Computer Communications 36 (2013) 386–395

Contents lists available at SciVerse ScienceDirect

Computer Communications

journal homepage: www.elsevier.com/locate/comcom

Analyzing web server performance under dynamic user workloads

Raúl Peña-Ortiz *, José A. Gil, Julio Sahuquillo, Ana Pont

Universitat Politècnica de València, Departament d'Informàtica de Sistemes i Computadors, Camí de Vera s/n, 46022 Valencia, Spain

ARTICLE INFO

Article history:
Received 23 April 2012
Received in revised form 13 November 2012
Accepted 15 November 2012
Available online 29 November 2012

Keywords:
Web performance evaluation
Web dynamism
User's dynamic behavior
Dynamic web workload

ABSTRACT

The increasing popularity of web applications has introduced a new paradigm where users are no longer passive web consumers but they become active contributors to the web, specially in the contexts of social networking, blogs, wikis or e-commerce. In this new paradigm, contents and services are even more dynamic, which consequently increases the level of dynamism in user's behavior. Moreover, this trend is expected to rise in the incoming web.

This dynamism is a major adversity to define and model representative web workload, in fact, this characteristic is not fully represented in the most of the current web workload generators. This work proves that the web user's dynamic behavior is a crucial point that must be addressed in web performance studies in order to accurately estimate system performance indexes.

In this paper, we analyze the effect of using a more realistic dynamic workload on the web performance metrics. To this end, we evaluate a typical e-commerce scenario and compare the results obtained using different levels of dynamic workload instead of traditional workloads. Experimental results show that, when a more dynamic and interactive workload is taken into account, performance indexes can widely differ and noticeably affect the stress borderline on the server. For instance, the processor usage can increase 30% due to dynamism, affecting negatively average response time perceived by users, which can also turn in unwanted effects in marketing and fidelity policies.

1. Introduction

There are few technological success stories as dramatic as that of the web. Originally designed to share static contents among a small group of researchers, today the web is used by many millions of people as a part of their daily routines and social lives. Our society is progressively becoming more densely connected, and the paradigm where users access the web from a desktop computer is making way for a new paradigm dominated by pervasive electronic devices like smart phones and tablets.

This incessant evolution has been possible thanks to the continuous changes in technology that have introduced new features in the current and incoming web, both in its applications, users, and infrastructure [1]. For instance, e-commerce systems, on-line social networks, blogs, wikis or service oriented architectures are some examples that manifest how websites are evolving from typical hypermedia information repositories of the first web generation (Web 1.0) to hypermedia distributed applications and services representative of the second generation (Web 2.0). With the emergence of this kind of applications and services, users are no longer passive consumers, but they become participative contributors to the dynamic content accessible on the web [2]. Nowadays, web contents and services are even more dynamic, which consequently increases the user's dynamic behavior [3]. Moreover, these changes are being more relevant and meaningful in the incoming web also referred to as Web 3.0 [4] or Future Internet [5].

As a system that is continuously changing, both in the offered applications and infrastructure, performance evaluation studies are crucial points that must be addressed in order to provide sound proposals when designing new web-related systems [6], such as web services, web servers, proxies or content distribution policies. As in any performance evaluation process, accurate and representative workload models must be used in order to guarantee the validity of the results. Regarding web systems, the implicit user's dynamism makes difficult the design of accurate web workload representing users' navigations. To deal with this shortcoming, we focused on modeling this dynamism in a previous work [7]. The resulting model is able to represent dynamic changes on users behavior during the same navigation session by adopting different roles (e.g. browsing or ordering roles in an e-commerce environment). These roles define different users reactions to the dynamic web contents and services.

Although real web users present dynamic behavior, many studies still check their approaches with traditional workloads; in spite that this way can lead to results quite distant from real workloads, as the experimental results will show. In this paper we analyze and

* Corresponding author. Tel.: +34 96 387 75 71; fax: +34 96 387 75 79.
E-mail addresses: rpenya@upvnet.upv.es (R. Peña-Ortiz), jagil@disca.upv.es (J.A. Gil), jsahuqui@disca.upv.es (J. Sahuquillo), apont@disca.upv.es (A. Pont).

0140-3664/$ - see front matter © 2012 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.comcom.2012.11.005

measure for the first time, to the best of our knowledge, the effect of using different levels of dynamic workload on web performance evaluation instead of traditional workloads. Results show that CPU utilization can increase as large as 30% with dynamic workloads. These more realistic workloads show that processor change is not uniformly balanced along time, but overloaded peaks rise when considering user's dynamic behavior in a more accurate way. As a consequence, the probability of a long response time is higher, and the number of user abandonments can increase (up to 40%).

The remainder of this paper is organized as follows. Section 2 discusses the reasons that motivated us to perform this work and reviews the state of the art in this field. In Section 3 we briefly describe the experimental testbed devised to carry out fair comparison studies. Section 4 presents the dynamic workloads proposals and how they can be modeled by the approaches under study. Then, Section 5 shows the effect of the different workloads on the web system performance. Finally, we draw some concluding remarks and future work in Section 6.

2. Background and motivation

As mentioned above, the new web technologies have a strong impact on the system performance. In this context, some previous attempts have been published. For instance, Cecchet et al. [8] investigate the effect of different J2EE application deployments on the performance scalability of application servers. Schneider et al. [9] point out that the use of AJAX and mashups generates more aggressive and bursty network usage compared to the overall HTTP traffic. Similar conclusions but considering server performance are presented in [10]. Unfortunately, these studies only consider specific web paradigms, thus the workload used is not representative enough of current users' navigations.

There are three main challenges that must be addressed in a progressive way when modeling the user's behavior on realistic dynamic workloads. First, the user's dynamic behavior must be modeled [6]. Then, the different user's roles in the web must also be characterized [11]. Finally, continuous changes in these roles must be represented and considered [12].

There have been few but interesting efforts to model user's behavior in order to obtain more representative dynamic workloads for specific web applications. Menascé and Almeida [13] introduced the Customer Behavior Model Graph (CBMG) that describes patterns of user's behavior in the workloads of e-commerce sites. CBMG is a workload model included in the TPC Benchmark™ W (TPC-W) [14], which is a commonly accepted benchmark when dealing with e-commerce website studies [15,16]. Duarte et al. [17] applied this model for workload definition of blogspace. Shams et al. [18] extended CBMG to capture an application inter-request and data dependencies. Benevenuto et al. [19] introduced the Clickstream Model to characterize user's behavior in online social networks. However, these models only characterize web workload for specific paradigms or applications, but they either do not model user's dynamic behavior for a general context and in an appropriate and accurate way (first challenge) or do not consider user's dynamic roles (second and third challenges).

These shortcomings motivated us to propose a new workload model called Dweb (Dynamic web workload model) [7], which permits to define dynamic web workload for general contexts, taking into account the mentioned challenges by introducing user's dynamic behavior in the workload characterization. Dweb introduces two concepts in order to consider different levels of user's dynamism. First, the *navigation concept* allows us to represent dynamic reactions of users when they interact with web content and services (first challenge). Second, user's roles and continuous changes among them (second and third challenges) are defined using *the workload test concept*. Based on the Dweb model, the Universal Generator of Dynamic Workload under WWW Platforms (GUERNICA) was also developed in order to produce dynamic workload mimicking the behavior of the real web users community.

