A Hierarchical Characterization of User Behavior*

Adriano Pereira†‡  Gustavo Franco†  Leonardo Silva†  Wagner Meira Jr.†

†Federal University of Minas Gerais - e-SPEED Lab.
Av. Antônio Carlos 6627 - ICEx - CEP 31270-010
Belo Horizonte – Minas Gerais – Brazil
{adrianoc, gorgulho, leosilva, meira}@dcc.ufmg.br

‡Smart Price Research Center
Rua Lauro Jacques, 20 - Floresta
Belo Horizonte – Minas Gerais – Brazil
adrianoc@smartprice.com.br

Abstract

Understanding the characteristics of Internet services workloads is a crucial step to improve the quality of service offered to Web users. This paper presents a hierarchical and multiple time scale approach based on [16], which propose a characterization at the session, function, and request levels. This work extends it, adding new insights to these three levels and characterizing a new level that comprises the user behavior. The approach is illustrated by presenting a characterization of a proxy-cache server from one of the biggest Brazilian federal universities. Through this case study we show clearly how to apply the methodology considering some new analysis included for the three original levels (request, function, and session) and for the new one (user). This new level of characterization is the novelty of this research area, once it considers the interaction between users and servers, that answer their requests, in order to model user behavior. This modeling shows the reaction of the users to the system performance and clarify how the quality of the offered service affects their interaction with the computer system.

1 Introduction

Understanding the nature and characteristics of Internet services workloads is a crucial step to design systems with better performance and scalability, what is directly related to the quality of service experienced by the users. This can be done using characterization techniques, such as [16].

User behavior modeling can aggregate valuable information to the characterization methodologies, specially to understand the way users react to variations in the quality of service of Web applications. In this context, we want to characterize and model the typical behavior of Internet users and evaluate changes in the human interactions with the computer systems, when the performance of the system changes.

This work proposes a characterization methodology, based on a hierarchical and multiple time scale approach, to characterize Web services workloads. Analyzing the relation between the inter-arrival time (IAT) and the latency (time to process and answer the request) of the requests, we classify the user behavior into five different profiles, which determine how users behave during their interaction with the Web application servers.

This paper is organized as follows. In Section 2 we provide an overview of related work. Section 3 defines the methodology to characterize Web services workloads. We then discuss how to use the methodology, presenting a case study in Section 4. Finally we conclude in Section 5.

2 Related Work

Many studies tried to understand workloads of web applications and have identified several common characteristics like file size and popularity distributions, self-similarity in web traffic, reference locality and user request patterns, as can be seen in [3, 4, 9, 15]. Menascé et al. [16] and Veloso et al. [17] proposed hierarchical methodologies to characterize e-business and streaming media workloads, respectively, considering three levels of characterization: request, function and session. Nevertheless, these works do not have focus in the user behavior analysis.

Streaming media workload characterizations were studied in [1, 2, 17], however the user behavior understanding is superficial. [8] studied the correlation between requests, trying to determine tendencies in the user interaction process.

There are other works that deals with user interaction with Internet services. [7] tried to model the click-stream in the context of web advertising, and [11] considered the latency to study the user behavior, but in the specific context of a game application, and detected that network delay
has some effect on players’ behavior. [5] characterizes the user behavior in a public wireless network, considering distribution of the users, session duration, data rates, application popularity and mobility. [13] proposed a user behavior model framework, built in a top-down manner, consisting of various layers and based on mathematical models. This work was used to produce a user oriented workload generator [12].

At the best we know, our work has two main innovations. The first one is the user behavior analysis based on user session study and its correlation to the quality of the service provided by web application. The other one is the generic characterization methodology of web applications which considers explicitly the user behavior.

3 The Characterization Methodology

Our characterization methodology is based on a hierarchical and multiple time scale approach to characterize Web services workloads [15, 16]. We divide the methodology into four main levels: request, function, session, and user. Within each layer, an analysis across several time scales and criteria has to be conducted.

In the context of this work our main objective is to present the new characterization level denominated user level. We present here the new level and only some steps of the other levels, not presented in the original methodology, which can enrich the analysis of the workload. The user level intends to model the user behavior considering how the response time affects the user actions.

