of performance considerations. In addition, there have been great difficulties with forecasting site access patterns and dealing with the scalability issues. Thus, it is not surprising that Web-based applications frequently experience problems with poor reliability, long response times, and other important issues. This paper describes Web-based metrics for detecting and resolving such problems. Consequently, this paper has two main objectives: the first is to design a reliable and cost effective design using UBMG via Web server log files as prime attributes. The second objective is to compare the Web page replacement algorithms for defining the usability and maintainability index to estimate the effort to develop Web-based hypermedia applications, and finally, to choose the one that gives the best results, according to several measures of accuracy. Finally, our analyses suggest several areas (including reliability, usability index, maintainability, complexity, cost, time requirements and type of nature of Web design) where both Web-based designers, engineers and managers would be benefitted from better guidance about the proper implementation of Web-based applications [1] [2] [3].

2. Qualities of good software metric

Lord Kelvin once said that when you can measure what you are speaking about and express it in numbers, you know something about it. Measurement is fundamental to any engineering discipline. The terms "measure", "measurement", and "metrics" are often used interchangeably, but according to Pressman [4] a measure provides a quantitative indication of the extent, amount, dimensions, capacity, or size of some attribute of a product or process. Measurement is the act of determining a measure. The IEEE Standard Glossary of Software Engineering Terms [5] defines metrics as "a quantitative measure of the degree to which a system, component, or process possesses a given attribute". Ejiogu [6] suggested that a metric should possess the following characteristics: (a) Simple and computable: It should be easy to learn how to derive the metric and its computation should not be effort and time consuming, (b) Empirically and intuitively persuasive: The metric should satisfy the engineer's intuitive notion about the product under consideration. The metric should behave in certain ways, rising falling appropriately under various and conditions, (c) Consistent and Objective: The metric should always yield results that are unambiguous. The third party would be able to derive the same metric value using the same information, (d) Consistent in its use of units and dimensions: It uses only those measures that do not lead to bizarre combinations of units, (e) Programming language independent, (f) An effective mechanism for quality feedback. In addition to the above-mentioned characteristics, Roche [7] suggests that metric should be defined in an unambiguous manner. According to Basil [8] Metrics should be tailored to best accommodate specific products and processes.

3. Motivations of UBMG and RSWAEA

The techniques we propose have the following key objectives: (a) Derive the UBMG in a manner that capture complete details for valid sessions, and number of occurrences of invalid sessions, and (b) Derive the RSWAEA method to estimate the development effort of small to large-size projects, especially in scenarios that require fast estimation with little historical information. The valid sessions have metrics such as session count, reliability of session, probability of occurrence of the session, and transition probability of the pages in the session. On the basis of RSWAEA method, the Web-based software effort estimations are examined with user's cost, cost drivers, data Web objects, compatibility, usability, maintainability, complexity, configuration, time requirements, and number of interfaces.

3.1 Implementation and Analysis of User Behavior Model Graph (UBMG)

UBMG can be represented in form of a graph or a matrix notation [9]. In the graph view, nodes represent the pages, and arcs represent the transition from one node to another. In the matrix representation each cell (*i,j*) corresponds to probability of transition from page *i* to page *j*. We extend UBMG by adding an additional node to the graphical view, and a column in case of the matrix view to represent errors encountered while traversing. The construction of UBMG starts with the navigational model and access logs as described in [10], where the navigational model represents the complete overview of the different pages and the flow between the pages in the Web system. The access logs store information regarding the timestamp, page accessed client-id, referrer-id, HTTP return code etc. for determining session information. A sample format of IIS log file is shown in figure1.

```
#Fields: date time c-ip s-port cs-uri-stem cs-uri-query
sc-status time-taken cs (User-Agent) cs (Referrer)
```

Fig. 1. Format of IIS server log files

```
<Date and Time> <Client-id> <URL> <Referrer-id>
   2008-04-25 00:00:00 201.124.225.77 a.asp
   2008-04-25 00:00:02 201.124.225.77 b.asp
   2008-04-25 00:00:03 201.124.225.77 c.asp
   2008-04-25 00:00:05 201.124.225.77 e.asp f.asp
   2008-04-25 00:00:07 201.124.225.77 f.asp a.asp
   2008-04-25 00:00:06 201.124.225.77 d.asp f.asp
   2008-04-25 00:00:05 201.124.225.77 f.asp g.asp
   2008-04-25 00:00:10 201.124.225.77 d.asp g.asp
   2008-04-25 00:00:06 201.124.225.77 f.asp h.asp
   2008-04-25 00:00:06 201.124.225.77 a.asp h.asp
```