In order to fulfill the mentioned gaps in web performance studies, this paper aims to analyze the effect of dynamic workload on the system performance by evaluating the impact of the user's dynamic behavior on the main system resources. The study shows that resource usage widely varies when properly modeling the user's dynamic behavior characteristic of the real workload.

3. Testbed overview

This section describes the testbed used to carry out the experiments, the experimental setup and the main measured performance metrics.

The testbed used in this work [20] provides support to evaluate the system performance considering traditional and dynamic workloads by using CBMG and Dweb models, respectively. To this end, the testbed integrates both TPC-W and GUERNICA in the same framework.

TPC-W is a transactional web benchmark that models an on-line bookstore environment, which is a representative e-commerce system. The TPC-W specification defines a full website map for the on-line bookstore that consists of 14 unique pages and their navigation transitions. Fig. 1 depicts a reduced TPC-W website map, where pages with related functionality are included in the same group: *ordering, shopping, browsing, admin* and *search*. Navigation hyperlinks among them are also indicated.

The *search* group provides a book searcher by using Search page to request the query and Search Results page to show a list of results. The *browsing* group embraces the Best-sellers and the New Products pages, which arrange the bookstore catalog according to the sales and the publication date, respectively. The *shopping* group is the largest set of pages and provides (i) sale functionality by managing the shopping cart (Shopping Cart page), (ii) the buy request and its confirmation (Buy Request page and Buy Confirm page, respectively), and (iii) the pay through a secured navigation (Customer Register page). The *ordering* group includes a set of pages that allows checking the order status (Order Inquiry page and Order Display page). The *admin* group is focused on managing the catalog of books (using Admin Request and Product Updated pages). Finally, the most referred pages (Home page and Product

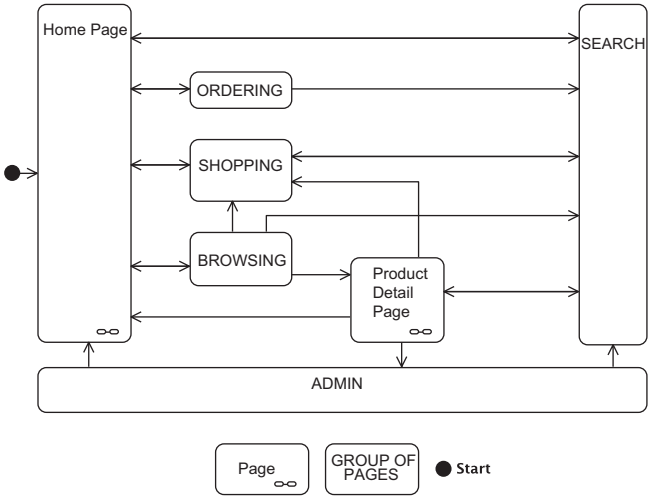


Fig. 1. TPC-W reduced website map.

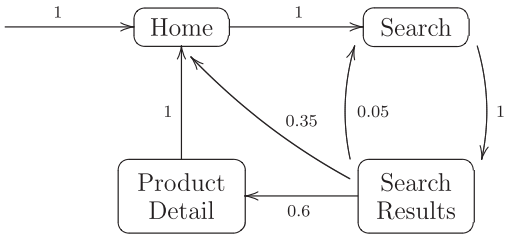


Fig. 2. Example of a simplified CBMG model for TPC-W.

Detail page) are also included. *Search* and *shopping* groups implement the most interactive and personalized functionality in the website, so they are potentially interesting as dynamic workload.

TPC-W uses CBMG model to reproduce the workload generated by multiple on-line browser sessions over the web application, which serves dynamic and static contents of the bookstore activities (e.g. catalog searches or sales). The CBMG model consists of all the website pages and the associated transition probability. Fig. 2 depicts an example of a simplified CBMG model for the search process of the on-line bookstore, showing that customers may visit several pages and move among these pages according to the arcs weight. Numbers in the arcs indicate the probability of taking that transition. For example, the probability of going to the Product Detail page from the Search Results page is 60%. This value means that after a search, regardless of whether the search returns a list of books or nothing, the Product Detail page will be visited 60% of the times.

Using this model, TPC-W defines three scenarios to characterize the web workload: shopping, browsing, and ordering. The shopping scenario presents intensive browsing and ordering activities while the browsing and ordering scenarios reduce ordering and browsing activities, respectively.

Based on the shopping scenario, we use Dweb to define more realistic cases where customers take decisions according to their own navigation objectives and the results of the visited pages (contents retrieved), and change their navigation roles generating dynamic navigations patterns as occurs in the real web. This behavior can be easily defined and generated as a dynamic workload by using Dweb and GUERNICA, respectively, but traditional approaches like CBMG generate an almost static approach. Because of this reason, we replaced TPC-W generation by the workload generation process of GUERNICA.

3.1. Experimental setup

The experimental setup used in this study is a typical two-tier configuration that consists of an Ubuntu Linux Server back-end and an Ubuntu Linux client front-end tier. The back-end runs the TPC-W server part, whose core is a Java web application (`TPC-W`