The idea of the hierarchy of levels is to guide the analysis of the workload into different views. This eases the characterization process, once it can be done according to different perspectives associated to each level. Therefore this process becomes clear and produce a more detailed characterization.

1. Request level characterization

In this level, we study the nature of the arrival process of the requests to allow for the extraction of statistically significant features towards classification, understanding, and modeling of request workload.

We extend the original methodology including the following perspectives: source of the request, nature of the request, bytes transferred associated with the request, inter-arrival time between requests, popularity of target domain and server, popularity of objects requested, and response time of the request.

2. Function level characterization

In this level we make a classification and analysis of the functions related to the web application being characterized. By function we represent an action of a given user that has a meaning for the service being characterized. For example in the context of an E-business application, the function could be the operations of search, browse, pay, etc. that are submitted by the user when interacting with the service.

We add the following analysis: source of the function, nature of the function, bytes transferred associated with the function, inter-arrival time between functions, popularity of target domain and server, and response time of the function.

3. Session level characterization

Session boundaries are delimited by a period of inactivity by a customer. In other words, if a customer has not issued any request for a period longer than a threshold $\tau$, his session is considered finished.

We analyze the sessions and add the following perspectives: session length, session duration, session composition in terms of functions, bytes transferred per session, and inter-arrival time between sessions.

4. User level characterization

Here we present the steps to analyze and characterize the user level. The objective of this level is to model the user behavior considering the offered quality of service.

(a) The log preparation: Generate a temporary log $Lu$ by putting together the user sessions;
(b) Analyze users from the following perspectives: inter-arrival times between requests of the same user, latency associated with requests of the same user, inter-arrival times and latency ratio, and inter-arrival times and latency difference;
(c) Discretize the set of values using a function that correlates inter-arrival times and latency. An approach to do this can consider the ratio and the difference between them. This function is useful in order to cluster the similar values considering user behavior and quality of service;
(d) Transform user sessions into sequences of classes using the discretization done in the last step;
(e) Mine the sequence of classes for each user session. To do this we recommend a sequence mining algorithm, such as [18];
(f) Evaluate the sequences in order to groups them according to similarity;
(g) Process the log $Lu$ applying a function $f(Lu)$, which maps sequence of classes to the defined groups of last step;
(h) Apply a clustering technique such as Key-Means to discover a new clusters of similar user sessions;
We apply the methodology at the request level, analyzing 4 weeks of logs comprising about 9 million requests issued from about 500 unique IP addresses statically assigned, generating a traffic of almost 90 Gbytes. As expected, 98% of the requests are HTTP and the other ones corresponds to FTP and SSL. A visual inspection of the number of requests arriving at the server on different time scales, i.e., in time intervals of varying length (see Figure 1), reveals, even to the inexperienced eye, an apparent strong dependence that shows long sequences of increase or decrease of volume (trends), particularly pointed at intermediate time scales. The purpose of our analysis is to decide whether these trends are purely due to changes in traffic volume during the day and week or whether there is predictable behavior beyond these cycles. It is important to be able to detect strong dependencies since they degrade estimation by increasing the variance of the estimation error. On the positive side, by detecting strong dependencies one can foresee not only mean behavior but also temporary phases of increase or decrease in volume and variability in workloads leading to a more accurate assessment of performance.