Fig. 2. Access log entries of IIS server

We consider referrer-id and the client-id fields as the basis to do a depth first search on the access logs. This approach will segregate valid and invalid sessions. To understand, consider an application with only two independent sessions- S1 with pages (a→ b→ c→ d → f→ g) and S2 with pages (a→ b→ c → e→ f→ h). Let the access log have entries as shown in figure 2. We derived two valid sessions when depth first search is based on client-id fields. However, with the referrer-id field we determine the invalid path consisting of pages (a→b→h). The count of all such invalid sessions is determined, and the construction of UBMG is done only for the valid sessions. Let us consider the example of an Online Airline Reservation System (OARS), where the two sessions defined in the navigational model are: *Session 1 (S1): "Book a seat" with pages SeatSelection.asp→ SeatDetails.asp→SeatBooking.asp→SeatConfirmed.asp→ SeatUnconfirmed.asp→ReservationControlNumber.asp→Payment.asp. Session 2 (S2): "Cancel a seat" with pages SeatSelection.asp→SeatDetails.asp→SeatBooking.asp→ SeatConfirmed.asp→SeatUnconfirmed.asp→ReservationControlNumber.asp→Refund.asp.* We tag an alias for the pages as given in figure 3, 4 and 5.

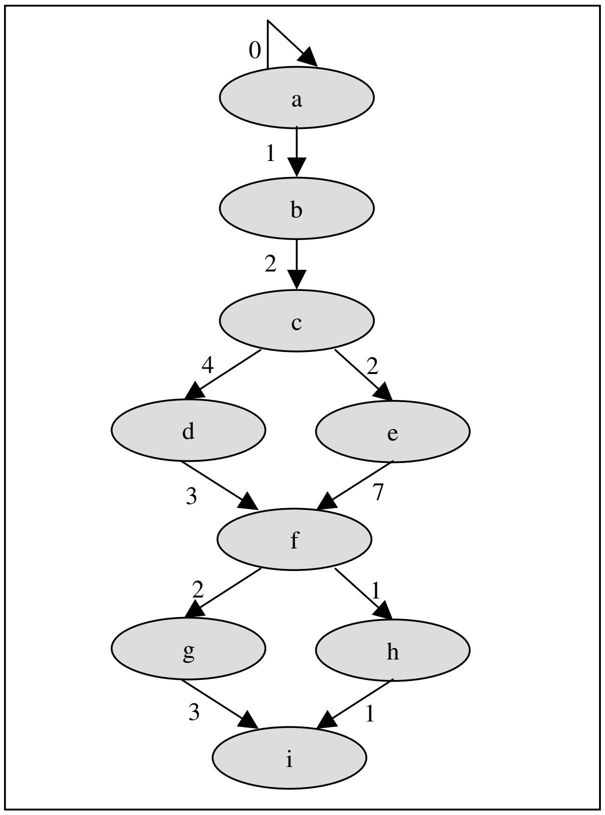


Fig. 3. Graphical view of UBMH

Where: a-SeatSelection.asp; b-SeatDetails.asp; c-SeatBooking.asp; d-SeatConfirmed.asp; e-SeatUnconfirmed.asp; f-ReservationControlNumber.asp; g-Payment.asp; h- Refund.asp. Figure 3 shows the graphical view of UBMG with the exit node 'i'. The matrixes of transition probabilities for graphs 3 and 4 have been shown in table 1 and 2 respectively. The matrix of table 1 considers only those sessions that have completed successfully, and the matrix of table 2 considers both successful sessions and sessions related to error nodes. For example, sum of probabilities of the paths out of the node b is 1 (table 1, 2 and 3) indicating that 10% of clients had either dropped out or encountered errors.

