

Document downloaded from:

<http://hdl.handle.net/10251/66229>

This paper must be cited as:

Peña Ortiz, R.; Gil Salinas, JA.; Sahuquillo Borrás, J.; Pont Sanjuan, A. (2013). The impact of User-Browser Interaction on web performance. 28th ACM Symposium on Applied Computing (SAC 2013). ACM. doi:10.1145/2480362.2480497.



The final publication is available at

<http://dl.acm.org/citation.cfm?id=2480497>

Copyright ACM

Additional Information

© ACM 2013. This is the author's version of the work. It is posted here for your personal use. Not for redistribution. The definitive Version of Record was published in ACM, In Proceedings of the 28th Annual ACM Symposium on Applied Computing (pp. 695-702). <http://dx.doi.org/10.1145/2480362.2480497>

# The impact of User-Browser Interaction on web performance

Raúl Peña-Ortiz  
rpenya@upvnet.upv.es

José A. Gil  
jagil@disca.upv.es

Julio Sahuquillo  
jsahuqui@disca.upv.es

Ana Pont  
apont@disca.upv.es

Departament d'Informàtica de Sistemes i Computadors  
Universitat Politècnica de València  
Camí de Vera s/n, 46022 - Valencia, Spain  
Tel: +34 96 387 75 71, Fax: +34 96 387 75 79

## ABSTRACT

The user interaction with the current web contents is a major concern when defining web workloads in order to precisely estimate system performance. However, the intrinsic difficulty to represent this behavior in a workload model leads many research works to still use workloads non representative of the current web navigations. In contrast, in previous works we demonstrated that the use of an accurate workload model that considers user's dynamism when navigating the web clearly affects system performance metrics.

In this paper we analyze, for the first time, the effect of considering the User-Browser Interaction (UBI) as a part of user's dynamic behavior on web workload characterization in performance studies. To this end, we evaluate a typical e-commerce scenario and compare the obtained results for different UBI behaviors, such as the use of the back button and parallel browsing originated by using browser tabs or opening new windows when surfing a website.

## Categories and Subject Descriptors

H.4 [Information Systems]: World Wide Web—*Web applications*

## Keywords

Web performance evaluation, Web dynamism, User's dynamic behavior, Dynamic web workload, User-Browser Interaction

## 1. INTRODUCTION

Web applications and services have been introduced as a part of our daily life, evolving users from passive consumers to active contributors to the available dynamic content [9]. Furthermore, this trend is rising in the current Web where

desktop web interfaces are making way to mobile devices like smart phones or tablets, which increase users' contributions [21].

New interaction methods have influenced user's habits, such as tabbed browsing, mouse gestures or screen interaction. Consequently, web user's navigations are different than they were a few years ago [25]. Moreover, the most important behavioral changes concerning user's dynamism are still expected to come in the incoming Web.

Nevertheless, this kind of user's behavior is not properly modeled in performance evaluation studies, because typical workload models do not consider the implicit dynamism of users when characterizing their navigations. However, these studies are necessary to provide sound proposals when designing new web-related systems [6], such as web services, web servers, proxies or content distribution policies. Consequently, the lack of accurate and representative workload models can negatively affect the validity of the obtained results.

To deal with this shortcoming, we focused on modeling user's dynamism in a previous work [19]. The resulting model and web workload generator are able to consider user's dynamic behavior when characterizing and reproducing dynamic web workloads. By using them, we probed [18] that the dynamic behavior of web users is a crucial point that must be addressed in web performance studies in order to accurately estimate system performance indexes. Otherwise, performance studies can lead to results quite distant from the obtained with real systems.

This paper analyzes and measures the effect of considering the User-Browser Interaction (UBI) when modeling user's behavior on dynamic web workload characterization. To the best of our knowledge, this behavior has not been taken into account to generate workloads in existing web performance studies yet.

Results show that parallel browsing behavior, which is originated by using browser tabs or opening new windows when surfing the Web, permits users to increase the productivity of their navigations. That is, they achieve their objectives sooner than when browsing in a sequential way. This new behavior also affects the usage of the main resources of the system, whose bottleneck can move from one component to another.

The remainder of this paper is organized as follows. Sec-

tion 2 discusses relevant state-of-the-art work in this field. Section 3 briefly describes the experimental testbed devised to carry out fair comparison studies. In Section 4 we present the workload proposals that consider UBI when modeling user’s behavior. Section 5 shows the effect of the proposed workloads on the web system performance. Finally, we draw some concluding remarks and future work in Section 6.

## 2. MOTIVATION AND RELATED WORK

We are not aware of any previous work dealing with modeling UBI on dynamic web workload characterization. The closest pieces of related work are some previous attempts on modeling user’s behavior on workload characterization.

There are three main points that must be addressed 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 [25]. Finally, continuous changes in these roles must be represented and considered [12]. Nevertheless, the real challenge is to consider them all together in a general web scenario.

