Understanding and Modeling the Behavior of Web Map Users

Vinícius G. Braga[1], Welder B. de Oliveira[2], Vagner J. do Sacramento Rodrigues[1], Kleber V. Cardoso[1]

[1] Instituto de Informática, Universidade Federal de Goiás, Brazil
{viniciusgoncalves, vagner, kleber}@inf.ufg.br
[2] goGeo, Brazil
welder.oliveira@gogeo.io

Abstract. Geographic Information System (GIS) in general, and Web mapping systems in particular, have become part of our set of digital tools for regular usage similarly to a search engine, an online social network, or a messaging system. However, the literature still lacks works that investigate and model how users interact with these systems. The well-known Web user models are too general, and are unable to accurately represent some specific common actions, such as pan or zoom in a Web mapping system. In this article, we describe how a user interacts with a Web mapping system and present a methodology that we have designed to retrieve this information from a traditional system, the Google Maps. We also develop a synthetic descriptive model of the user's behavior and we employ this model to create a proof-of-concept workload generator. Finally, we make a performance evaluation with another real-world GIS, called goGeo, to compare our workload generator with a conventional benchmark application. This shows how adding user features can significantly change the effective workload imposed on a GIS, and how this implies that performance evaluations of these systems in the past have often been based on imprecise assumptions.

Categories and Subject Descriptors: H.3.4 [**Information Storage and Retrieval**]: Systems and Software—*Performance evaluation (efficiency and effectiveness)*; H.4.0 [**Information Systems Applications**]: General

Keywords: GIS, Performance Evaluation, User Behavior, Web Map Tile Services, Workload Modeling

1. INTRODUCTION

Web mapping systems, such as Google Maps and Bing Maps, are examples of GIS that have become very popular tools. These systems are used in an independent, to carry out tasks such as finding a place or tracing a route, but they are also widely integrated into other applications. In this last approach, Web mapping systems may be used to give information about traffic conditions, weather forecast, location services, points of interest, and a wide range of other uses that can have financial, as well as social benefits. Moreover, there are several other applications that use Location Intelligence and Location Based Services in logistics, Business Intelligence, and CRM systems.

Two critical issues in the development of GIS applications are performance and scalability. In general, these issues arise from the large number of users, and the huge amount of data. Some GIS face additional challenges while employing large distributed systems to serve the users. The main factors that affect the performance of a system are design, implementation, and the workload imposed on the system [Feitelson 2015]. While the first two factors are usually taken into account by the GIS application developers, the last one tends to be neglected to some extent. Stress benchmarks are a very common approach for performance evaluation in GIS, but they usually fail to address the unique patterns of this kind of system [Guan et al. 2014]. This approach is useful for detecting bottlenecks and comparing systems; however it is not suitable, for example, for deployment planning, identification of the most demanded components in the system, or carrying out an effective evaluation of the user's Quality of Experience [Fiedler et al. 2010]. A properly designed workload generator can fill this gap.

This work has been partially supported by the CAPES, CNPq, and FAPEG.

The design of a workload generator for an in-production system is theoretically simple, since the users are already present. However, in practice there are some difficulties with regard to data collection, data analysis, development of the model, implementation, and validation. Our experience showed that the main issue concerning GIS is data collection. We could not find public databases with the necessary information and the instrumentation of the client application imposed some serious constraints. Despite of the difficulties, it is still noticeable the lack of works in the literature devoted to this subject, given the fact that Web mapping systems have been widely adopted. There are works that try to simulate a real GIS workload and evaluate its effects on geographical databases [Ray et al. 2011; Simion et al. 2012]. Other works employ the usage history to improve the server cache strategies [García et al. 2012; Quinn and Gahegan 2010]. Romoser et al. [2012] have adopted an methodological approach similar to ours, however to a very different GIS application called USGS EROS, a system for navigation on and downloading of images of the Earth.