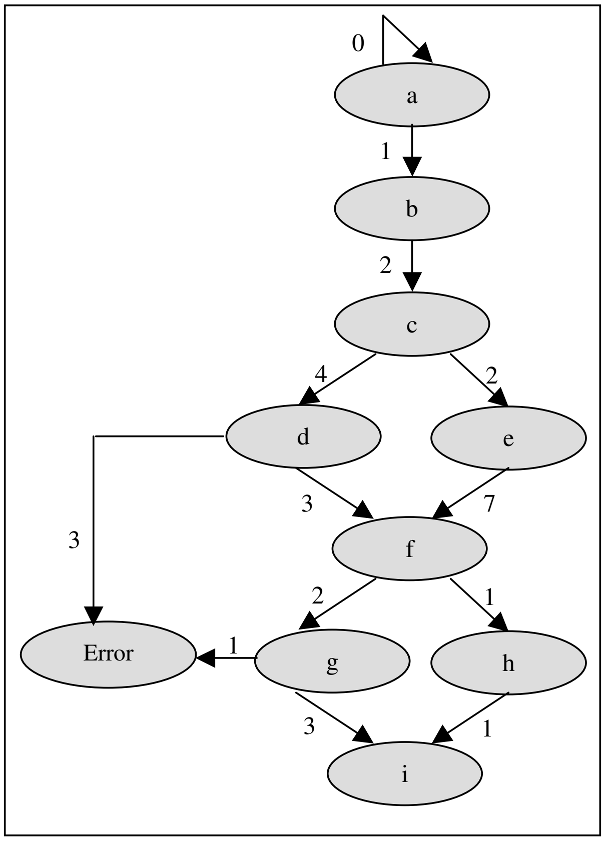


Fig. 4. Addition of an error node to UBMH

The probability of reaching a node *j* in the graph can be calculated using Markov property [10] [11] [12]. The generalized notation of using Markov property is:

$$Nj = N1 * P(1,j) + N2 * P(2,j) + \ldots + Nk * P(k,j) \qquad (1)$$

Where, k is the number of nodes that lead to node j. In the OARS example of table 1 (figure 3), to compute the probability of reaching the node 'f' is 3 * Nd + 7 * Ne and probability of reaching the node 'i' is 3 * Ng + 1 * Nh, where Na is equal to one. In the OARS example of table 2 (figure 4), to compute the probability of reaching the error node 'Error' is 3 * Nd + 1 * Ng. So, when an error node is included, then sum of all the incoming and outgoing weights of edges is increased. In the OARS example of table3 (figure 5), to compute the probability of reaching the error node 'Error' is 3 * Na + 3 * Nd + 1 * Ng. Finally, from the table 1, 2 and 3; it has been observed that whenever there is a existence of new node (either error or virtual node) the probability of reaching the new node is increased. Therefore, to resolve such type of issues, we have developed the Web replacement policies/algorithms (as discussed in section 4) to enhance the usability and reliability indexes of Web pages stored at different Web servers. The complexity of figures 3, 4, and 5 can be calculated using cyclomatic complexity (e-n+2 or e-n+1 or numbers of two edges in a single node+1, where: e is the total number of edges and n is the total number of nodes).

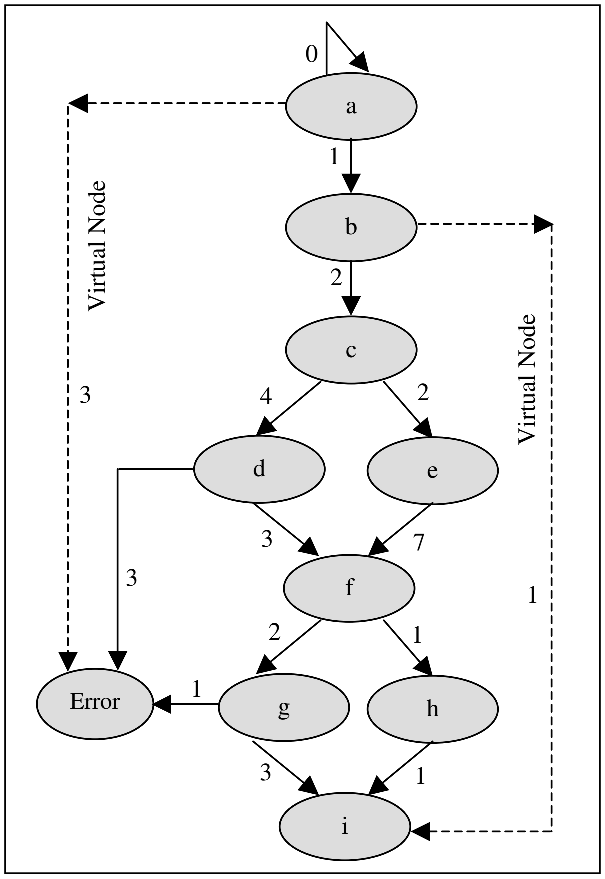


Fig. 5. Addition of an error and virtual nodes to UBMH.

	a	b	c	d	e	f	g	h	i	Sum
a	0	1	-	-	-	-	-	-	-	1
b	-	-	2	-	-	-	-	-	-	2
c	-	-	-	4	2	-	-	-	-	6
d	-	-	-	-	-	3	-	-	-	3
e	-	-	-	-	-	7	-	-	-	7
f	-	-	-	-	-	-	2	1	-	3
g	-	-	-	-	-	-	-	-	3	3
h	-	-	-	-	-	-	-	-	1	1
Sum	0	1	2	4	2	10	2	1	4	26

Table 1. Matrix of transition probabilities for OARS.