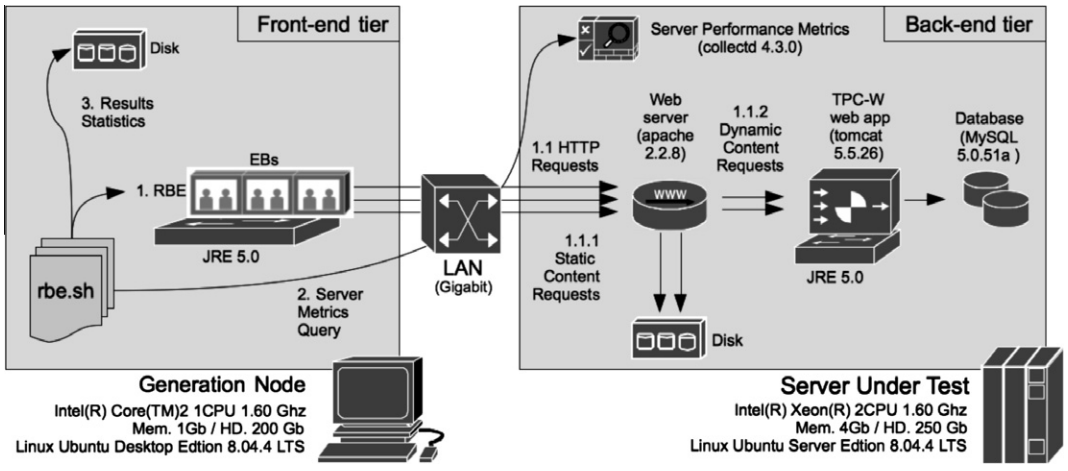


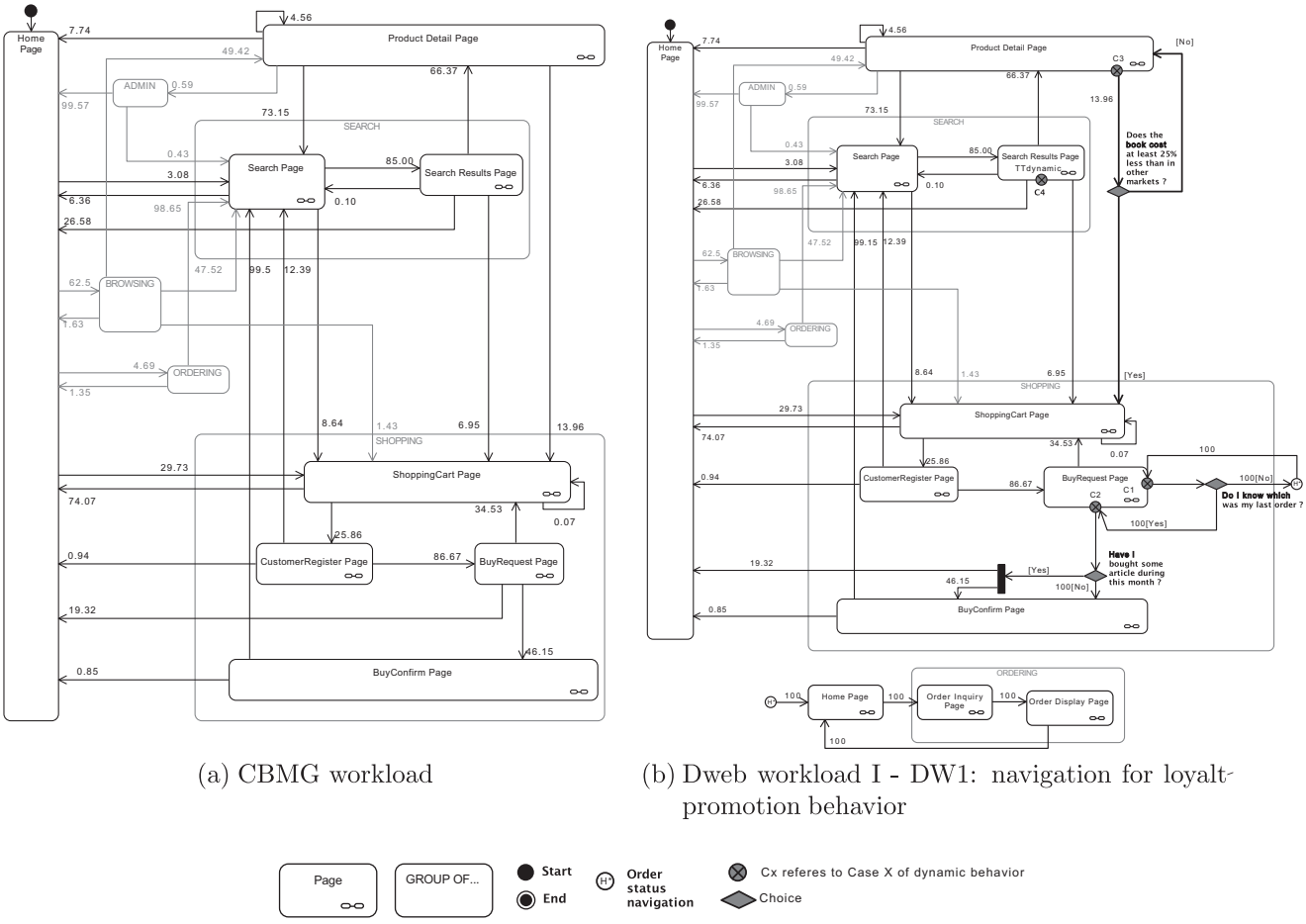
Fig. 3. Experimental setup.

Table 1
Performance metrics classification according to the evaluated resource.

Resource			Metric	Description/formula
Client side			Response time (WIRT)	Web Interaction Response Time (WIRT) is defined by TPC-W as $t2 - t1$, where $t1$ is the time measured at the emulated browsers when the first byte of the first HTTP request of the web interaction is sent by the browser to the server, and $t2$ is the time when the last byte of the last HTTP response that completes the web interaction is received
			Average Response Time ($\overline{WIRT}$)	$\overline{WIRT} = \frac{\sum_{i \in Pages} WIRT_i * Req_i}{\sum_{i \in Pages} Req_i}$
			Req_{page}	Requests per Page (Req_{page}) are the total number of connections for a page requested by emulated browsers and accepted by the server
Server side	Hardware	CPU	U_{CPU}	Metrics for hardware resources include utilization and throughput for the disk and the network
		Memory	U_{memory}	
		Disk	U_{disk}, X_{disk}	
		Network	U_{net}, X_{net}	
	Software	Apache	X_{apache}, CPU_{apache}, MEM_{apache}	Performance metrics for software components of server include: throughput, CPU and memory utilization, processes or threads, etc.
		Tomcat	X_{tomcat}, CPU_{tomcat}, MEM_{tomcat}	
		MySQL	X_{mysql}, CPU_{mysql}, MEM_{mysql}	

Table 2
Cases of dynamism in the loyalty promotion behavior.

Case	Description
1	If customers do not remember their last order status, they will check them by navigating into the *ordering* group of pages
2	Because the customer has an obligation of buying at least once a month to keep the discount, a buying session must finish with a payment when he has not bought anything during that month
3	A experienced customer only buys a book when its cost is 25% cheaper than in other markets
4	The higher the number of provided search results, the longer the time that a user takes to read and think about them



(a) CBMG workload

(b) Dweb workload I - DW1: navigation for loyalty promotion behavior

Fig. 4. Workload characterization for dynamic navigations.

`web app`) deployed on the Tomcat web application server [21]. Requests to static content of this web application, such as images, are served by the Apache web server, which redirects requests for dynamic content to Tomcat. TPC-W web application generates this type of content by fetching data from MySQL database. On the other hand, the front-end tier is able to generate the workload either using conventional or dynamic models. Both web application and workload generators are run on the SUN Java Runtime Environment 5.0 (`JRE 5.0`). Fig. 3 illustrates the hardware/software platform of the experimental setup.

Given the multi-tier configuration of this environment, system parameters (both in the server and in the workload generators) have been properly tuned to avoid that middleware and infrastructure bottlenecks interfere the results. TPC-W has been configured with 100 emulated browsers and a large number of items (100,000) that forced us to balance accesses in the database (e.g. pool connection size), static content service by Apache (e.g. number of workers to attend HTTP requests), or dynamic content service by Tomcat (e.g. number of threads providing dynamic contents). For each experiment, the measurements were performed during several runs having a 15-min warm-up phase and a 20-min measurement phase.

3.2. Performance metrics

Table 1 summarizes the performance metrics available in the experimental setup. The main metrics measured on the client side are the *response time* and the *total number of requests per page*. On the server side, the platform collects the server performance statistics required by the TPC-W specification (*CPU and memory utilization, database I/O activity, system I/O activity, and web server statistics*) as well as other optional statistics. These metrics allow a better understanding of the system behavior under test and permit to check the techniques used to improve performance when applying a dynamic web workload. The collected metrics can be classified in two main groups: metrics related with the usage of main hardware resources, and performance metrics for the software components of the back-end. For evaluation purposes, we

Table 3
Cases of dynamism in the new pre-sales promotion behavior.