Recently, Guan et al. [2014] carried out an investigation with a similar objective: to model the behavior of the users of Web mapping systems for a performance evaluation. Their work provides some valuable theoretical insights about some user characteristics, based on previous works in the literature about general Web workloads and an empirical knowledge of the use of Web mapping systems. However, our work is based on data collected from real users of the Google Maps, and we provide an in-depth analysis of mapping systems operations, such as zoom and pan.

In this article, we present the results of our investigation about the users' behavior when employing Web mapping systems, with an emphasis on Google Maps. We describe how a user interacts with a Web mapping system and the methodology that we have employed to collect and extract this information from the Web browser. Based on the collected data, we have developed a synthetic descriptive model of the user behavior. This model served as the basis for a proof-of-concept workload generator. Finally, we employed this workload generator to carry out a performance evaluation and make a comparison with a conventional benchmark application.

This work is structured as follows. In Section 2, there is a brief description of Web mapping systems in general and some relevant related works are examined. In Section 3, we describe our data collection and analysis. Section 4 describes our descriptive model that is the basis for our workload generator. In Section 5, we present a case study, comparing our workload generator with a traditional benchmark application in the context of a real-world Web tile server. Finally, we provide some final remarks and recommend some future work directions in Section 6.

2. BACKGROUND AND RELATED WORK

Web mapping systems render a map as a set of fixed scales (or zoom levels), dividing the map up into images of the same size, which are called *tiles*. Applications bind a specific resolution to their tiles, *e.g.*, Google Maps employs tiles of 256x256 pixels. The number of tiles grows exponentially as the zoom level increases, following the 4^{zoom} rule, *i.e.*, 1 tile at zoom level 0, 4 tiles at zoom level 1, 16 tiles at zoom level 2, and so on, as is illustrated in Figure 1. The highest zoom level provided by Google Maps is 21, which means that the highest map resolution is $256\text{x}256 \times 4^{21}$ pixels. The tile model that is based on the zoom level has become a standard for Web map systems [Masó et al. 2010]. Among other advantages, this model allows the users to narrow down their requests to the images in the part of the map they are interested in looking at. Naturally, it is important for any research or development that involves Web mapping systems to take account of the tile model. In our work, the tile model is a basic building block for the workload characterizing, modeling, and generating.

Despite the wide adoption of Web mapping systems, there has been a very limited the amount of research on understanding and modeling the real-world users of these systems. In the following, there is a brief description of the key works related to this topic. Romoser et al. [2012] analyzed the logs of the USGS EROS, a system for browsing and downloading Earth images, *i.e.*, an application that is

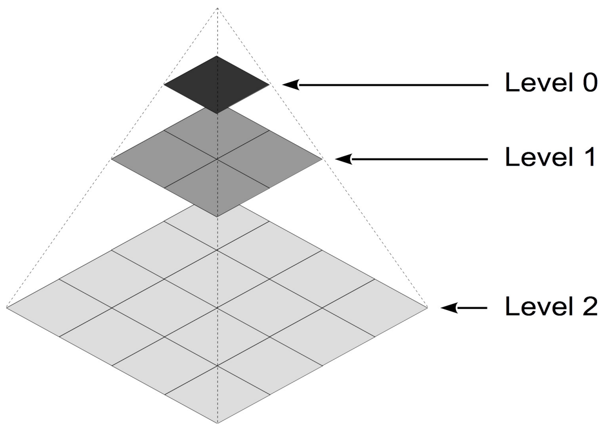

Fig. 1. Illustration of the tile model with three levels.

very different from Google Maps from the perspective of the user behavior. The authors investigated the workload imposed on this system, and described the access behavior according to users, images, and requests. They observed that most of the requests came from a small set of users, and these requests were made to a small set of popular sites, *i.e.*, specific places of the planet. Romoser et al. [2012] found out an access pattern to the images related to large natural disasters, such as earthquakes and tsunamis. These patterns were used to improve caching and prefetching techniques, and thus to improve the system performance.