	a	b	c	d	e	f	g	h	i	Error	Sum
a	0	1	-	-	-	-	-	-	-	-	1
b	-	-	2	-	-	-	-	-	-	-	2
c	-	-	-	4	2	-	-	-	-	-	6
d	-	-	-	-	-	3	-	-	-	3	6
e	-	-	-	-	-	7	-	-	-	-	7
f	-	-	-	-	-	-	2	1	-	-	3
g	-	-	-	-	-	-	-	-	3	1	4
h	-	-	-	-	-	-	-	-	1	-	1
Sum	0	1	2	4	2	10	2	1	4	4	30

Table 2. Matrix of transition probabilities with error node for OARS.

	a	b	c	d	e	f	g	h	i	Error	Sum
a	0	1	-	-	-	-	-	-	2	3	6
b	-	-	2	-	-	-	-	-	-	-	2
c	-	-	-	4	2	-	-	-	-	-	6
d	-	-	-	-	-	3	-	-	-	3	6
e	-	-	-	-	-	7	-	-	-	-	7
f	-	-	-	-	-	-	2	1	-	-	3
g	-	-	-	-	-	-	-	-	3	1	4
h	-	-	-	-	-	-	-	-	1	-	1
Sum	0	1	2	4	2	10	2	1	6	7	35

Table 3. Matrix of transition probabilities with error and virtual nodes for OARS.

3.2 Failure analysis of UBMG

Now, we extend the UBMG to include the failure data. To capture the failure data, the access logs are scanned for HTTP return error codes of 4xx and 5xx as mentioned in [13]. Besides this, the errors from other servers are also considered. Theoretically, the error node can stem from any page in the graphical view. We add the error node 'Er' and all the page errors are associated with this node. The matrix of transition probabilities will have an additional column to represent the error node. A cell (m, Er) of this column will include the probability of transitioning from the node m to error node Er. Considering the OARS example, the view of UBMG with the addition of error node is shown in figure 4. The matrix of transition probabilities for the figure 4 is shown in Table 2. The matrix considers only those sessions that have some error. Of all the requests that enter node d, 35% of them encountered some error. Before proceeding to failure analysis due to service-level agreements (SLA) violation, we define the term Session Response Time (SRT) which is the sum of the service times of all the pages in the session. We define the SLA at session level and hence we need the desired response time target for each session. The access log files can be used to determine the page service time (PST) values. For example, in the IIS Web server the time-taken field represents the time spend by server to respond to the request. SRT is computed as the sum of PST's of its individual pages [16]. Further, we compute the number of successful sessions where the SLA was violated. Let S1 and S2 be two sessions for the OARS example. Table 4 shows the sessions information, where each session is represented by a unique column, and includes number of successful sessions, number of instances of SLA violation,

Sessions	S1	S2
1. Total no. of successful session	125	150
2. Total no. of SLA violation $N_{SLA\text{-}FAIL}$	64	67
3. Probability of failures due to (2)	0.59	0.56
4. Prob. of reaching exit node for each session	0.78	0.76
5. Probability of SLA violation for each session using (3) and (4)	0.37	0.32

Table 4. Results of SLA violation probability.

etc. The probability of reaching exit node for a session is computed as the ratio of number of exits with respect to the number of visits at the entry page. The matrix of transition probabilities for the figure 5 is shown in Table 4.

3.2.1 Reliability Calculation

To compute the reliability of software code-level failures, we resort to determine the probability of encountering the failure node $P_{CODE\text{-}ERROR}$ represented in figure 4. To solve this probability of reaching the error node, we formulate a set of equations from the matrix and use techniques like Cramer' s Rule, Matrix Inversion or Gauss Jordan Elimination method (for solving the sets of simultaneous equations). We also compute (a) the total number of failures due to invalid session $N_{INVALID\text{-}SESSION}$, and (b) number of instances where successful sessions did not meet SLA as $N_{SLA\text{-}FAIL}$. The probability of occurrence of invalid sessions is computed using (a). The probability of failure for a session due to (b) is computed by considering the total number of its successful sessions. In the OARS example, the probability of such failures is 0.59 in Session 1 and 0.56 in Session 2. The probability of a session reaching the exit node, but violating SLA or invalid sessions needs to be computed. The total session failure probability $P_{SESSION\text{-}FAILURE}$ is calculated as the sum of all the individual session probabilities and the probability of occurrence of invalid sessions. The overall probability of failure $P_{TOTAL\text{-}FAILURE}$ for the system is calculated as sum of the probability of reaching error node $P_{CODE\text{-}ERROR}$, and the probability of session failure $P_{SESSION\text{-}FAILURE}$ for the entire system. The overall reliability R_{SYSTEM} of the system is calculated by the equation (2):

$$R_{SYSTEM} = 1 - P_{TOTAL\text{-}FAILURE} \qquad (2)$$

Thus the reliability computation is driven by failures at software code level, failures due to SLA violation and invalid sessions.