In contrast, the most relevant works in the open literature to model user’s behavior in order to obtain more representative dynamic workloads have been proposed for specific web applications. For instance, the Customer Behavior Model Graph (CBMG) [15] describes patterns of user’s behavior in the workloads of e-commerce sites. This model was applied for workload definition of blogspace [11], and extended to capture an application inter-request and data dependencies [23]. The Clickstream Model was introduced by [7] to characterize user’s behavior in online social networks. However, as mentioned above, the main drawback of these approaches is not only that they focus in specific paradigms and applications, but also they do not propose a user’s model for a general context that can represent the dynamic behavior in an appropriate and accurate way (first challenge), and consider the different user’s dynamic roles (second and third challenges).

These shortcomings led us to propose the Dynamic WEB workload model (DWEB) and the Universal Generator of Dynamic Workload under WWW Platforms (GUERNICA) [19]. DWEB permits to model dynamic web workload for general contexts, taking into account the mentioned challenges by introducing user’s dynamic behavior in the workload characterization. GUERNICA was also developed in order to exploit DWEB on generating dynamic workload by mimicking the behavior of the real web users community.

On the other hand, there is an evidence for an important change of user interaction with the Web. For instance, a recent study showed that 57.4% of web sessions involve parallel browsing behavior [14]. This behavior was originally found in the experienced web users, who surf the Web by using multiple browsers tabs or windows to support backtracking or multitasking with the aim of enhancing their navigation productivity [5, 24]. Moreover, the history-back button, included in any current web browser, is still one of the world’s most heavily used user interface components in the web context, and accounts for up to 31% of all revisits [16].

This important change has been considered in several studies and tools to improve the website usability [4], to test web applications [10] or learning user preferences [22]. However, to the best of our knowledge, UBI behaviors have not

been taken into account when modeling user’s dynamism on workload characterization in web performance studies yet.

## 3. PERFORMANCE EVALUATION

The testbed used in this work [17, 13] results from the integration between GUERNICA and a commonly used benchmark for e-commerce. It was developed with the aim of providing support to evaluate the system performance considering dynamic workloads generated with the DWEB model.

This testbed models an on-line bookstore environment, which is a representative e-commerce system. Figure 1 depicts a simplified website map for the on-line bookstore, where pages with similar functionality belong to the same group: *ordering*, *shopping*, *browsing*, *admin* and *search*. Navigation hyperlinks among them are also indicated.

A book searcher is provided by the *search* group, which embraces the Search and the Search Results pages to request the query and to show the list of results, respectively. The bookstore catalog is arranged according to the sales and the publication date by the pages of the *browsing* group (Best-sellers page and New Products page). The *shopping* group provides a sale functionality by managing the shopping cart (ShoppingCart page) and a buying process (Buy Request page and Buy Confirm page) that finishes with a payment through a secure navigation (Customer Register page). The *ordering* group (Order Inquiry page and Order Display page) offers the required functionality for checking the order status. The *admin* group manages the catalog of books. Finally, Home and Product Detail pages are also included since they are the most referenced ones.

Next, the main features of both the experimental setup and the performance metrics are summarized.

### 3.1 Experimental setup

The experimental setup used in this study is a typical two-tier configuration consisting of an Ubuntu Linux Server back-end tier and an Ubuntu Linux client front-end tier. The back-end runs the on-line bookstore, whose core is a Java web application (**web app**) deployed on the Tomcat web application server. Requests to static content of this

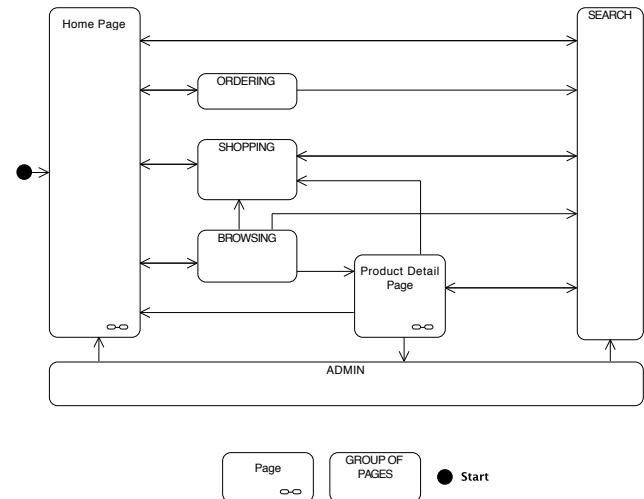


Figure 1: Testbed website map

web application, such as images, are served by the Apache web server, which redirects requests for dynamic content to Tomcat. The web application generates the dynamic content by fetching data from the MySQL database. On the other hand, the front-end tier is able to generate the workload using the DWEB model. Both web applications and workload generators are run on the SUN Java Runtime Environment 5.0 (JRE 5.0). Figure 2 illustrates the hardware/software platform of the experimental setup used in this work.

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. The on-line bookstore has been configured with 100 Emulated Browsers (EBs) and a large number of items (100,000 books) that forced us to balance accesses in the database (e.g. the pool connection size), static content service by Apache (e.g. the number of processes to attend HTTP requests), or dynamic content service by Tomcat (e.g. the number of threads providing dynamic contents). For each experiment, the measurements were performed during several runs with a 20-minute collecting-data phase after a 15-minute warm-up period.