Kang et al. [2001] proposed an algorithm to perform prefetching of the tiles that are most likely to be requested, based on information from the updated global access pattern. The authors also proposed an algorithm to replace the tiles in the cache based on the same probabilities. However, Kang et al. [2001] did not carry out an investigation about the real data access to the Web mapping systems as a means of assessing the effectiveness of their algorithms. Similarly, other authors have implemented prefetching strategies based on the history of previous accesses, with a view to identifying the most accessed areas, and storing them in a cache [Kefaloukos et al. 2012; García et al. 2012].

Ray et al. [2011] introduced the Jackpine, a tool for benchmarking spatial databases. The authors built models of several common scenarios of access to the spatial databases so that they could mimic real workload. Although they undertook several spatial database operations, the workload generator was developed without taking into account some key factors of GIS access, such as the *think time*. In Web mapping systems, the think time is the time a user takes to analyze the map and perform the next action. In a later work, Simion et al. [2012] employed Jackpine as a concurrent workload generator, to assist in the classification of different microbenchmarks based on CPU and disk usage. However, the workloads were defined without conducting a statistical analysis of the real data, since the authors preferred to set up Jackpine on the basis of their industrial experience and a sort of "common sense". As we will show in Section 5, without an accurate statistical analysis of the real-world demand, workloads can lead to serious mistakes in the performance evaluation of the Web mapping systems.

Zhang et al. [2007] analyzed traces of heterogeneous applications to characterize the workload imposed on the servers. However, the authors only focused on modeling the CPU usage and devising strategies for anomaly detection, and capacity planning. Cibulka [2013] investigated the influence of the latitude, longitude, and scale on the response time of tile-based Web mapping systems. As it could be expected, the results showed that the response time is actually dominated by the density of objects, although the author did not discuss the cache effects. The work also fails to explore other aspects of the user behavior that are needed for a more comprehensive workload model.

Guan et al. [2014] carried out a work that is parallel to ours, and with similar motivating factors. The authors created a theoretical workload based on the behavior of the Web mapping users, called HELP. The information about the users' behavior was estimated from previous findings in the literature about general Web workloads, and the authors supplemented the modeling with their empirical knowledge of the use of Web mapping systems. They proposed statistical distributions for session length, user think time, probability of zoom in/out and pan operations per zoom level. They also set out a technique to find hotspots on the map to decide on the starting point of the navigation. In our work, we employed data collected from users of the Google Maps, and conducted an in-depth deeper analysis of pan and zoom operations. This included a description of the use of the zoom levels and a comparison of the frequency of the most widely used operations. In addition, our work shows the distribution of the pan sizes in pixels and their relation with the zoom level and screen resolution. We also present the relationship between the screen resolution and the number of tiles requested by zoom and pan operations, together with probability distributions of the user think time and the session duration.

3. CONCEPTS AND METHODOLOGY

Geographic Information Systems offer access to information that is associated with a specific position on a map. Both the information and position can be provided to a GIS to help guide the user search. In general, a GIS offers the overlay of different georeferenced data layers, which can be useful for different kinds of analysis and investigation. A very common use of this technology is in the creation of Web mapping systems. These systems have become very popular, since they offer a simple user interface to explore the GIS resources without having to learn the concepts behind them.

Google Maps is one of the most popular Web mapping systems in the world and it shares the use of some traditional operations with other applications in this category, such as zoom and pan. Thus, from the standpoint of a performance evaluation, it is important to know how the users perform these operations, *i.e.*, it is worth characterizing and modeling these operations. This sort of model can be used as the basis for the development of workload generators that can be employed in the evaluation of different Web mapping systems. Since February 2014, Google Maps have provided data in its URLs that make it possible to retrieve information about some operations, including zoom and pan, but also search and routes. This context motivated us to choose Google Maps to perform our data collection and investigation.

In the following subsections, we present some additional details about Google Maps and the information provided by the URLs. We also describe our data collection campaign and our methodology to obtain and analyze the information of interest.

3.1 Data Collection

The Google Maps system provides a Web and a mobile version, but in this investigation our focus is on the Web version. This version updates the URLs as a result of user actions, which means the URLs have a lot of useful information about what the user did and in what sequence the actions occurred. This allows the user actions to be tracked from the set of URLs of a map navigation session. The two most basic pieces of information in the URLs are the geographic coordinate and the zoom level. In the following, the URL structure is described and an example of a URL with both types of information highlighted in bold. The first two numbers represent the latitude and longitude, while and the latter, followed by the letter **z**, is the zoom level. In Google Maps, the zoom may vary from 0 to 21 on the road map.