<table>
<thead>
<tr>
<th>Type of Request</th>
<th>#Requests</th>
<th>#MegaBytes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Image</td>
<td>1573731</td>
<td>4253</td>
</tr>
<tr>
<td>Query</td>
<td>326083</td>
<td>3038</td>
</tr>
<tr>
<td>Other</td>
<td>310193</td>
<td>6475</td>
</tr>
<tr>
<td>Directory</td>
<td>104535</td>
<td>906</td>
</tr>
<tr>
<td>HTML</td>
<td>69414</td>
<td>642</td>
</tr>
<tr>
<td>Lookup</td>
<td>22931</td>
<td>26</td>
</tr>
<tr>
<td>Text</td>
<td>9432</td>
<td>60</td>
</tr>
<tr>
<td>Exec</td>
<td>8900</td>
<td>1497</td>
</tr>
<tr>
<td>PDF</td>
<td>3368</td>
<td>204</td>
</tr>
<tr>
<td>SHTML</td>
<td>3064</td>
<td>39</td>
</tr>
<tr>
<td>Bundle</td>
<td>2434</td>
<td>698</td>
</tr>
<tr>
<td>Movie</td>
<td>1662</td>
<td>16539</td>
</tr>
<tr>
<td>Applet</td>
<td>1607</td>
<td>5</td>
</tr>
<tr>
<td>Software</td>
<td>1501</td>
<td>5</td>
</tr>
<tr>
<td>Audio</td>
<td>1096</td>
<td>15</td>
</tr>
</tbody>
</table>

Table 2: Frequency of Access by Request Type - Week 1

(i) Analyze the clusters and classify them according to the semantics associated with user behavior;
(j) Use the distribution of sessions in the clusters to conclude the user behavior analysis.

4 Case Study

This section presents a case study used to validate the characterization methodology presented in Section 3. Our purpose is to show the novelties of this methodology, more specifically the user level. In the other levels we will show only basic aspects and explain the main results.

We collect two months of data corresponding to log files from the Squid proxy-cache server of the Federal University of Minas Gerais (UFMG). The log contains the following information: timestamp, latency, source IP, resource requested, mime type, cache status of the resource, response time, and bytes transferred. We define latency as the amount of time spent to serve a request.

We perform a detailed analysis of this workload in order to determine the diversity of the user population, the variety of Web sites being accessed and the response time variation that the users experience while accessing these sites, among other criteria.

4.1 Request level characterization

Initially we organize the dataset in groups of weeks, and use the first four weeks to make the initial characterization. We analyze the log in order to verify the generality of the workload. This is the proxy-cache workload of one of the biggest federal universities from Brazil and it has a considerable number of requests per day. Its analysis shows that traditional characterizations aspects, such as object size, object popularity, and request arrival distribution, are equivalent to literature references in this research area [6, 16].

Table 1 presents general information about the log. As can be seen, there are more than two million requests per week, a significant amount of unique objects, unique IPs and user’s sessions.

We merge by timestamp the proxy-cache logs to produce a single log denoted by $L$.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td># Requests</td>
<td>2.44</td>
<td>2.10</td>
<td>2.15</td>
<td>2.08</td>
</tr>
<tr>
<td>MegaBytes</td>
<td>34.4</td>
<td>19.9</td>
<td>16.8</td>
<td>20.7</td>
</tr>
<tr>
<td># Unique IPs</td>
<td>488</td>
<td>485</td>
<td>497</td>
<td>507</td>
</tr>
<tr>
<td># Sessions</td>
<td>6952</td>
<td>6979</td>
<td>7369</td>
<td>7756</td>
</tr>
<tr>
<td># Unique Obj.</td>
<td>4.82</td>
<td>4.13</td>
<td>4.29</td>
<td>3.97</td>
</tr>
</tbody>
</table>

Table 1: General information about the logs

We merge by timestamp the proxy-cache logs to produce a single log denoted by $L$. We apply the methodology at the request level, analyzing 4 weeks of logs comprising about 9 million requests issued from about 500 unique IP addresses statically assigned, generating a traffic of almost 90 Gbytes. As expected, 98% of the requests are HTTP and the other ones corresponds to FTP and SSL. A visual inspection of the number of requests arriving at the server on different time scales, i.e., in time intervals of varying length (see Figure 1), reveals, even to the inexperienced eye, an apparent strong dependence that shows long sequences of increase or decrease of volume (trends), particularly pointed at intermediate time scales. The purpose of our analysis is to decide whether these trends are purely due to changes in traffic volume during the day and week or whether there is predictable behavior beyond these cycles. It is important to be able to detect strong dependencies since they degrade estimation by increasing the variance of the estimation error. On the positive side, by detecting strong dependencies one can foresee not only mean behavior but also temporary phases of increase or decrease in volume and variability in workloads leading to a more accurate assessment of performance.
When checking the amount of bytes transferred, the situation is the reverse. These two top-level domains account for 95% of the traffic, but the .com domain is the source of 73% of the bytes transferred while .br is responsible for 22%. These results are very similar to other reports [15, 16] on client workload while accessing Web services and demonstrate that our data is representative. The next levels are applied to the first week of this workload.