4. Web page trace algorithms

Consider a Web page trace $W(n)=r(1)*r(2)*\text{----}*r(n)$ consisting of n Web page numbers (WPNs) requested in discrete time from 1 to n, where r(t) is the WPN requested at time t, and Web page trace is a sequence of Web page numbers. Therefore we define two references distances between the repeated occurrences of the same page in W(n). The forward distance '$f_t(x)$' for page 'x' is the number of time slots required from time 't' to the first repeated reference of Web page 'x' in the future:

$$f_t(x) = \begin{cases} k, \text{ if } k \text{ is the smallest integer such that } r(t+k)=r(t) \\ =x \text{ in } W(n) \\ \infty, \text{ if } x \text{ does not reappear in } W(n) \text{ beyond time } t \end{cases}$$

Similarly, we define a backward distance $b_t(x)$ as the number of time slots from time 't' to the most recent reference of Web page 'x' in the past:

$$f_t(x) = \begin{cases} k, \text{ if } k \text{ is the smallest integer such that } r(t-k)=r(t) \\ =x \text{ in } W(n) \\ \infty, \text{ if } x \text{ never appeared in } W(n) \text{ in the past} \end{cases}$$

Let R(t) be the resident set of all Web pages residing in Web server under the indexed Web page at time 't'.

4.1 Web page replacement policies

The following Web page replacement policies are specified in a Web server system for a Web Frame Repetition (WFR), which results in Web page faults at time 't' [14].

(a) Least recently used (LRU): This policy replaces the Web page in R(t) which has the longest backward distance or it will replace the Web page that has not been recently used for the longest period of time.
(b) Optimal (OPT) algorithm: This policy replaces the Web page in R(t) with the longest forward distance or replaces the Web page that will not be used for the longest period of time.
(c) First-in-first-out (FIFO): This policy replaces the Web page in R(t) which has been in memory for the longest time or when a Web page must be replaced, the oldest Web page is chosen.
(d) Least frequently used (LFU): This policy replaces the Web page in R(t) which has been least referenced in the past or it will replace the Web page that has the smallest count.
(e) Most frequently used (MFU): This policy replaces the Web page in R(t) which has been most referenced in the past or it will replace the Web page that has the largest count.

Consider a Web server system with two-level hierarchy: main memory M1 and disk memory M2. The number of Web page frame (WPF) is 3, labeled a, b, and c; and the number of pages in M2 is 11, 13, and 15 as presented in table 5, 6 and 7. For these experimental validations a sequence of random number has been generated. The sequence of Web page numbers so formed is the Web page trace. The following three Web page trace numbers have been taken to experiment with the Web page replacement policies: (i) 0 1 2 4 2 3 7 2 1 3 1; (ii) 0 1 2 4 2 3 7 3 2 1 1 3 1 (with an error node); and (ii) 0 1 2 3 2 4 2 3 7 3 2 1 1 3 1 (with error and virtual nodes). The results from figure 6 indicate the superiority of LRU and LFU policy over others. However, the OPT is very difficult to implement in practice. The LRU policy performs better than FIFO due to the locality of references. The LFU policy shows better results than MFU policy due to the least frequent usability index of Web pages. The MFU and LFU policies are useful to keep a counter of the number of Web page references that have been made to each page. From these, results, we realize that the LRU is generally better than FIFO, MFU and LFU. But, exceptions still exist due to the hyperlinks dependence on program behavior.

4.2 Usability index for Web pages

The usability index has been implemented by means of a set of suggestions (page links) dynamically generated on the basis of the active user session, which are used to personalize page requested. Typically, the Web usability is structured according to two components, performed off-line and on-line analysis with respect to the Web server activity. By analyzing the historical data (i.e. server access log files), the off-line component builds a knowledge base, which is used in the on-line phase to generate the personalized content. This content can be expressed in several forms, such as links to pages or advertisements considered of interest for the current user. User sessions are identified by means of cookies stored on the client side. Cookies contain the keys to identify the client sessions. For each Web page URL request has been generated.

The Web server knowledge base updated according to the characteristics of the current session, and then generated. Presuming that interest in a Web page depends on its content and not on the order in which a Web page is visited during a session, the edge weight is computed as $W=N_{ij}/max\{N_i, N_j\}$, where N_{ij} is the number of sessions containing both pages I and j, and Ni and Nj are the number of sessions containing only page i or j. Generally, LRU algorithm is applied. According to this algorithm, information about a Web page less recently accessed is replaced with that for a currently accessed Web page. Therefore, the system performance due to least recently used Web page (s) will enhance the Web site size and their performance.