### 3.2 Performance metrics

The experimental setup provides more than one hundred metrics, which can be classified in two main groups according to the side they are collected: metrics from the client side and metrics from the server side.

The *response time* and the *total number of requests per page* are the main metrics used in performance studies on the client side, but we also considered other metrics related to the user’s navigation sessions, such as the *number of finished sessions*, the *session length* or the *number of visited pages during a session*.

On the server side, we used a middleware named `collected[1]` that gathers system performance statistics periodically. This middleware collects the required server performance statistics that can be classified into two main groups according to the kind of resource they evaluate: metrics related to the usage of main hardware resources (*CPU and memory utilization, system I/O activity and network statistics*), and performance statistics for the software components of the back-end (*web server statistics, web application server activity and database I/O metrics*).

## 4. WORKLOAD DESIGN

This work uses DWEB to introduce different levels of user’s dynamism on workload characterization, specifically dynamism related to UBI behaviors. Section 4.1 focuses on user’s goals when surfing a website and defines user’s navigations according to this end. After that, a more realistic dynamic workload is defined in order to consider rapid return to recently visited pages by using the history-back button on web browsers. Section 4.2 introduces parallel tab browsing behavior on workload characterization.

### 4.1 The back button: rapid return to recently visited pages

An important factor of the back button success is the chance of rapid return to recently visited pages [8], which can avoid new HTTP requests and consequently changes the causes of the stressing conditions of a given website.

Case	Description
1	If customers do not remember their last order status, they will check them by navigating into the <i>ordering</i> 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.
5	A customer leaves the website when the buying session finishes because his goals have been satisfied.

Extra case in the extended behavior

6	A customer can return to recently visited listings of books (browsing listings or search results) without repeating a request to the web server by using the back button.
---	---

**Table 1: Cases of dynamism in the loyalty promotion behaviors**

However, only certain visited pages can be cached by a web browser with the aim of going back using the back button. That is, some web contents such as audio or video streaming, dynamic contents or web forms are not cached according to their HTTP headers. For these types of contents the back button does not have effect because pages are completely reloaded.

In a first step, a dynamic workload (LOY) conducted by user’s goals is defined. For this purpose, we assume a common scenario of an e-commerce website that pursues to avoid the defection of customers as objective. A large percentage (more than 60% in some sectors) of new customers defect before their third anniversary with an e-commerce website [20]. Consequently, these websites care deeply about customer retention and consider loyalty of huge importance to the success of their on-line operations.

The scenario consists of an on-line bookstore that defines a loyalty promotion considering a general discount only for those customers who buy at least once a month. The promotion introduces a new user behavior with five cases of dynamism as summarized in Table 1.

Secondly, a new DWEB workload (LOYB) is defined by extending the LOY workload with an extra case of dynamism to characterize the back button on web browser (see Table 1).

Figure 3 and Figure 4 show the workloads generated using DWEB for the loyalty promotion behavior conducted by goals (LOY) and its extended version (LOYB) considering the back button, respectively.

For the less dynamic pages in the website (e.g. the Home page or the Product Detail page), both workloads assume a *think time* ( $TT$ ) defined in [3] as  $TT = T2 - T1$ , where  $T1$  is the time measured at the 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 browser to the server. This definition considers that each think time must be taken independently from a negative exponential

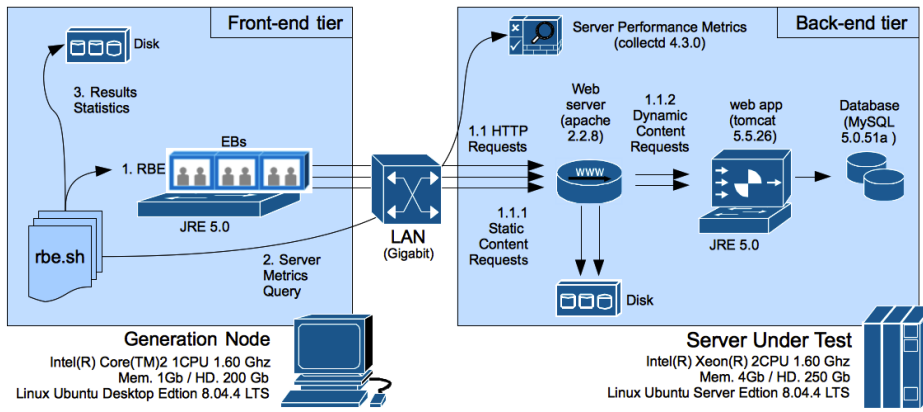


Figure 2: Experimental setup

distribution, with the restriction that the average value must be greater than 7 seconds and lower than 8 seconds as shown in equation (1). An important drawback of this approach is that it does not consider the results of the current search (web contents) on the actual user think time as occurs in real navigations.

$$TT = -\ln(r) * p, \text{ where } (0 < r < 1) \wedge (7 \leq p \leq 8) \quad (1)$$

In contrast, DWEB allows us to define a *dynamic think time* ( $TT_{dynamic}$ ) according to the number of items returned by the search as shown in equation 2, which is closer to real

web activities like the one defined in the *case 4* listed in Table 1. Both workloads use  $TT_{dynamic}$  in the Search Result page.