Case	Description
1	Books that are Best-sellers or New Products cannot be added to the shopping cart because the bonus is only valid for common books
2	The customer can buy a book only if the cost of the resulting shopping cart is lower than the bonus value
3	A buying session (navigation session) finishes with a payment when the shopping cart cost is at least 75% of the bonus value
4	A customer leaves the website when the buying session finishes
5	The higher the number of provided search results, the longer the time that a user takes to read and think about them

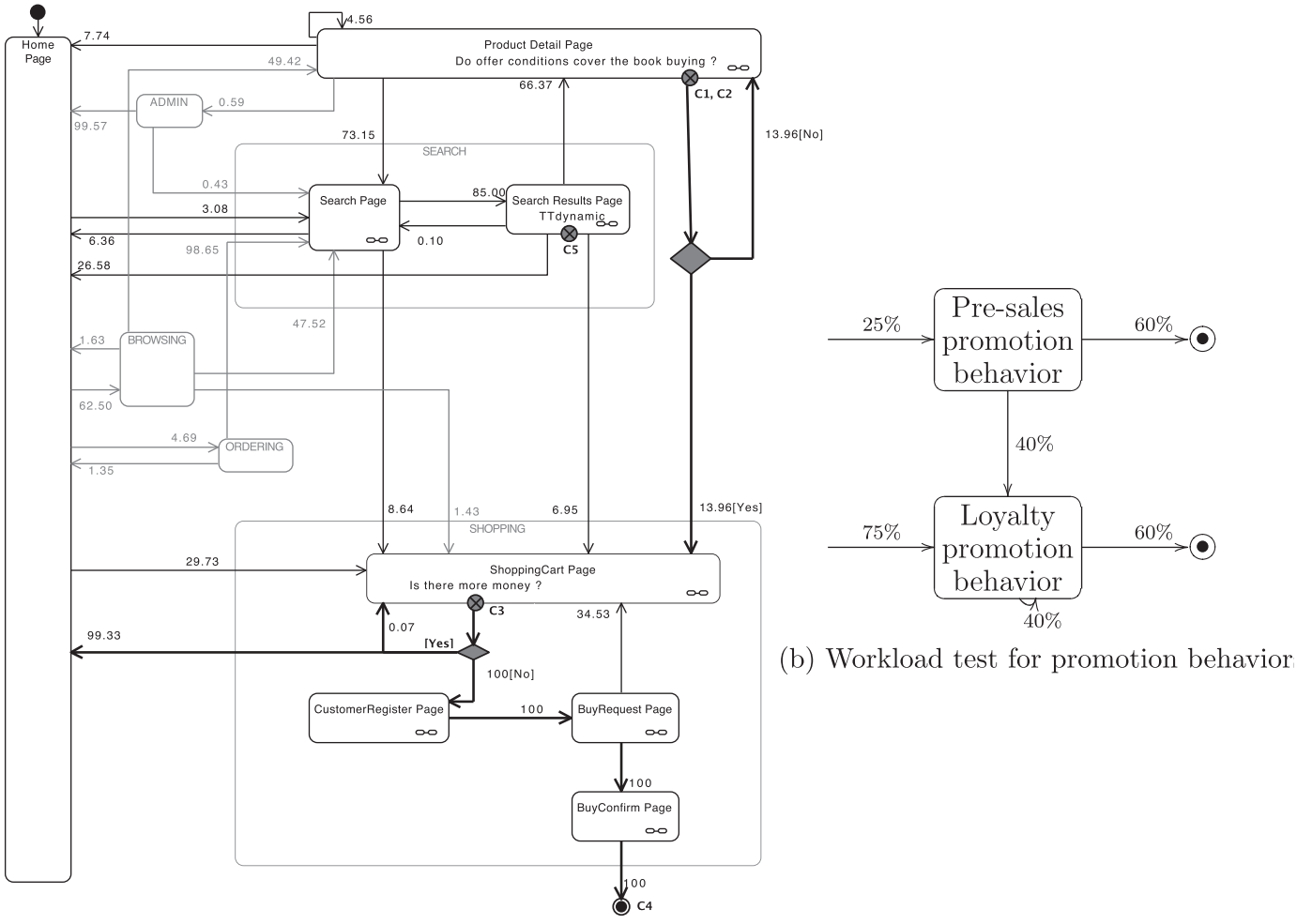


Fig. 5. Dweb workload II – DW2: characterization based on user's dynamic roles.

used a middleware named `collectd`[1] that collects system performance statistics periodically.

4. Workload design

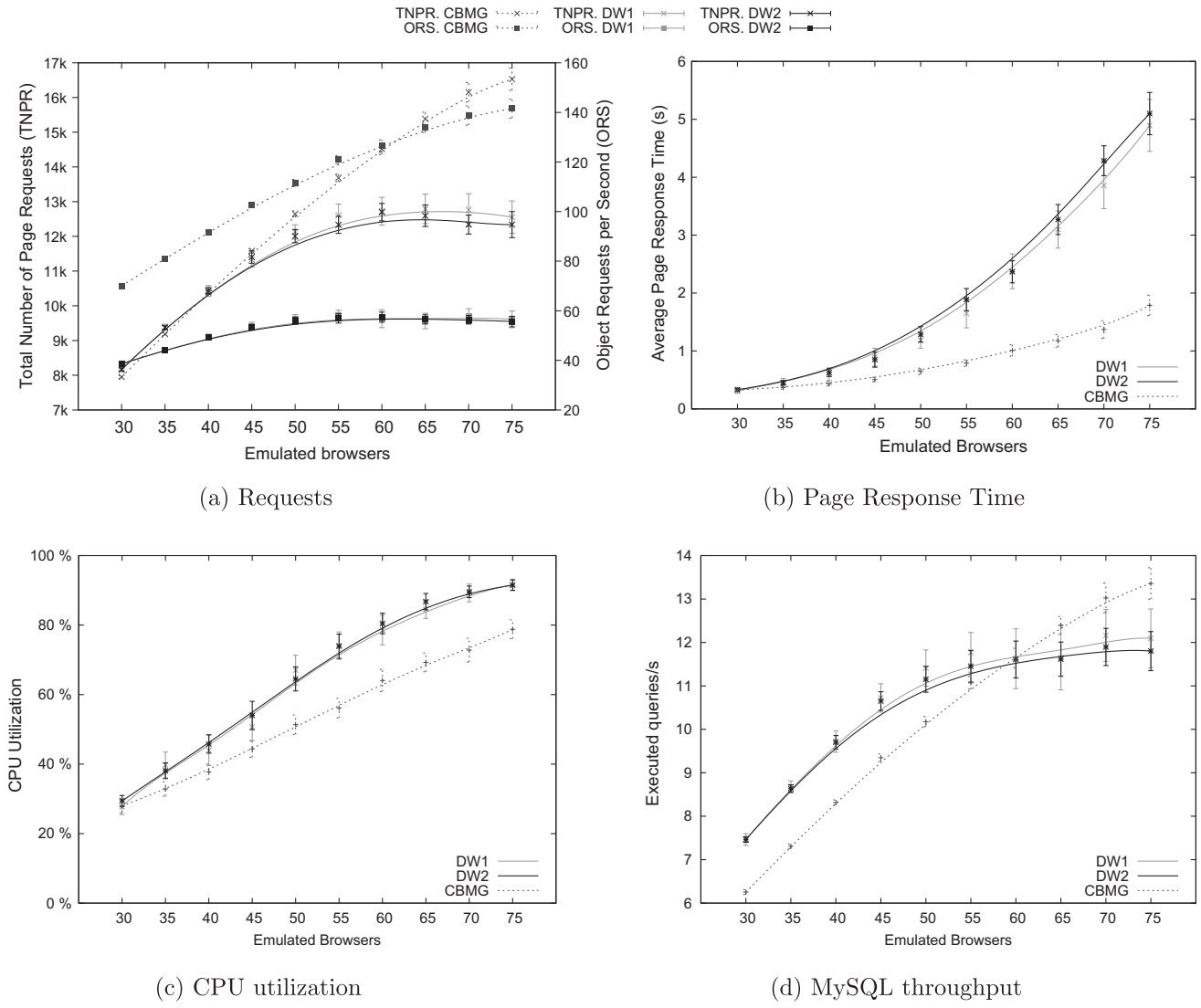
This section presents the experimental workloads used in the study. We would like to emphasize that the aim of this work is not to present a detailed dynamic workload based on current user's behavior, but is to explore how typical web performance metrics are affected by introducing different degrees of dynamism.