4.2 Function level characterization

A characterization in this level must include some actions as the generation of a temporary log with the relevant requests, the summarization of each of the function frequencies, the multi-scale analysis of the requests for each of the functions, and any other relevant analysis particular to any of them.

But in the context of our work, this level has not the same importance as for other applications, like an E-business service, because it is not possible to guess the function related to any of the requests, once this kind of information is particular to the application being used. When we examine the requests passing through the proxy-cache we can verify its type, duration and the URL being contacted, but the kind of processing that should be made is unknown.

4.3 Session level characterization

Considering only the requests to HTML objects and using a threshold of 1800 seconds, 3714 user sessions were identified for 371 unique IPs. Continuing the characterization of the session level, we generate a log $L_s$ for each session $s$ and plot two graphics: (1) session length (number of requests) and (2) session duration (in seconds).

Analyzing the session length we notice that more than 90% of the sessions are composed by at most 200 requests and the average session length is 21 requests. The session duration analysis shows that most of the sessions have less than 9000 seconds and the average session duration is 1350.6 seconds. The Figure 2 shows a considerable variation in the distribution of requests among sessions. More-over the observed average session length reforces that this case study is suitable to the main focus of our work, once sessions with very few requests do not provide enough information to model user behavior.

4.4 User level characterization

This subsection describes the user level characterization done for the case study according to the proposed methodology. First we generate a temporary log $L_u$ by putting together the sessions of each unique user, separated by sessions.

We analyze the use data from the following perspectives: inter-arrival times between consecutive requests, latency associated with user requests, inter-arrival times and latency ratio, and inter-arrival times and latency difference.

We generate the probability and cumulative distribution functions (PDF,CDF) of the ratio between inter-arrival times and latency. And we make the same using the difference of the two metrics. According to the methodology we can use then to discretize the set of values into classes, but in this case study this technique does not give a good result, thus we decide to use other discretization technique provided by the methodology.

Considering this, we discretize the set of values using functions that correlates inter-arrival times and latency, using ratio (RAT) and difference (DIF) metrics. These functions are defined as:
\[
DIF(k) = I(k, k+1) - L(k), \forall k \in Lu; \\
RAT(k) = \begin{cases} 
I(k+1)/L(k), & DIF(k)>0 \\
L(k)/I(k,k+1), & DIF(k)<0; \\
1, & DIF(k)=0
\end{cases}
\]

where \( k \) is a user request, \( I(k, k+1) \) is IAT between request \( k \) and \( k+1 \), and \( L(k) \) is the latency associated to the request \( k \).

The Figure 3 presents the discretization model we define considering the described functions. The \( x \) axis represents the DIF function and the \( y \) the RAT function. This model defines seven user classes (A to G), using two limit values for each axis. Values \( k_1 \) and \( k_2 \) divide the positive and negative sides of DIF function, defining a zone near zero, where we can not say much about the user behavior. This zone corresponds to values of IAT and latency very close to each other, which can represent situations such as: users who request objects and ask another one few seconds before this request answer arrives; and users who request objects and do not process the answer once they ask another one immediately after the request answer arrives. As can be seen in Figure 3, we group these interval between \( k_1 \) and \( k_2 \) in a group \( D \), which comprises the complete interval of vertical space. This decision was made because the RAT function does not cause significant change in this scenario. Values \( k_3 \) and \( k_4 \) break the vertical scale in three different zones, according to RAT function that quantify the correlation between IAT and latency. Considering this, we have three classes (A, B and C) in the correspondent side of values less than \( k_1 \). Classes A, B and C contain behaviors where users do not wait for the answer to their requests. In the correspondent side of values greater than \( k_2 \) there exists three classes (E, F and G). Classes E, F and G contain behaviors where users wait for the answer to their requests before ask for another ones. It is important to detach that as near from \( x \) axis (where \( y \) axis has value one) as close are values of IAT and latency, representing users who ask new objects faster after processing the last one (class E), or users that do not receive the object in desired time, but wait a significant time according to the latency of the respective object (Class C).