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

The remaining cases of dynamism have been characterized using conditional transitions with DWEB. The transition from the Buy Request to the Buy Confirm pages depends on the last customer's purchase. If the customer does not buy

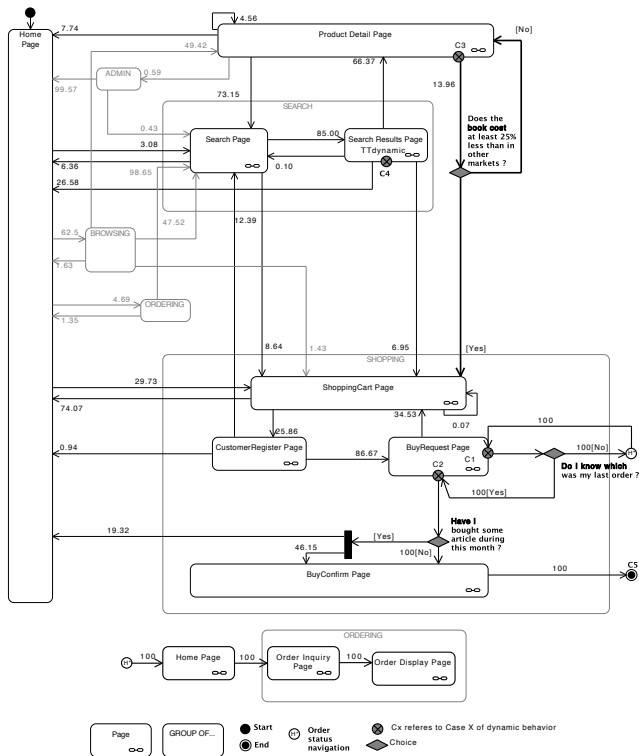


Figure 3: LOY workload: loyalty promotion behaviors conducted by goals

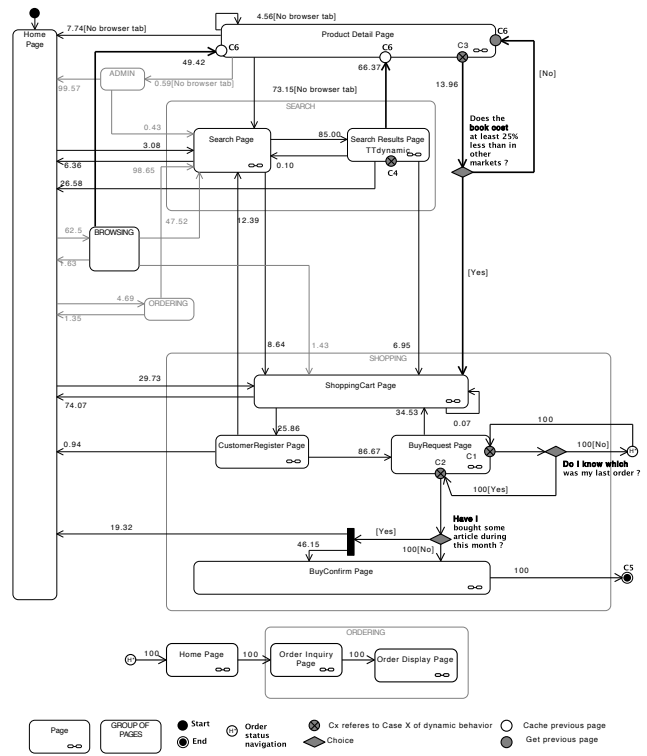


Figure 4: LOYB workload: LOY workload considering the back button

any book during a given month, he has to commit the buying process, and it implies the end of the navigation session (*case 5*). Otherwise, he may finish the purchase or navigate to the Home page according to estimated probabilities of arcs as defined in *case 2* and shown in Figure 3 and Figure 4. 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*. *Case 3* has been implemented with a conditional transition between the Product Detail and the Shopping Cart pages. This transition represents users adding a book to the shopping cart because its cost is 25% cheaper than in other markets. Finally, the back button has been characterized in LOYB workload as a cache that only stores the immediately previous listing page (*case 6*) as shown in Figure 4.

## 4.2 Optimizing user productivity: the parallel tab browsing behavior

Parallel browsing describes a behavior where users visit web pages in multiple concurrent threads by using web browsers tabs or windows. To help the understanding of how parallel browsing works, Figure 5 illustrates a parallel tab browsing session applied to the on-line bookstore. It shows how a web user uses parallel browsing based on three web browser tabs to improve his navigation time avoiding heavy searches of books. The user begins a navigation session in a window with the aim of buying a book that fulfills several requirements as soon as possible. After he visits some pages, he arrives at a search result. At this point of time, he starts a parallel tab browsing behavior by opening three new tabs with the detail of three different books selected from the results. After that, he takes some time to evaluate the content of each tab until he finds a book satisfying his requirements, such as *Think Time 4* or *Think Time 5* (see Figure 5) that