This work assumes the approach used by TPC-W to characterize traditional web workload and Dweb to introduce different levels of user's dynamism. First, Section 4.1 focuses only on user's navigation in order to represent dynamism on workload characterization by using the Dweb *navigation concept*. Second, Section 4.2 introduces a more realistic dynamic workload that also models changes on the user's roles by using the Dweb *workload test concept*.

4.1. Considering dynamism on user's navigations

In the first step, a dynamic workload (Dweb workload I – DW1) is defined with the aim of introducing user's dynamic navigations on workload characterization. For this purpose, we assume a common scenario of e-commerce where the main objective is to avoid the defection of customers. A large percentage of new customers – more than 60% in some sectors – defect before their third anniversary with an e-commerce website [22]. Consequently, these websites care deeply about customer retention and consider loyalty vital to the success of their on-line operations.

[1] <http://collectd.org/>.



(a) Requests

(b) Page Response Time

(c) CPU utilization

(d) MySQL throughput

Fig. 6. Main performance metrics values.

For the studied scenario, regarding customer retention in the on-line bookstore, we define a loyalty promotion consisting of a general discount only for those customers who buy at least once a month. The promotion introduces a new behavior with four cases of dynamism as summarized in Table 2.

Dweb allows the modeling of these cases of dynamism, which cannot be represented with this level of accuracy using traditional approaches such as CBMG. Fig. 4 shows the workloads generated using CBMG (Fig. 4(a), CBMG workload) and Dweb (Fig. 4(b), DW1 workload), focusing on *searching* and *shopping* groups, and highlighting their main pages and transitions. Both workloads consider the *think time* based on the TPC-W specification. This benchmark defines the user think time (TT) as $TT = T2 - T1$, where $T1$ is the time measured at the emulated browser when the last byte of the last web interaction is received from the server, and $T2$ is the time measured when the first byte of the first HTTP request of the next web interaction is sent from the emulated browser to the server. TPC-W considers that each think time must be taken independently from a negative exponential distribution, with the restriction that the average value must be greater than 7 s and lower than 8 s, as shown in Eq. (1). An important drawback of this approach is that it does not consider the results of the current search (i.e. web contents) on the actual user think time as occurs in real navigations.

$$TT_{TPC-W} = -ln(r) * p, \quad \text{where } (0 < r < 1) \wedge (7 \leqslant p \leqslant 8) \tag{1}$$

In contrast, Dweb allows defining a *dynamic think time* ($TT_{dynamic}$) according to the number of items returned by the search as shown in Eq. (2), which is closer to real web activities like the one defined in *case 4*.

$$TT_{dynamic} = -ln(r) * p, \quad \text{where } (0 < r < 1) \wedge \left(p = 7 + \frac{\text{Number of Search Results}}{\text{Max. Search Results}}\right) \tag{2}$$

The remaining cases of dynamism have been characterized using conditional transitions with the Dweb model. The transition from the Buy Request to the Buy Confirm pages depends on the last customer's purchase. If the customer does not buy any book during a given month, he has to commit the buying process, otherwise, he may finish the purchase or navigate to the Home page according to estimated probabilities of arcs as defined in *case 2*. Notice that, when a customer does not remember the date of his last purchase, he must visit the *ordering* group in order to find out it, as defined in *case 1*. Finally, *case 3* has been implemented in DW1 workload with a conditional transition between the Product Detail and the Shopping Cart pages. This transition only allows users to add a book to the shopping cart in case that its cost is 25% cheaper than in other markets.

Table 4
Consumption of CPU time (jiffies) by application.

EBs	Workload	Apache	Tomcat	MySQL
30	CBMG	7.71	4.55	886.48
	DW1	2.64	4.65	972.87
	DW2	2.61	4.93	1 044.66
35	CBMG	9.36	5.41	1 103.24
	DW1	3.44	6.47	1 390.73
	DW2	3.25	6.34	1 462.59
40	CBMG	11.01	6.45	1 301.23
	DW1	3.87	7.51	1 612.87
	DW2	3.86	7.92	1 693.67
45	CBMG	13.16	8.03	1573.35
	DW1	4.52	9.05	1 862.63
	DW2	4.67	9.59	2 003.51
50	CBMG	14.95	9.64	1760.76
	DW1	5.45	11.52	2 486.49
	DW2	5.32	11.45	2 398.61
55	CBMG	17.13	11.59	1979.87
	DW1	6.00	12.95	2703.26
	DW2	6.04	13.03	2732.06
60	CBMG	19.04	12.98	2171.47
	DW1	6.45	13.82	2868.51
	DW2	6.55	14.11	2 948.50
65	CBMG	21.97	15.42	2 489.40
	DW1	6.81	14.97	3 043.73
	DW2	7.00	15.19	3 152.04
70	CBMG	23.51	16.81	2 575.40
	DW1	6.97	14.99	3 190.34
	DW2	7.28	15.72	3 222.43
75	CBMG	25.68	18.08	2 748.36
	DW1	7.23	15.33	3 246.03
	DW2	7.32	15.80	3 264.57

4.2. One step ahead: evolving user's profile using dynamic roles

A common behavior characteristic of the web users community is the dynamism in the user's roles. In other words, the different roles that users adopt and the changes among them. Liu et al. [23] reported three phases of marketing in an e-commerce website that can induce the mentioned dynamic evolution of web users. These phases are: pre-sales, on-line, and after sales. The pre-sales phase includes company efforts to attract customers by advertising, public relations, New Products or service announcements, and other related activities such as discounts in some products or freebies (e.g. Apple promotions: back to school and 12 days of Christmas). Customers' electronic purchasing activities take place in the on-line sales where orders and charges are done through web facilities. The after-sales phase includes customer service, problem resolution, etc.