	WPF	0	1	2	4	2	3	7	2	1	3	1	Hit Ratio
LRU	a	0	0	0	4	4	4	7	7	7	3	3	
	b		1	1	1	1	3	3	3	1	1	1	
	c			2	2	2	2	2	2	2	2	2	3/11
	WFR					*			*			*	
OPT	a	0	0	0	4	4	3	7	7	7	3	3	
	b		1	1	1	1	1	1	1	1	1	1	
	c			2	2	2	2	2	2	2	2	2	4/11
	WFR					*			*	*		*	
FIFO	a	0	0	0	4	4	4	4	2	2	2	2	
	b		1	1	1	1	3	3	3	1	1	1	
	c			2	2	2	2	7	7	7	3	3	2/11
	WFR					*						*	
LFU	a	0	0	0	4	4	4	7	7	7	3	3	
	b		1	1	1	1	3	3	3	1	1	1	
	c			2	2	2	2	2	2	2	2	2	3/11
	WFR					*			*			*	
MFU	a	0	0	0	4	4	4	4	2	1	1	1	
	b		1	1	1	1	1	7	7	7	7	7	
	c			2	2	2	3	3	3	3	3	3	3/11
	WFR					*					*	*	

Table 5. Results of Web page replacement policies for figure (3).

	WPF	0	1	2	3	2	4	2	3	7	3	2	1	1	3	1	Hit Ratio
LRU	a	0	0	0	3	3	3	3	3	3	3	3	3	3	3	3	
	b		1	1	1	1	4	4	4	7	7	7	1	1	1	1	
	c			2	2	2	2	2	2	2	2	2	2	2	2	2	8/15
	WFR					*		*	*		*	*		*	*	*	
OPT	a	0	0	0	3	3	3	3	3	3	3	3	3	1	1	1	
	b		1	1	1	1	4	2	2	7	3	3	3	3	3	3	
	c			2	2	2	2	2	2	2	2	2	1	1	1	1	5/15
	WFR					*			*			*			*	*	
FIFO	a	0	0	0	3	3	3	3	3	3	3	2	2	2	2	2	
	b		1	1	1	1	4	4	4	4	4	4	1	1	1	1	
	c			2	2	2	2	2	2	7	7	7	7	7	3	3	6/15
	WFR					*		*	*		*			*		*	
LFU	a	0	0	0	3	3	3	3	3	3	3	3	3	3	3	3	
	b		1	1	1	1	4	4	4	7	7	7	1	1	1	1	
	c			2	2	2	2	2	2	2	2	2	2	2	2	2	8/15
	WFR					*		*	*		*	*		*	*	*	
MFU	a	0	0	0	3	3	3	3	3	3	3	2	1	1	3	1	
	b		1	1	1	1	1	2	2	7	7	7	7	7	7	7	
	c			2	2	2	4	4	4	4	4	4	7	1	4	4	4/15
	WFR					*			*		*			*			

Table 6. Results of Web page replacement for figure (4) after addition of an error node.

	WPF	0	1	2	4	2	3	7	3	2	1	1	3	1	Hit Ratio
LRU	a	0	0	0	4	4	4	7	7	7	1	1	1	1	
	b		1	1	1	1	3	3	3	3	3	3	3	3	
	c			2	2	2	2	2	2	2	2	2	2	2	6/11
	WFR					*			*	*		*	*	*	
OPT	a	0	0	0	4	4	3	3	3	3	3	3	3	1	
	b		1	1	1	1	1	1	7	7	1	1	1	1	
	c			2	2	2	2	2	2	2	2	2	2	2	5/13
	WFR					*			*	*		*	*		
FIFO	a	0	0	0	4	4	4	4	4	2	2	2	2	2	
	b		1	1	1	1	3	3	3	3	1	1	1	1	
	c			2	2	2	2	7	7	7	7	7	3	3	4/13
	WFR					*			*			*		*	
LFU	a	0	0	0	4	4	4	7	7	7	1	1	1	1	
	b		1	1	1	1	3	3	3	3	3	3	3	3	
	c			2	2	2	2	2	2	2	2	2	2	2	6/13
	WFR					*			*	*		*	*	*	
MFU	a	0	0	0	4	4	4	4	4	4	4	4	4	4	
	b		1	1	1	1	1	7	7	7	7	7	7	7	
	c			2	2	2	3	3	3	2	1	1	3	1	3/13
	WFR					*			*			*			

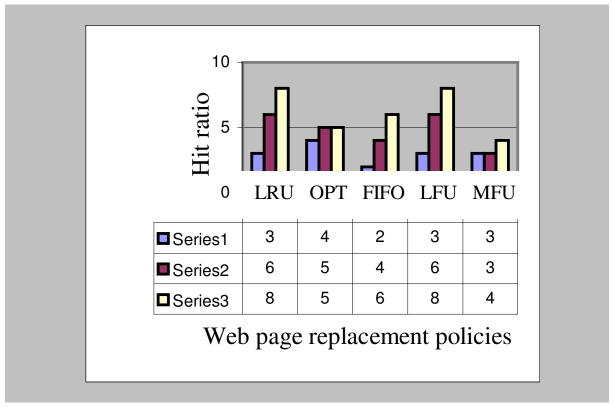
Table 7. Results of Web page replacement for figure (5) after addition of an error and virtual nodes.