WEBSITE/maps/@**LAT,LONG,ZOOM**

https://www.google.com/maps/@**37.3075744,-95.4133961,4z**

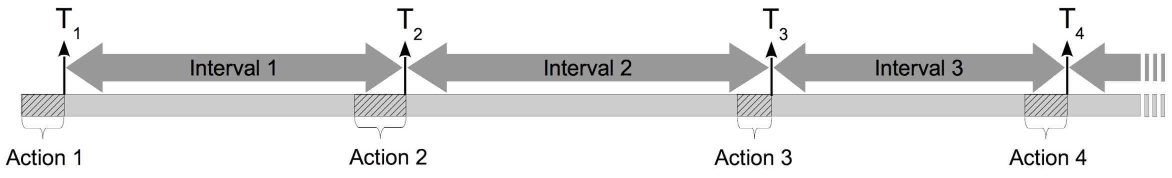

Fig. 2. Procedures of extraction of timestamps and computation of intervals between actions.

In addition to the information about the coordinate and zoom level, it is possible to retrieve information about searches, routes, StreetView visualization, and so on. This information can be easily obtained by employing properly crafted regular expressions. Thus, we have developed a Google Chrome extension to anonymously collect the Google Maps URLs and some mouse movements inside the page, while also recording the respective timestamps. Additionally, the extension collects the page resolution in pixels, which are equivalent to the width and height of the part of the map that appears on the screen. The page resolution is collected for every user action, because the page can be resized during the session. We published the extension in the Chrome Web Store. Nearly 120 users installed the extension, which provided us with more than 60,000 URLs for a period of 5 months.

We defined a user navigation session as the time the user spends on Google Maps, which is measured as follows. The time starts when the user accesses the Google Maps page and ends when he/she finishes the task, either by closing the tab/window or accessing another address in the same tab. During a navigation session in Google Maps, the extension collects anonymous data on user actions. As soon as the user has completed the navigation session, the information is gathered and sent to our server.

3.2 Data Analysis

Initially, we employed the timestamps of the user actions to identify the intervals between consecutive actions. A timestamp indicates when the action has ended, but does not give information about when the action started. We assumed that most of the actions do not diverge significantly from the average and this average tends to be small in comparison with the interval between the actions. In view of this, we created a software tool that iterates through a set of URLs from a user session and computes the difference interval $T_{i+1} - T_i$, where T_{i+1} and T_i are the timestamps of the URL_{i+1} and URL_i, respectively. This interval represents the user think time, *i.e.*, the time the user takes, after completing an action, to analyze the map and then perform the next action. Figure 2 illustrates the extraction procedure for timestamp and the computation of intervals, which are applied to the whole session of every user.

After extracting the timestamps and computing the intervals between the actions, we needed to know what actions were performed during a session and at what zoom level each action was performed. First, the zoom level of each user access on the road map, which is the default map, was extracted and used to estimate the proportion of a session that the user stays in each zoom level. As we will describe in Section 4, this proportion is an important part of the behavior we are investigating. On the hybrid map, *i.e.*, on the map overlaid by satellite images, the zoom level is replaced by information about distance, which was not suitable for the zoom level analysis.