In the second scenario, we define a new Dweb workload (Dweb workload II – DW2) that reproduces how user's behavior evolves from pre-sales to on-line user profiles. We adopt the loyalty promotion behavior presented above as an example of on-line user profile, and define a new behavior based on a pre-sales promotion on the studied on-line bookstore. The pre-sales promotion consists of 1000 bonus to acquire common books in a buying session. This promotion should present a different user behavior with five cases of dynamism as summarized in Table 3.

Fig. 5 depicts the DW2 workload. This characterization defines the pre-sales promotion behavior using the Dweb navigation concept (Fig. 5(a)). *Case 1* and *case 2* have been implemented with a conditional transition between the Product Detail and the Shopping Cart pages. This transition depends on the users state (the available user's bonus) and the users navigation path. That is, it only allows users to add a book to the shopping cart in case that they arrive at the Product Detail page from other pages than the Best-sellers or New Products ones. The transition from the Shopping Cart to the Customer Register page depends on the content of the first page (*case 3*), and it implies the end of the user navigation when the buying process is committed (*case 4*). We also define a dynamic think time according to the number of items returned by the search (*case 5*).

Finally, we combine both promotion behaviors (i.e. behaviors for loyalty and pre-sales promotions) as dynamic roles by using the workload test concept of Dweb (Fig. 5(b)). The workload test automaton considers that the average response of the pre-sales promotion is 25% of users, based on the suggestions made in [24]. The transition between the pre-sales promotion and the loyalty promotion behaviors models the evolution from a new user to a loyal customer, according to the average percentage of customer retention (40%) reported in [22], while the arcs arriving to a final state represent the users defection (60%).

5. Impact of the dynamic workloads on web system performance

Experimental tests have been devised to compare performance metrics obtained with the CBMG workload versus those obtained with the Dweb workloads (DW1 and DW2). With the aim of finding out the stress borderline of the server for each workload, we varied the number of emulated browsers ranging from 30 to 75 in 5-user steps. All the experiments were done repeating 50 runs to obtain a margin of error with 99% confidence level.

Although experiments measured all the performance metrics listed in Table 1, only those present significative differences between traditional and dynamic workloads (e.g. number of requests, response time, CPU utilization and MySQL throughput) are shown in Fig. 6. Note that, in general, considering user's dynamism degrades the service conditions more than using traditional workloads, even though dynamic workloads generate a lower number of requests.

The *object requests per second* generated by the CBMG workload is 45–60% higher than the generated by dynamic workloads (see Fig. 6(a)). However, the response time for the Dweb workloads presents exponential curves with more pronounced slopes than the CBMG curve (see Fig. 6(b)), for instance, the DW2 workload increases the difference with respect to CBMG workload by 10%. On the other hand, the *total number of page requests* shows different values depending on the stressing server conditions (see Fig. 6(a)). When the server is characterized by a poor stress level (e.g. less than 40 browsers), the number of page requests generated by the dynamic workloads is by 3% higher than the generated by CBMG, because $TT_{dynamic}$ reduces idle times when there are simultaneous requests on a search process. However, the CBMG workload requests a higher number of pages than the Dweb workloads for a significative level of stress (e.g. more than 40 browsers), because the complexity of the dynamism reduces the service rate by increasing the service time per request.

Regarding the utilization of the main hardware resources (CPU, memory, network and disk), only the processor presents significant differences. As expected, the CPU utilization increases with the number of browsers as shown in Fig. 6(c). However, although the workloads present similar CPU utilization for a low number of browsers, differences between dynamic and traditional workloads can be as large as 30% when considering dynamism. Notice that the high CPU utilization values denote that the processor acts as the main performance bottleneck.

To better understand why the CPU utilization is so high, we studied how the main software components use the processor (Apache, Tomcat and MySQL). As observed in Table 4, MySQL almost monopolizes the processor time since its execution time is more than two orders of magnitude higher than the time devoted to Tomcat, specially with dynamic workloads. Fig. 6(d) shows the executed queries rate by MySQL for each workload. As observed, for a relatively low

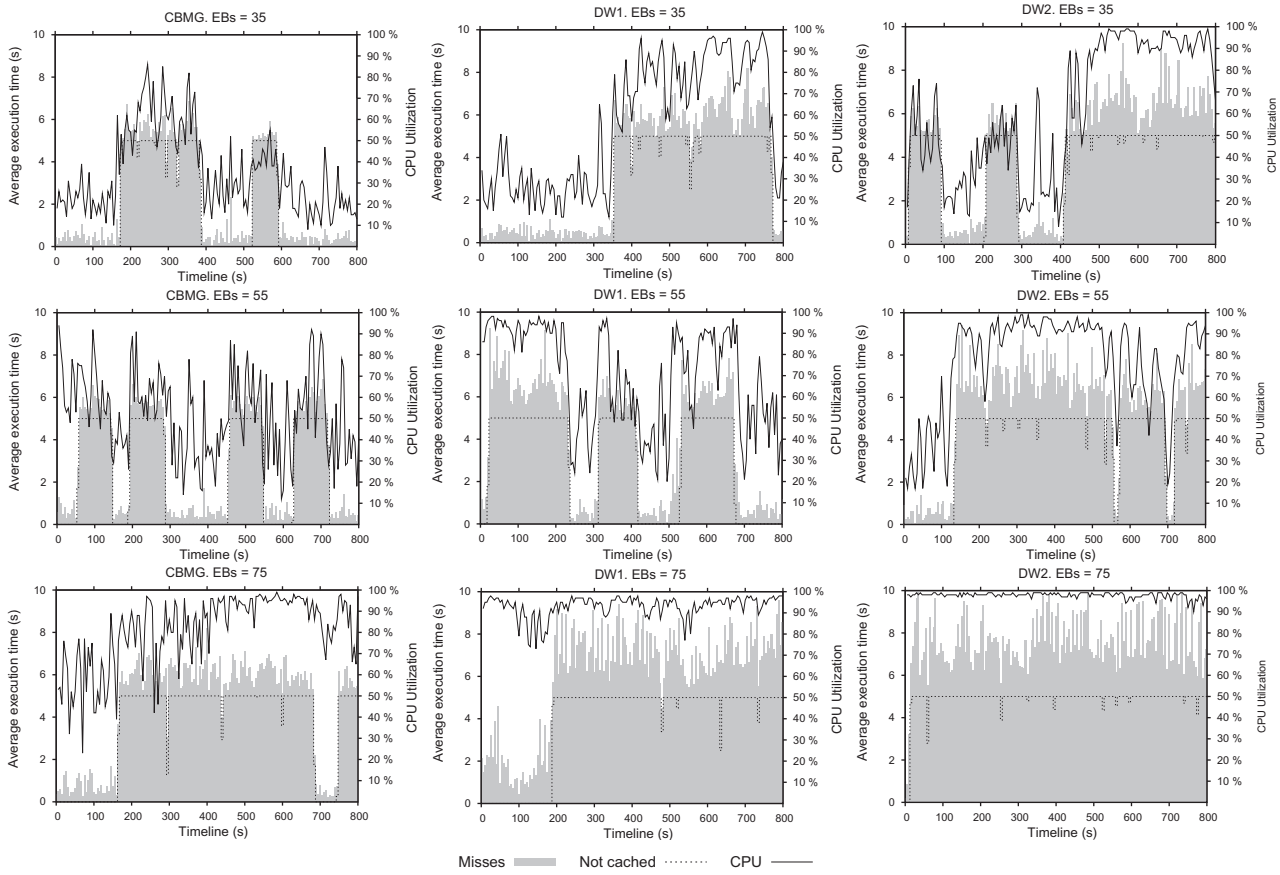


Fig. 7. CPU utilization by query cache status.