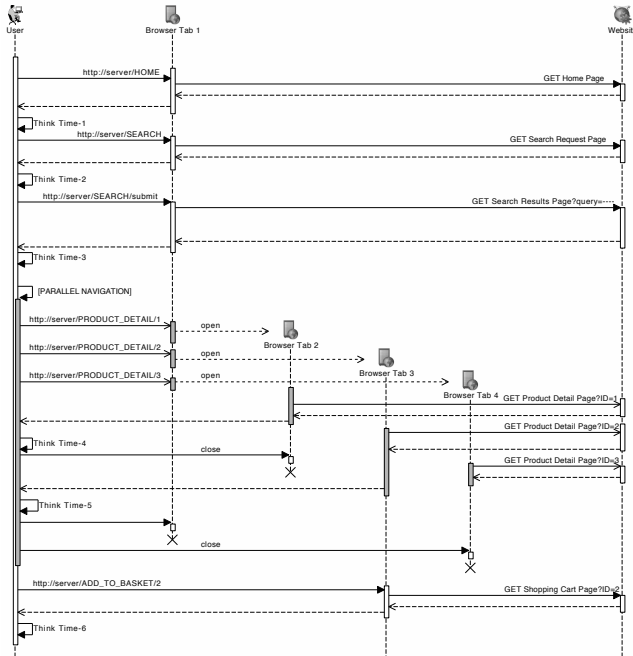


Figure 5: Example of parallel tab browsing session

are calculated according to equation (1). When a book does not fulfill the requirements, the user closes its associated tab and switches to the next one. Otherwise he discards the rest of tabs and adds the book to the shopping cart, finishing the parallel browsing session. Notice that the user does not take any *think time* on a book detail page when he discards its tab.

In the third scenario, we define a new DWEB workload (LOYT) that reproduces parallel tab browsing behavior in the loyalty promotion. With this aim, we extended the loyalty promotion behavior presented above by characterizing the navigation on web browser tabs as summarized in Table 2.

Figure 6 depicts the LOYT workload. This characterization defines the same behavior than LOY workload (Figure 3) except the navigations related to parallel tab browsing (*cases 6 and 7*). *Case 6* has been implemented with a pool of five parallel navigation threads, one for each tab. A navigation thread is killed when its book does not fulfill user's requirements, or when the user finds the required book in other thread, as defined in *case 7*. The navigation becomes sequential again when only a thread is kept alive. Notice that even though LOYT workload does not consider the possibility of opening new tabs from another existing tab in this study, DWEB and GUERNICA provide mechanisms to model and execute multi-level in tab branching, respectively, if required.

## 5. IMPACT OF UBI ON WEB SYSTEM PERFORMANCE

Experimental tests have been carried out to compare the performance achieved with the loyalty promotion workload (LOY) and that achieved with the extended versions (LOYB and LOYT) which consider UBI when modeling the user's behavior. All the experiments were done assuming 100 EBs and repeating 50 runs to provide an scenario with significant stress conditions and to obtain a margin of error with 99% confidence level, respectively.

New metrics, measured at the client side, have been defined in order to quantify the customer's productivity, such as *number of finished sessions*, *session length* and *number of visited pages per session*. Notice that a navigation session finishes when a customer leaves the website after his goals are satisfied. That is, after he buys some books. Thus, the higher the number of finished sessions, the higher the customer's productivity.

Although we measured all the performance metrics summarized in Section 3, only those showing significant differences in the studied workloads are discussed below.

The *number of finished sessions* for the UBI workloads

Case	Description
6	When a user has to review a listing of books, such as the result of a search or a browsing request, he begins a parallel tab browsing session with five tabs.
7	A user closes a tab when its book does not fulfill his buying requirements or when he has found the wished book in other tab.

Table 2: Extra cases of dynamism in the loyalty promotion behavior to represent parallel tab browsing