We transform user sessions into sequences of classes using this discretization. This transformation is a direct map one-to-one from application of functions RAT and DIF in each request of user session to a user class. Doing this we have a pair \( u(DIF ((k), RAT((k))) \) of user request, where \( k \) is the current request in the user session. This pair corresponds to a locale in the discretization model, defining the user class at this time.

Following the methodology we apply a sequence mining to each sequence of user classes obtained in the last step. It considers the occurrence order of the user actions, which is important to identify user behavior tendencies. We use the \textit{SPADE} algorithm [18], a fast discovery algorithm for sequential patterns. This algorithm utilizes combinatorial properties to decompose the original problem into smaller sub-problems, that can be independently solved in main-memory using efficient lattice search techniques and simple join operations.

The result of sequence mining give us the frequent sequences according to a mining support. This support is the percentage of transactions, user sessions in the context of our work, in which that sequence is present. We decide to use twelve as support value, once we identify that using it almost all classes (only class E does not appear) are present in the result of frequent sequences. From 3714 sessions, the result shows the distribution presented in Table 3.

<table>
<thead>
<tr>
<th>Class</th>
<th>Frequency</th>
<th>Percent. (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>1428</td>
<td>38,4%</td>
</tr>
<tr>
<td>B</td>
<td>685</td>
<td>18,4%</td>
</tr>
<tr>
<td>C</td>
<td>942</td>
<td>25,3%</td>
</tr>
<tr>
<td>D</td>
<td>1690</td>
<td>45,5%</td>
</tr>
<tr>
<td>E</td>
<td>275</td>
<td>7,4%</td>
</tr>
<tr>
<td>F</td>
<td>413</td>
<td>11,1%</td>
</tr>
<tr>
<td>G</td>
<td>3007</td>
<td>81,0%</td>
</tr>
</tbody>
</table>

Table 3: Sequence Mining - Distribution of User Classes

This sequence mining generates 9561 frequent sessions, varying from size 2 to 32. The most frequent sequence is \( G \rightarrow G \) which occurs in 2533 users sessions. We decide to adopt sequence of size three based on two arguments: involves four consecutive requests of the user, a significant quantity to extract a tendency in the user behavior; and presents a reasonable amount of frequent sessions to analyze. Using this we get 76 different sequences, from which \( G \rightarrow G \rightarrow G \) was the most popular (occurs in 59,1% of sessions).

Then we evaluate this sequences in order to group them according to similarity. To do this we base on user behavior tendency, defining what we denominate \textit{patience scale}, presented in Figure 4.

Considering the scale presented in Figure 4, we define
thirteen classes of sequences, which represent a user behavior tendency in the patience scale, with the following characteristics.

- **Class 1**: sequences that move right in the scale (e.g., $A \rightarrow D \rightarrow G$, or $A \rightarrow B \rightarrow C$, or $E \rightarrow F \rightarrow G$). Represents a user behavior patient tendency.

- **Class 2**: sequences that move left in the scale. This class represents an impatience tendency (e.g., $G \rightarrow D \rightarrow A$, or $C \rightarrow B \rightarrow A$, or $G \rightarrow F \rightarrow E$). Represents a user behavior impatient tendency.

- **Class 3**: sequences that present variation in positive side of the scale, including zero (e.g., $E \rightarrow G \rightarrow F$, or $G \rightarrow E \rightarrow F$, or $E \rightarrow D \rightarrow G$). Represents a variation in the user behavior, keeping the patient tendency.

- **Class 4**: sequences that present variation in negative side of the scale, including zero (e.g., $C \rightarrow A \rightarrow B$, or $A \rightarrow C \rightarrow B$, or $C \rightarrow D \rightarrow B$). Represents a variation in the user behavior, keeping the impatient tendency.