number of browsers (e.g. 45), this rate is by 15% higher when considering dynamism. But, more than 60 browsers produce that the number of executed queries becomes almost constant with dynamic workloads. That is due to a higher CPU utilization (greater than 80%) as depicted in Fig. 6(c). Consequently, MySQL database is the major candidate to be a software bottleneck.

With the aim of evaluating the impact of dynamism on server stress peaks, we analyze the database use done by dynamic workloads. But first, we need to understand how MySQL database works. This database includes `qcache` as a cache of executed queries, where queries result in *hit* or *miss*. We also distinguish *not cached* queries as a part of misses that cannot be cached for their dynamic nature or their complexity.[2] Listing 1 shows a *not cached* query example at TPC-W website.

Listing 1: *Not cached* queries example at TPC-W website

```
SELECT ol_i_id FROM orders, order_line
WHERE
  orders.o_id = order_line.ol_o_id
  AND NOT (order_line.ol_i_id = 93234)
  AND orders.o_c_id IN (
    SELECT o_c_id FROM orders, order_line
    WHERE orders.o_id = order_line.ol_o_id AND
    orders.o_id > (SELECT MAX(o_id) ...)
) ...
```

For a deeper study, we compared the cache status of the executed queries versus CPU utilization for several stress levels. Fig. 7 shows the values for 35, 55 and 75 emulated browsers as examples of poor stress, significative stress, and overloaded situation at the server side, respectively. DW2 workload (the right column) presents a higher number of misses than DW1 and CBMG workloads (center and left columns, respectively). CPU overload peaks become larger and larger with the increase of *not cached* queries, which is caused by the dynamic query nature, specially for the DW2 workload that presents higher increase than the DW1 workload. Note that hits are not represented because their execution time is around 1 ms while the execution time for misses might be more than 8 s.

Finally, we show an example to illustrate how the system level of stressing caused by user's dynamism might induce higher probability of user abandonment. According to a Jupiter Research report [25] commissioned by Akamai, web page rendering should be kept to not more than 4 s to avoid user abandonment. Fig. 8 depicts how dynamic workloads increase the probability that a response takes over 4 s, specially when considering changes of user's behaviors in a overloaded system. Consequently, if the server is not appropriately tuned, the extra workload induced by the user's dynamic behavior can increase the probability of user abandonment (up to 40%).

In summary, results show that considering user's dynamism when characterizing web workload affects system performance. Dynamism on workloads characterization introduces new patterns of HTTP requests. These patterns, in general, reduce the number of requests to objects but increases the number of requests to dynamic web content so incurring differences in the system performance metrics, specially on the processor usage and the database throughput. As a result, the server performance degradation affects service conditions, increasing the average response time that induces a higher number of user abandonments. Results have also proved that considering user's dynamic navigations on workload

[2] MySQL documentation: how the Query Cache Operates, <http://dev.mysql.com/doc/refman/5.1/en/query-cache-operation.html>.

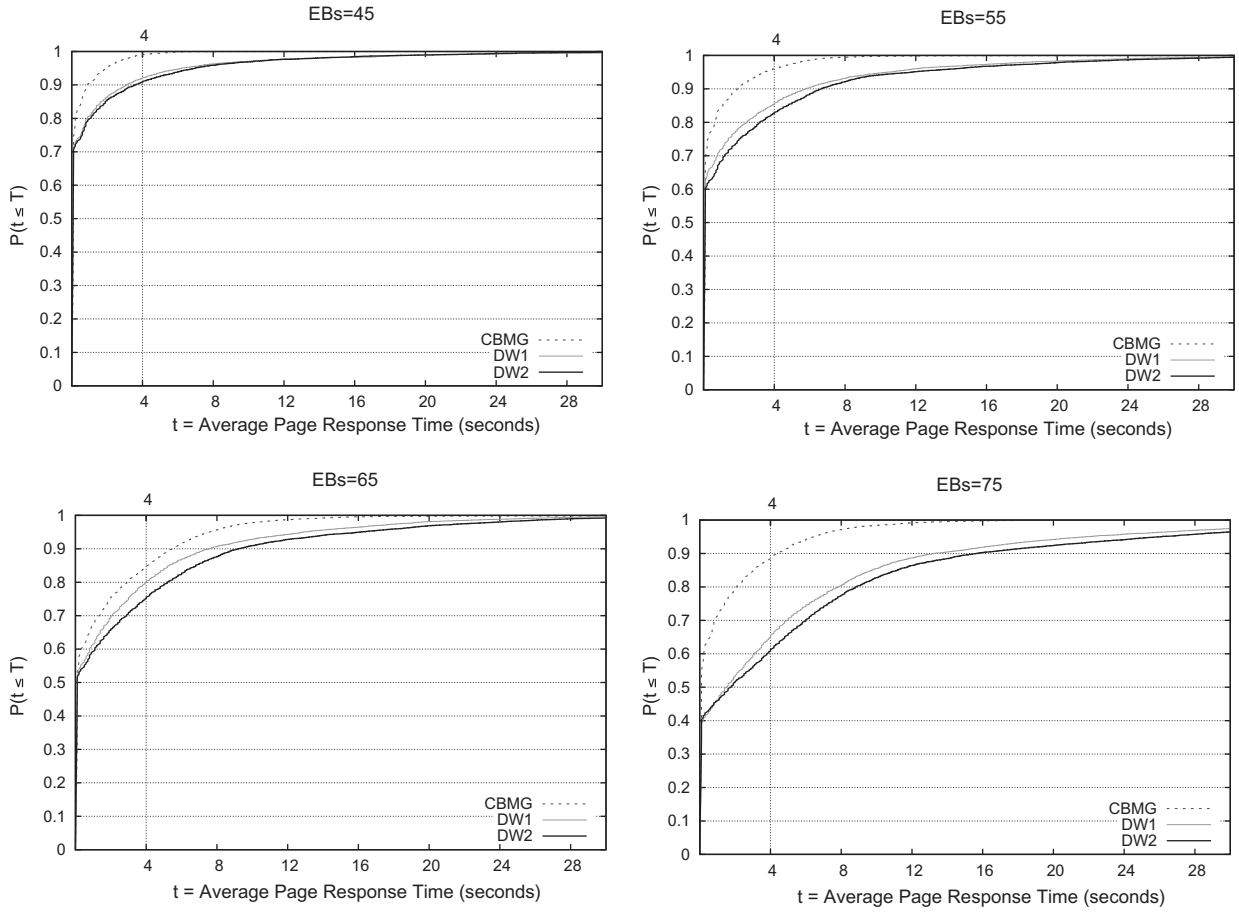


Fig. 8. Cummulative distribution for page response time.

characterization has a higher impact on the system performance metrics than modeling changes in the behavior because the dynamism is more present in the navigations than in the changes.