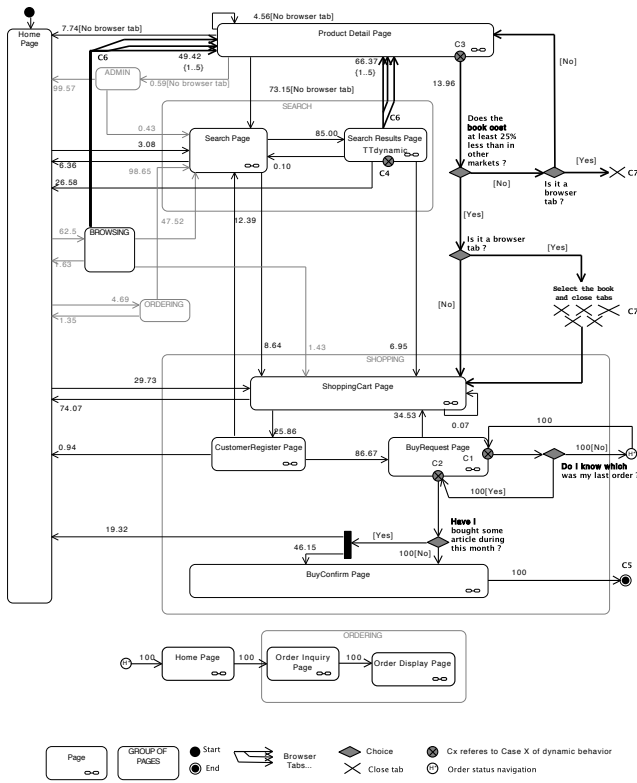


Figure 6: LOYT workload: parallel tab browsing behavior in loyalty promotion

are by 96.12% and 196.5% higher than the LOY workload because the *session length (seconds)* and the *number of visited pages per session* are lower in these workloads as shown in Table 3. Therefore, customer’s productivity is improved when user changes his navigation patterns as a result of parallel browsing behavior or using the back button in the defined scenario.

On the other hand, the UBI workloads also increase the server throughput, even though LOY workload degrades the service conditions more than considering the interactive behaviors as depicted in Figure 7. The *mean served pages* is by 16% and 101% higher for the UBI workloads (Figure 7a) because their user’s behaviors avoid heavy searches and increase requests to other pages, specially for the LOYT workload that presents a significant increase in the *number of detail pages served*. Consequently, the *Apache HTTP requests per second* generated by LOY workload is by 16% and 40% lower than generated by LOYB and LOYT workloads, respectively (Figure 7b).

Regarding the utilization of the main hardware resources

Metric	LOY	LOYB	LOYT
Num. of finished sessions	177.530	348.180	526.380
Session length (sec)	598.024	230.032	132.839
Num. of visited pages per session	78.909	26.146	18.814

Table 3: Customer’s productivity for 100 EBs

(CPU, memory, network and disk), only the processor presents significant differences. The *CPU utilization* decreases by 15% and 44% in the LOYB and LOYT workloads, respectively. Notice that the high CPU utilization value for the LOY workload denotes that the processor acts as the main performance bottleneck, which might differ for the UBI workloads (Figure 7c).

To better understand why the CPU utilization decreases, we studied how the main software components use the processor (Apache, Tomcat and MySQL). As observed in Table 4, MySQL almost monopolizes the processor since its execution time is more than two orders of magnitude higher than the time devoted to Tomcat and Apache. CPU consumption of MySQL for the UBI workloads is by 15.38% and 47.27% lower than that of the LOY workload, in spite of the executed queries rate for the UBI workloads is by 24% and 28% higher than the LOY workload as depicted in Figure 8a. Consequently, MySQL database is the major candidate to be a software bottleneck for the LOY workload, but it is not plainly clear for the LOYB and the LOYT workloads.

For a deeper study, we analyze how the database is used by these workloads. MySQL database includes *qcache*[2] as a cache of executed queries, where a query results in a *hit* or a *miss*. Figure 8b shows the average execution time for hits, misses and total queries. In general, the execution time of a query is by 40% and 85% lower for the UBI workloads than for the LOY workload, because avoiding heavy searches reduces the complexity of misses (their execution time decreases 98% and 99%) and increases the number of hits.

In summary, results show that considering user interaction with the web browser on workload characterization affects system performance. Modeling UBI behaviors introduces new patterns of HTTP requests. These patterns, in general, increase the number of requests to objects, but avoid heavy searches in the database allowing users to achieve a higher productivity. This fact causes noticeable differences in the performance metrics, specially in the processor usage and the server throughput. As a result, the stress borderline of the server presents significant changes, so the bottleneck can move to a different system resource.

## 6. CONCLUSIONS AND FUTURE WORK

The web evolution introduces a new paradigm where users become contributors to the dynamic contents and services offered. As a result, the behavior of users and their interaction patterns have evolved and are more active and dynamic. Consequently, web performance studies have to take into account this new user behavior in the workload models to guarantee the validity of the results.

This paper has explored the effects of considering UBI behaviors in web performance evaluation studies. To this

Software	LOY	LOYB	LOYT
MySQL	2883.360	2439.830	1520.414
Tomcat	22.542	24.454	27.841
Apache	9.157	9.422	10.534