A single URL does not provide enough information to identify the operation performed by the user. Thus, we developed the Algorithm 1 to process every pair of URLs and identify the performed operation. Algorithm 1 receives a pair of consecutive URLs as input and then compares the differences between the URL_i and the URL_{i+1}. On the basis of these differences the algorithm can correctly infer the performed operation. The first URL of a navigation session is a special case, since this URL has no predecessor and so the URL_i parameter is provided with a $NULL$ value. We have designated this initial operation as $BEGIN$. Every conditional statement (lines 4, 6, 8, and 10) checks if the

Algorithm 1 Evaluate the difference between URL_i and URL_{i+1} and identifies the operation.

```
Input: $URL_i$ and $URL_{i+1}$
Output: The $operation$ performed by the user.
 1: if $URL_i = NULL$ then
 2:     $operation \leftarrow BEGIN$;
 3: else if $searchChanged(URL_i, URL_{i+1})$ then
 4:     $operation \leftarrow SEARCH$;
 5: else if $routeChanged(URL_i, URL_{i+1})$ then
 6:     $operation \leftarrow ROUTE$;
 7: else if $zoomChanged(URL_i, URL_{i+1})$ then
 8:     $operation \leftarrow ZOOM$;
 9: else if $coordinateChanged(URL_i, URL_{i+1})$ then
10:     $operation \leftarrow PAN$;
11: else
12:     $operation \leftarrow ANOTHER$;
13: end if
14: return $operation$;
```

consecutive URLs differs in a specific part of the string. For example, if the search information differed from that of the URLs then a $SEARCH$ operation was performed.

As a first example, we will evaluate the consecutive URLs 1 and 2 that follow. In this pair of URLs there was a change in the zoom level, from 4z to 5z. Thus, the difference between the consecutive URLs is defined as a $ZOOM$ operation.

(1) www.google.com/maps/@37.0625,-95.677068,**4z**

(2) www.google.com/maps/@37.0625,-95.677068,**5z**

In fact, it is common for consecutive URLs to have multiple differences from each other. This does not mean the user has performed multiple actions, but that one action has generated multiple differences. We observed that the correct action could be identified if the difference between the two consecutive URLs was evaluated in a specific order. This order is the one employed for the conditional statements in Algorithm 1. For example, in the following two consecutive URLs 3 and 4, there are differences in the zoom level and in the central coordinate. Despite this, the operation is identified as a $ZOOM$, since the difference in the central coordinate is caused by the $ZOOM$ operation. The central coordinate is updated after a zoom, depending on the mouse/pointer position, which does not mean that a pan has been performed.

(3) www.google.com/maps/@**37.0625,-95.677068,4z**

(4) www.google.com/maps/@**38.2971386,-65.7063659,5z**

During the search and route operations, the central coordinate and the zoom level can also be changed. Thus, Algorithm 1 initially checks differences in the parts of the URLs related to the actions of route and search. We provide a final example employing the following three consecutive URLs (5, 6, and 7). When the user searches for a **drugstore**, the URL 5 changes to include the search information, as shown in bold in URL 6. Naturally, the difference between these two URLs is correctly identified as a result of a $SEARCH$ operation. In the sequence, the user performs a pan, and the URL 7 keeps the search information, *i.e.*, there is no difference in this part of the URL. Since the only difference between URL 6 and URL 7 is the central coordinate, Algorithm 1 can properly identify the PAN operation.

(5) .../maps/@37.0625,-95.677068,4z

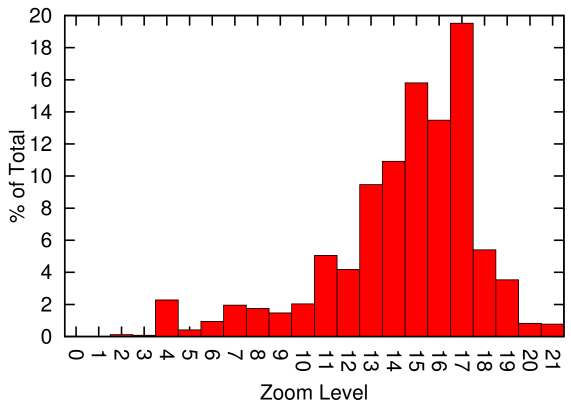

Fig. 3. Choice of the zoom levels in the users sessions.