6. Conclusions and future work

The current web introduces a new paradigm where users are no longer passive consumers, but they become participative contributors to the dynamic web content accessible on the web. Consequently, this new dynamic behavior should be considered when evaluating the performance of current web systems.

This paper has explored the effects of using dynamic workload on web performance evaluation studies. The obtained results have been compared against those generated using traditional workloads. To this end, a scenario based on a typical e-commerce website has been recreated and different user's dynamic behaviors have been reproduced. We have used Dweb model to define the dynamic workloads because it permits easily to model these behaviors, which cannot be represented with this level of accuracy with traditional approaches. Furthermore, Dweb allowed us to define the workloads considering not only the dynamism in user's interaction but also the dynamic changes in the role when navigating the website. With the aim of evaluating the effect of these levels of dynamism on the system performance, a large and representative set of the most commonly used performance metrics has been measured for each experiment, such as client metrics (total number of page requests or response time), server hardware metrics (CPU utilization), or metrics for the main software components at the server (executed queries rate).

Experimental results have shown that dynamic workloads affect the stress borderline on the server, which is more degraded than when using traditional workloads. The object requests rate has decreased between 45% and 60%, but dynamic workloads have also changed the request nature, which has become more dynamic. This change has implied an important growth by 15% in the number of executed queries by the database with a significant increase in their dynamic nature that led the database to be an overloaded application. Consequently, the CPU utilization increased by 30% consolidating the processor as the main performance bottleneck. The server performance degradation affects the service conditions increasing the average response time exponentially, which yield to higher probability of user abandonment (up to 40%).

In addition, we observed that considering user's dynamic navigations on workload characterization has a stronger impact on the system performance metrics than modeling changes in the role, because dynamism is more present in the navigations than in modeling changes.

As for future work we plan to focus on a more extensive evaluation of the effect of real dynamic workload on the web performance, exploiting the characteristics and capabilities of Dweb model in a wider way.

Acknowledgment

This work has been partially supported by the Spanish Ministry of Science and Innovation under Grant TIN-2009-08201.

References

[1] P. Rodriguez, Web infrastructure for the 21st century, in: 18th International World Wide Web Conference, 2009.
[2] G. Cormode, B. Krishnamurthy, Key differences between Web 1.0 and Web 2.0, First Monday Journal 13 (6) (2008).

[3] T. O'Reilly, What is Web 2.0. Design Patterns and Business Models for the Next Generation of Software, O'Reilly Online Publishing, 2007.
[4] J. Hendler, Web 3.0 emerging, Computer 42 (1) (2009).
[5] G. Tselentis, Towards the Future Internet: A European Research Perspective, 2009.
[6] P. Barford, M. Crovella, Generating representative web workloads for network and server performance evaluation, in: Proceedings of the 1998 ACM SIGMETRICS Joint International Conference on Measurement and Modeling of Computer Systems, 1998.
[7] R. Peña-Ortiz, J. Sahuquillo, A. Pont, J. Gil, Dweb model: representing Web 2.0 dynamism, Computer Communications 32 (6) (2009).
[8] E. Cecchet, J. Marguerite, W. Zwaenepoel, Performance and scalability of EJB applications, SIGPLAN Notices 37 (11) (2002).
[9] F. Schneider, S. Agarwal, T. Alpcan, A. Feldmann, The new web: characterizing AJAX traffic, in: 9th International Conference on Passive and Active Network Measurement, PAM'08, 2008.
[10] M. Ohara, P. Nagpurkar, Y. Ueda, K. Ishizaki, The data-centricity of Web 2.0 workloads and its impact on server performance, in: IEEE International Symposium on Workload Characterization, 2009.
[11] H. Weinreich, H. Obendorf, E. Herder, M. Mayer, Off the beaten tracks: exploring three aspects of web navigation, in: 15th International Conference on World Wide Web, WWW 2006, 2006.
[12] S. Goel, A. Broder, E. Gabrilovich, B. Pang, Anatomy of the long tail: ordinary people with extraordinary tastes, in: Third ACM International Conference on Web Search and Data Mining, 2010.
[13] D. Menascé, V. Almeida, Scaling for E-Business: Technologies, Models, Performance, and Capacity Planning, Prentice Hall PTR, 2000.
[14] TPC BENCHMARK(TM) W Specification, Version 1.8, Tech. rep., Transaction Processing Performance Council, 2002.
[15] R. Dodge JR, D. Menascé, D. Barbará, Testing e-commerce site scalability with TPC-W, in: Computer Measurement Group Conference, 2001.
[16] C. Amza, E. Cecchet, A. Chanda, A. Cox, S. Elnikety, R. Gil, J. Marguerite, K. Rajamani, W. Zwaenepoel, Specification and implementation of dynamic web site benchmarks, in: Fifth Workshop on Workload Characterization, 2002.
[17] F. Duarte, M. Mattos, J. Almeida, V. Almeida, M. Curiel, A. Bestavros, Hierarchical Characterization and Generation of Blogosphere Workloads, Tech. rep., Computer Science, Boston University, 2008.
[18] M. Shams, D. Krishnamurthy, B. Far, A model-based approach for testing the performance of web applications, in: International Workshop on Software Quality Assurance, 2006.
[19] F. Benevenuto, T. Rodrigues de Magalhães, M. Cha, V. Almeida, Characterizing user behavior in online social networks, in: Ninth ACM SIGCOMM Conference on Internet Measurement Conference, 2009.
[20] R. Peña-Ortiz, J.A. Gil, J. Sahuquillo, A. Pont, Integración del comportamiento dinámico del usuario en TPC-W, in: XXXVII Conferencia Latinoamericana de Informática (CLEI'11), 2011, pp. 34–48.
[21] H.W. Cain, R. Rajwar, M. Marden, M.H. Lipasti, An architectural evaluation of Java TPC-W, in: Seventh International Symposium on High Performance Computer Architecture, 2001, p. 229.
[22] Frederick F. Reichheld, Phil Schefter, E-loyalty: your secret weapon on the web, Harvard Business Review 78 (2000) 105–113.
[23] C. Liu, K. Arnett, L. Capella, Web sites of the Fortune 500 companies: facing customers through home pages, Information & Management 31 (6) (1997) 335–345.
[24] Srini S. Srinivasana, Rolph Andersona, Kishore Ponnavolu, Customer loyalty in e-commerce: an exploration of its antecedents and consequences, Journal of Retailing 78 (2002) 41–50.
[25] Retail Web Site Performance: Consumer Reaction to a Poor Online Shopping Experience, Tech. rep., Jupiter Research a Division of Jupiter Kagan, Inc., 2006.