- **Class 5**: sequences that present fixed classes at zero (e.g., $D \rightarrow D \rightarrow D$). Represents a fixed tendency in the user behavior. Represents situations where the latency of the requested object and the inter-arrival time are very close. This is a typical web robot behavior or users whose tendency is not well-defined.

- **Class 6**: sequences that present fixed classes at positive side (e.g., $E \rightarrow E \rightarrow E$, or $G \rightarrow G \rightarrow G$). Represents situations where the latency of the requested object is smaller than the inter-arrival time, pointing out a tendency to keep the level of patience of the user behavior.

- **Class 7**: sequences that present fixed classes at negative side (e.g., $C \rightarrow C \rightarrow C$, or $A \rightarrow A \rightarrow A$). Represents situations where the latency of the requested object is greater than the inter-arrival time, pointing out a tendency to keep the level of impatience of the user behavior.

- **Class 8**: sequences that move from the negative side to the positive side and return to the negative side or zero - and sequences that move from the negative side or zero to the positive side and return to the negative side. (e.g., $C \rightarrow E \rightarrow B$, or $A \rightarrow G \rightarrow D$).

- **Class 9**: sequences that move from the positive side to the negative side and return to the positive side or zero - and sequences that move from the positive side to the negative side and then move left to a positive side class. (e.g., $B \rightarrow G \rightarrow F$, or $C \rightarrow F \rightarrow E$).

- **Class 10**: sequences that move from the negative side to the positive side and then move left to a positive side class. (e.g., $G \rightarrow A \rightarrow C$, or $E \rightarrow B \rightarrow C$).

- **Class 11**: sequences that move from the positive side to the negative side and then move right to a negative side class. (e.g., $G \rightarrow A \rightarrow C$, or $E \rightarrow B \rightarrow C$).

- **Class 12**: sequences that move from the negative side and then move to the positive side (e.g., $B \rightarrow A \rightarrow E$, or $C \rightarrow B \rightarrow F$).

- **Class 13**: sequences that move right from the positive side and then move to the negative side (e.g., $F \rightarrow G \rightarrow A$, or $E \rightarrow G \rightarrow B$).

After we identify the common sequences using sequence mining and define similar groups, which give us the thirteen classes described, we can process the log $Lu$, as described in our methodology. The new user session representation consists of sequences of classes, where each class is a patience scale tendency. Only the class number 10 has not occurred, and the classes 6, 1, 2, and 9 have presented the higher frequencies of occurrence in the sessions, respectively.

Following the methodology, we apply a clustering algorithm to discover clusters of similar user behaviors. In order to improve the quality of clustering analysis, once many groups difficult the understanding of the clustering semantics, we first define user profiles considering the semantics of the pre-defined groups. Table 4 presents these profiles.

<table>
<thead>
<tr>
<th>Id</th>
<th>Profile</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Impatient</td>
<td>3, 6</td>
</tr>
<tr>
<td>2</td>
<td>Patient</td>
<td>4, 7</td>
</tr>
<tr>
<td>3</td>
<td>Not Determined</td>
<td>5, 8, 9</td>
</tr>
<tr>
<td>4</td>
<td>Impatient Tendency</td>
<td>2, 11, 13</td>
</tr>
<tr>
<td>5</td>
<td>Patient Tendency</td>
<td>1, 10, 12</td>
</tr>
</tbody>
</table>

Table 4: User Profiles

We use a Java tool-bench for machine learning and data mining [10], which implements regression, association rules and clustering techniques. More specifically, we use the algorithms Key-Means(KM) and Expectation Maximization(EM) [14].

Here we present the results with $EM$, that show interesting conclusions related to user behavior. Without defining the number of resultant clusters, $EM$ algorithm defines seven clusters. The distribution of user profiles as presented in Table 4 for each cluster is presented in Table 5.

Analyzing the clusters, we can describe them as:
Cluster 1: Represents sessions where appear all profiles, except number 1 (Impatient). The Not Determined profile is the majority. But disregarding it Not Determined, once we can not say much about it, a Patient behavior is observed, with a balance between Patient Tendency and Impatient Tendency.