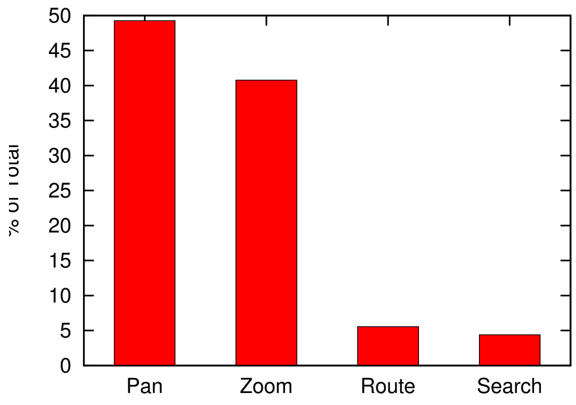

Fig. 4. The four most frequent operations.

(6) .../maps/**search/drugstore**/@37.0625,-95.677068,4z
(7) .../maps/search/drugstore/@**40.2194479,-80.7356618**,4z

With the information about the central coordinate and zoom level, it is possible to compute the equivalent pixel position and the tile position on the map. This is illustrated in an example of the use of the API Google Maps[1]. On the basis of the algorithm introduced in this example, we were able to identify the tiles pattern accessed by a user. By noting the central pixel position and the page resolution, we could also calculate the *bounding box* of the user screen and the amount of tiles requested in each operation.

We also collected the mouse positions during the drag movements so as to be able to compute the sizes of the pans. However, due to inertia effect on the map, this information cannot be used with accuracy. As an alternative solution, we relied on the central geographic coordinate and information about the zoom level of the URLs for the road map to compute the sizes of the pans. In a pan operation, the central geographic coordinate and the zoom level of each URL are used to extract the central pixel positions. Moreover, the difference between these central pixel coordinates of the two consecutive URLs is the pan size.

4. MODELING THE BEHAVIOR OF THE USERS

In this section, we present the main results observed during the data analysis. These results reveal some important aspects of user behavior, which can be used to create user models and synthetic workload generators that properly resemble real-world conditions. As described in [Feitelson 2015], well-designed workload generators are very important tools for the performance evaluation of systems and can also provide valuable inputs for the systems development.

At first, we investigated the user preference concerning the zoom level. Figure 3 shows the histogram of the choice of zoom level during the users sessions, *i.e.*, the zoom level selected by the user while performing every operation. As was expected, the features of human biology mean that some zoom levels are more comfortable for most of the operations performed by most of the users. Close to 50% of all operations are performed in only three zoom levels: 15, 16, and 17. Zoom level 17 has an additional influence; it is the default choice from Google Maps for a search operation if the returned results can be displayed in this zoom level. Additionally, zoom level 17 allows a comfortable navigation with a high Level of Details (LoD). Zoom levels 15 and 16 also have high LoD and allow comfortable navigation. Zoom levels higher than 17 take the user to a very close visualization of the Earth, which

[1]Google Maps API: https://developers.google.com/maps/documentation/javascript/examples/map-coordinates

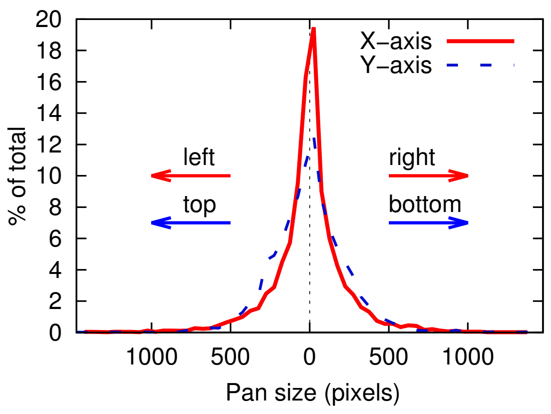

Fig. 5. Distribution of the pan size.