4.3 Web page error estimating measure

We consider 'H' as hyperlink and 'I' as index page of a Web based application. The probability value P(HPA) refers to the probabilistic hyper-threading information between the presence or absence of H in an index page 'I'. The P(rel) indicates the probabilistic relation of 'H' for a given index page 'I'. Hence error-estimating function EE(I) is given by:

$$EE(I) = \log P(HPA/ rel) / P(HPA) \quad (3)$$

Where: P(HPA) is an approximate probability. If P(HPA) tends to be '1', then equation (3) becomes EE(I) = log P (HPA/rel). Since the hyperlinks frequently updates the concepts in case of dynamic or Active X pages. Therefore, the unique or monotonous results are not obtained during search. This leads a fact that the value of Web page under hyperlink Hi (*where i=1,2,3,----,n*) will change to another H_{i+1}, when reliability percentage of a Web page is concerned. The relation Hi and Hi can be compared at any instance based on the comparator rules such as (i) Hi = Hi; (ii) Hi ≈ H_{i+1}; (iii) Hi < H_{i+1}; and (iv) Hi > H_{i+1}. The rules indicate that the term is focused on utilization of hyperlinks.



▨ Series1	3	4	2	3	3
■ Series2	6	5	4	6	3
□ Series3	8	5	6	8	4

Fig. 6. Final characteristics of Web page replacement policies for table 5, 6 and 7.

5. The RSWAEA Method

In order to deal with the problem of effort estimation we have been studying the last two years the Web-based software development processes, related to the development of small and medium size Web-based information systems. Based on the analysis of these results, we identified a low usability of the well-known effort estimation methods and a necessity of a model to support estimation in such scenario. Due to this, we developed a method for fast estimating the Web-based software development effort and duration, which will definitely be adapted by the software community for the development of Web-based hyper media applications. We called it RS Web Application Effort Assessment (RSWAEA) method. The method will be very useful to estimate the development effort of small to large-size Web-based information systems. The DWOs (Data Web Objects) are an approximation of the whole size of the project; so, it is necessary to know what portion of the whole system DWOs represent. This knowledge is achieved through a relatively simple process (briefly described in next subsection). Assuming that the estimation factors in the computation of the effort are subjective, flexible and adjustable for each project, the role of the expert becomes very relevant. Once the value of the portion or representativeness is calculated, the expert can adjust the total number of DWOs and he/she can calculate the development effort using the following equation (4).

$$E= (DWO \,.\, (1+X^{*}))^{P} \,.\, CU \,.\, \prod_{i=1}^{8} cd_i \quad (4)$$

Where: E is the development effort measured in man-hours, CU is the cost of user, cd_i is the cost drivers, DWO corresponds to the Web application size in terms of data Web objects, X* is the coefficient of DWO representativeness, and P is a constant. The estimated value of real data Web objects (DWO) is calculated as the product of the initial DWOs and the representativeness coefficient *X**. This coefficient is a historical value that indicates the portion of the final product functionality that cannot be inferred from the system data model. The value of X* (coefficient of DWO representativeness) is between 1 to 1.3 depending upon small to large-size We-based applications. The process of defining such coefficient is presented in the next section. The cost of each user is has the values between 0 and 5. A value of CU of 0 means the system reuses all the functionality associated with each user type; so, the development effort will also be zero. On the other hand, if the cost of user is five, this means that there is no reuse of any kind to implement the system functionality for each user type. It represents the system functionality that is associated with each user type. The defined cost drivers (cd_i) are similar to those defined by Reifer for WebMo [15]. The last adjustable coefficient in RSWAEA corresponds to constant P that is the exponent value of the DWO. This exponent is a value very close to 1.01, and it must neither be higher than 1.12 nor lower than 0.99. This constant's value depends on the project size measured in DWOs. In order to determine this value, various statistical analyses have been done on various Web-based applications. As a result, this constant was assigned the value 1.09 for projects smaller than 300 DWOs, and 1.03 for projects larger than 300 DWOs [16].