Cluster 2: Presents occurrence of three profiles (Patient, Patient Tendency, and Impatient Tendency) with majority to Patient profile.

Cluster 3: Represents sessions where appear all profiles, with majority to profile Patient tendency. Moreover this patient profile is reinforced with 24% of occurrence of Patient one.

Cluster 4: Only profile Impatient does not occurs. The Patient profile is the majority with almost half and can be observed a balance between Patient Tendency and Impatient Tendency.

Cluster 5: All the users clustered in this group presents a Patient profile during their sessions.

Cluster 6: All the profiles have been identified in this cluster, where a Patient Tendency profile is majority, but the Impatient profile (20%) dominates the Patient one (11%).

Cluster 7: The profiles Patient and Patient Tendency have been identified in this last cluster. It is a group of typical patient users, that maintain a patient tendency when some variation occurs in their behavior.

Table 5: Clusters - Distribution of User Profiles

<table>
<thead>
<tr>
<th>Cluster Id (%)</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0,0</td>
<td>14,5</td>
<td>49,0</td>
<td>19,5</td>
<td>17,0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0,0</td>
<td>40,0</td>
<td>00,0</td>
<td>43,5</td>
<td>16,5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>2,0</td>
<td>24,0</td>
<td>23,0</td>
<td>27,5</td>
<td>23,5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>0,0</td>
<td>49,0</td>
<td>16,0</td>
<td>17,0</td>
<td>18,0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>0,0</td>
<td>100,0</td>
<td>00,0</td>
<td>0,0</td>
<td>0,0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>20,0</td>
<td>11,0</td>
<td>20,5</td>
<td>26,5</td>
<td>22,0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>0,0</td>
<td>40,0</td>
<td>00,0</td>
<td>0,0</td>
<td>60,0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 6: Distribution of Sessions in Clusters

Not Determined profile and Patient Tendency; clusters 2, 3 and 7 present users with a Patient behavior, with Patient Tendency in situations where the users change their typical behavior; cluster 6 can be attributed to users who present a Impatient behavior, otherwise they have a Patient Tendency in situations where their typical profile change; cluster 5 represents the group of users that present a completely Patient behavior, once they always stay in the positive side of the patient scale; finally the cluster 4 consists of users who have a Patient profile and a balanced distribution of Impatient Tendency and Patient Tendency.

Table 7: User Behavior Distribution

<table>
<thead>
<tr>
<th>User Behavior</th>
<th>Frequency(%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Users with Not Determined profile and Patient Tendency</td>
<td>16,7</td>
</tr>
<tr>
<td>Users with Patient behavior with Patient Tendency in situations where the users change their typical behavior</td>
<td>39,7</td>
</tr>
<tr>
<td>Users who present a Impatient behavior, otherwise they have a Patient Tendency in situations where their typical profile change</td>
<td>3,3</td>
</tr>
<tr>
<td>Users that present a completely Patient behavior</td>
<td>20,5</td>
</tr>
<tr>
<td>Users who have a Patient profile and a balanced distribution of Impatient Tendency and Patient Tendency</td>
<td>19,8</td>
</tr>
</tbody>
</table>

From the analysis of the distribution of sessions in the clusters and the study of the behavior of each cluster we can summarize the results of user level characterization in Table 7. The next section presents our conclusions and on going work.

5 Conclusion

Several studies have been published regarding the workload of Internet services providers. However, none of them provides a generic methodology to model user behavior considering the quality of service, which can change clearly the client actions. This paper presents a hierarchical model for workload characterization based on [16], which proposed a characterization at the session, function, and request levels. Our work extends it, adding new insights to these three levels of characterization and a new level that comprises the user behavior.

We validate the model through a case study, using the proxy-cache server from one of the biggest Brazilian federal universities, where we show the novelties of the characterization model with emphasis on the user level.
This work is the base of future researches in at least two directions: (1) workload generation, considering our user behavior approach, which decreases the gap between the traditional models of characterization, based on typical statistical distributions, and the real workload; (2) innovative mechanisms to improve the scalability and the quality of Internet services.

Understanding better the workload characteristics, the service providers can improve their software performance engineering and capacity planning techniques.

References