Table 4: Main software components CPU time (jiffies) for 100 EBs

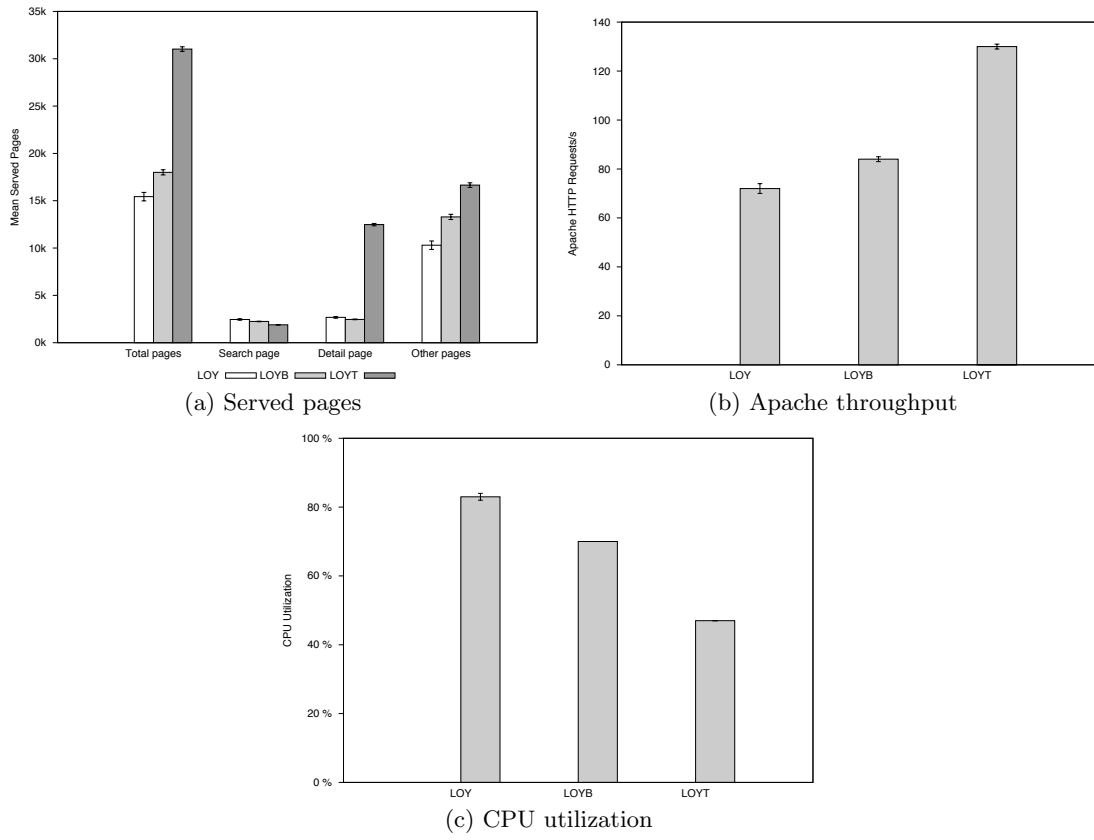


Figure 7: Main performance metrics values for 100 EBs

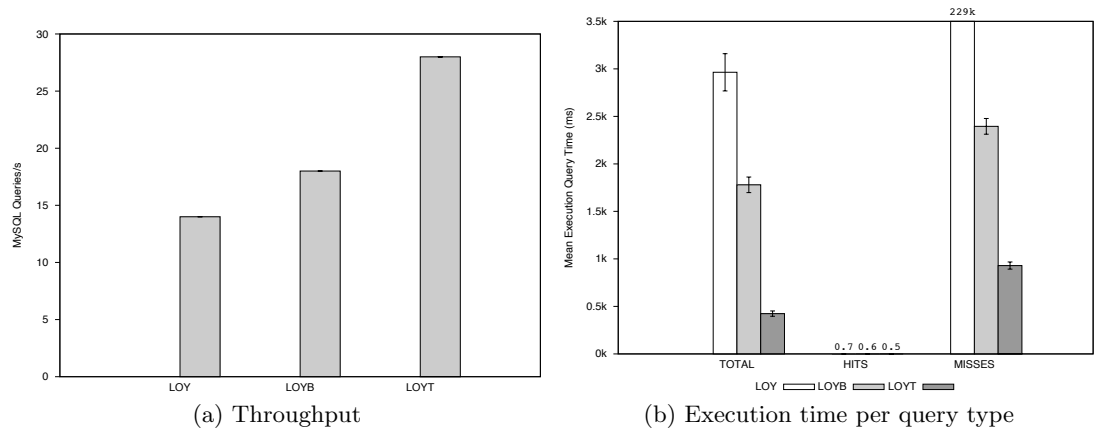


Figure 8: MySQL metrics values for 100 EBs

end, a scenario based on a typical e-commerce website has been recreated and a dynamic workload conducted by user's goals has been reproduced. Additionally, with the aim of improving the user's productivity, we included in the navigation patterns the use of the back button and the parallel tab browsing, as examples of UBI behaviors. DWEB model has permitted us to take into account these behaviors in an easy and flexible way when modeling web browsing patterns. The study has considered a wide set of traditional perfor-

mance metrics measured both at client side and server side. In addition, new metrics have been defined in order to quantify the productivity in the web navigations from the user point of view, such as number of finished session, number of visited pages per session or length session. This paper probed that navigations using the back button or opening new tabs, which result from considering a dynamic user interaction with the provided contents, clearly allow users to achieve their goals in less time, so increasing



productivity. These browsing patterns also affect the stress borderline on the server and the system resources utilization. Experimental results showed that parallel tab browsing behavior increases the user's productivity (measured in number of finished sessions) with respect to browsing the website in a sequential way by 196.5%. The new navigation patterns also increase the number of served pages (up to 101%). Nevertheless, they avoid heavy database searches so reducing the complexity of executed queries and decreasing their execution time (up to 85%). This means an important decrease in the processor utilization (up to 44%) and a noticeable increase in the web server throughput (up to 80%), that can generate a new bottleneck in the server components which are in charge of serving the requests.