6. Conclusions and future work

In this paper we have introduced an approach for determining the reliability, usability index, error estimating function, Web replacement policies, UBMH, RSWAEA and effort estimation for Web-based systems. These method work by offline and online analysis of Web logs and come up with useful metrics like usability index, error estimating function, Web replacement policies, cost, RSWAEA, UBMG, session count, SRT computation etc., and these metrics can effectively be used for the computation of reliable efforts for small to larger-size Web-based applications. Although these methods do not replace the expert estimator, but they provide him/her with a tool for achieving a more accurate estimation, based on real data in a shorter time. Estimating the cost, usability, error, duration and reliability of Web developments has a number of challenges related to it. To handle these challenges, we have analyzed many findings drawn from the experienced and expert opinions. Finally, by taking the good qualities of a software metric and an accessible Web design, we validated that the proposed models have better effort predictive accuracy than any existing traditional methods. In near future the development of Web-based applications using an object-oriented frame work, component-based framework, parametric-based framework or project-level framework and hardware-software co-designs related framework for sensitivity analysis and risk identification need to be designed. Our future work may include the study of lexical analysis together with COTS to develop the complete framework for effort assessment for authoring Web-based applications. However, positive results would suggest that the various efforts applied to estimate Web-based applications, would be an invincible task for the upcoming future.

7. Acknowledgments

A major part of the research reported in this was carried out at U.I.E.T and D.C.S.A, K.U.K, Haryana, India. We are highly indebted and credited by gracious help from the ERNET section of K.U.K for their constant support and help while testing our proposed models in different environment. The authors would like to thank those nameless individuals who worked hard to supply the data.

8. References

[1] Sanjeev Dhawan, Rakesh Kumar, Web Metrics for Evaluating Effort and Design of Hyperdocuments. *Association for Computing Machinery New Zealand Bulletin (ACM)*, 3 (1), 2007. 14-26. (ISSN 1176-9998).

[2] Sanjeev Dhawan, Rakesh Kumar, Measuring Quality Attributes of Web-based Applications, Part-I: Assessment and Design, *International Journal of Academic Open Internet Journal (AOIJ-2008)*, Bourgas, Bulgaria (In Press).

[3] Sanjeev Dhawan, Rakesh Kumar, Measuring Quality Attributes of Web-based Applications, Part-II: Analysis and Models, *International Journal of Academic Open Internet Journal (AOIJ-2008)*, Bourgas, Bulgaria (In Press).

[4] Pressman S. Roger, *Software Engineering- A Practitioner's Approach* (McGraw-Hill, 1997).

[5] *IEEE Trans. Software Engineering,* Vol. SE-10, pp-728-738, (1984).

[6] Ejiogu, L., *Software Engineering with Formal Metrics* (QED Publishing, 1991).

[7] Roche, J.M., Software Metrics & Measurement Principles, *Software Engineering Notes, ACM*, Vol. 19, no. 1, pp.76-85, 1994.

[8] Basili, V.R., & D.M.Weiss, A Methodology For Collecting Valid Software Engineering Data, *IEEE Software Engineering Standards*, Std. 610.12-1990, pp.47-48, 1993.

[9] Daniel A. Menasce, Virgilio A.F. Almeida, Scaling for E-Business Technologies, Models, Performance, and Capacity Planning (*Prentice Hall PTR, pp. 49-59, 2000*).

[10] Shubhashis Sengupta, Characterizing Web Workloads– a Transaction-Oriented View, *IEEE/ IFIP 5th International Workshop on Distributed Computing (IWDC- 2003).*

[11] Wen-Li Wang, Mei-Huei Tang, User-Oriented Reliability Modeling for a Web System, *14th International Symposium on Software Reliability Engineering (ISSRE)*, November 17 - 21, 2003.

[12] D.A. Menace, V.A.F. Almeida, R. Fonseca, M.A. Mendes, A Methodology for Workload Characterization of E-Commerce Sites*, Proceedings of 1st ACM conference on Electronic Commerce*, 99.

[13] The Internet Society, Request for Comments (RFC): 2616. Hypertext Transfer Protocol–HTTP/1.1, http://www.w3.org/Protocols/rfc2616/rfc2616.html.

[14] Kai Hwang, *Advanced Computer Architecture Parallelism Scalability Programmability*, TMH-New Delhi, 2004.

[15] D.J. Reifer, Web Development: Estimating Quick–to-Market Software, *IEEE Software*, Vol. 17, No. 6,Pages 57 - 64, 2000.

[16] Sanjeev Dhawan, Rakesh Kumar, Effort Assessment and Predictions for High Volume Web-based Applications: a Pragmatic Approach, *International ERCIM Workshop on Software Evolution 2006* (Université des Sciences et Technologies de Lille, France, pp. 57-64).