As future work, we plan to focus on a deeper evaluation of the effect of dynamic user workloads on the web performance, exploiting the characteristics and capabilities of DWEB model in a wider way.

## Acknowledgements

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

## 7. REFERENCES

- [1] Collectd – The system statistics collection daemon. [online] <http://collectd.org/>.
- [2] MySQL documentation: How the Query Cache Operates. [online] <http://dev.mysql.com/doc/refman/5.1/en/query-cache-operation.html>.
- [3] TPC BENCHMARK(TM) W Specification. Technical report, Transaction Processing Performance Council, Feb. 2002.
- [4] E. Arroyo, T. Selker, and W. Wei. Usability tool for analysis of web designs using mouse tracks. In *Conference on Human factors in computing systems*, pages 484–489. ACM, April 2006.
- [5] A. Aula, N. Jhaveri, and M. Käkki. Information Search and Re-access Strategies of Experienced Web Users. In *Conference on World Wide Web*, pages 583–592, May 2005.
- [6] P. Barford and M. Crovella. Generating representative web workloads for network and server performance evaluation. In *SIGMETRICS '98/PERFORMANCE '98, Joint International Conference on Measurement and Modeling of Computer Systems*, pages 151–160, 1998.
- [7] F. Benevenuto, T. Rodrigues de Magalhães, M. Cha, and et al. Characterizing user behavior in online social networks. In *Internet Measurement Conference*, pages 49–62, 2009.
- [8] A. Cockburn and S. Greenberg. Issues of Page Representation and Organisation in Web Browser's Revisitation Tools. *Australasian Journal of Information Systems*, 7(2), May 2000.
- [9] G. Cormode and B. Krishnamurthy. Key differences between Web 1.0 and Web 2.0. *First Monday Journal*, 13(6), 2008.
- [10] G. Di Lucca and M. Di Penta. Considering browser interaction in web application testing. In *International Workshop on Web Site Evolution*, pages 74–81. IEEE, September 2003.
- [11] F. Duarte, B. Mattos, J. Almeida, V. A. F. Almeida, M. Curiel, and A. Bestavros. Hierarchical characterization and generation of blogosphere workloads. Technical report, October 2008.
- [12] S. Goel, A. Broder, E. Gabrilovich, and B. Pang. Anatomy of the long tail: ordinary people with extraordinary tastes. In *ACM International Conference on Web Search and Data Mining*, pages 201–210. Yahoo Research, 2010.
- [13] T. W. A. R. Group. GUERNICA and TPC-W integration. [online] [http://www.gii.upv.es/web\\_architecture/tools/GUERNICA\\_TPCW.zip](http://www.gii.upv.es/web_architecture/tools/GUERNICA_TPCW.zip).
- [14] J. Huang and R. W. White. Parallel browsing behavior on the web. In *Conference on Hypertext and hypermedia*, pages 13–18. ACM, June 2010.
- [15] D. A. Menascé and V. A. F. Almeida. *Scaling for E-Business: Technologies, Models, Performance, and Capacity Planning*. Prentice Hall, 2000.
- [16] H. Obendorf, H. Weinreich, E. Herder, and M. Mayer. Web page revisitation revisited: implications of a long-term click-stream study of browser usage. In *Conference on Human factors in computing systems*, pages 597–606, April-May 2007.
- [17] R. Peña-Ortiz, J. A. Gil Salinas, J. Sahuquillo, and A. Pont. Providing TPC-W with web user dynamic behavior. *CLEI electronic journal*, 15(2):1–12, August 2012.
- [18] R. Peña-Ortiz, J. A. Gil Salinas, J. Sahuquillo, and A. Pont. The impact of user's dynamic behavior on web performance. In *IEEE International Symposium on Network Computing and Applications*, pages 143–150, August 2012.
- [19] R. Peña-Ortiz, J. Sahuquillo, A. Pont, and J. A. Gil Salinas. Dweb model: representing Web 2.0 dynamism. *Computer Communications Journal*, 32(6):1118–1128, April 2009.
- [20] F. F. Reichheld and P. Schefter. E-Loyalty: Your Secret Weapon on the Web. *Harvard Business Review Magazine*, 78:105, December 2000.
- [21] P. Rodriguez. Web Infrastructure for the 21st Century. In *Conference on World Wide Web*. Telefónica I+D, January 2009.
- [22] Y. Seo and B. Zhang. Learning user's preferences by analyzing Web-browsing behaviors. In *International conference on autonomous agents*, pages 381–387. ACM, June 2000.
- [23] M. Shams, D. Krishnamurthy, and B. Far. A model-based approach for testing the performance of web applications. In *International Workshop on Software Quality Assurance*, pages 54–61, November 2006.
- [24] A. Thatcher. Web search strategies: The influence of Web experience and task type. *Information Processing & Management*, 44(3), May 2008.
- [25] H. Weinreich, H. Obendorf, E. Herder, and M. Mayer. Off the beaten tracks: exploring three aspects of web navigation. In *Conference on World Wide Web*, pages 133–142, May 2006.