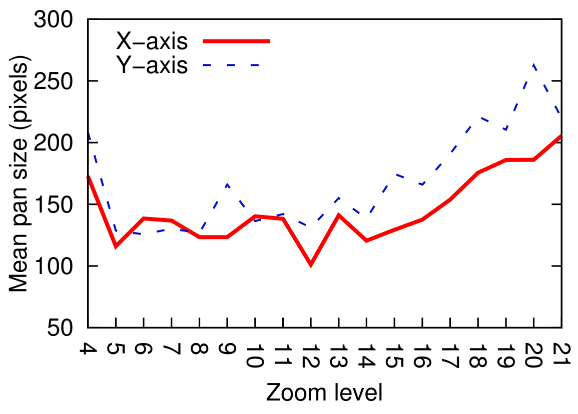

Fig. 6. Average pan size as a function of the zoom level.

makes the navigation generally uncomfortable for human beings. Zoom levels from 14 and below have a low LoD and do not bring any benefits to most of the searches. A similar behavior was found by García et al. [2012] when they analyzed traces of Web map servers.

To understand the Web map user behavior, it is important to know how often this user performs each operation. Thus, we computed the frequencies of use of the four most common operations: pan, zoom, search, and route. A route operation is composed of two or more steps: a starting point, an end point, and any number of intermediate steps. Since each step consumes resources in a Web map server, it was regarded as a route operation. Figure 4 shows the frequency or percentage of these operations in all the evaluated sessions. The operations of pan and zoom are the most frequent, together they correspond to nearly 90% of the users's actions.

The prevalence of pans and zooms in the user behavior led to a deeper investigation of these operations. Starting from the pan operation, Figure 5 shows the distribution of the sizes of the pans in pixels. Movements to the left and to the top of the screen are represented at the left side of the 0 value and movements to the right and to the bottom are represented at the right side. Most of the pans are small, and the percentage of the sizes that are higher than 1,000 pixels is close to zero. Moreover, the users tend to make larger movements on the Y-axis, which seems to be related to the widescreen format. This screen format provides less information on the Y-axis, and so there is a need for larger movements in this direction. The widescreen format is currently the most adopted, as can be seen in the StatCounter Web page[2]. It may not be clear in the figure, but the total number of pans is the same in both axes. The reason for this is that every pan operation is included in both axes even in movements that are only horizontal or only vertical.

Figure 6 shows the average size of the pan for both axes as a function of the zoom level. The movements in the Y-axis have larger average than movements on the X-axis and the size of the pan operations tends to increase as the zoom level grows. In fact, the higher the zoom level, the smaller the relative movement over the Earth that is generated by identical pan operations. In other words, the size of the pan must follow the zoom level so that it can carry out relevant movements. In the period that we collected the data, the default zoom level of Google Maps was 4, which motivates the users to explore this scale and explains the increased average in this level. The zoom levels smaller than 4 were not included in Figure 6 because the data collected does not have representative information about pan operations in these zoom levels. The sum of all the pan operations below the zoom level 4 is only 0.5% of the total.

[2]Top 10 Desktop Screen Resolutions on Feb 2015 | StatCounter Global Stats: http://gs.statcounter.com/#desktop-resolution-ww-monthly-201501-201502-bar

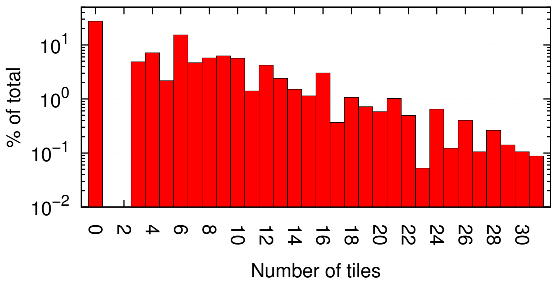

Fig. 7. Number of tiles requested per pan operation.

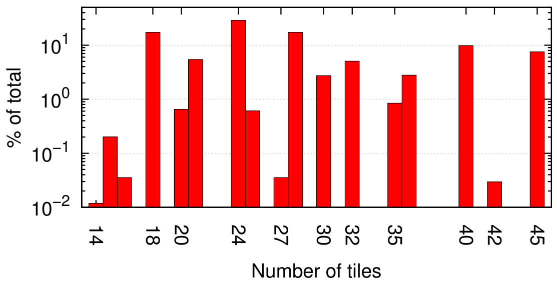

Fig. 8. Number of tiles requested per zoom operation.

We also analyzed the impact of the pan and zoom operations on the number of tiles requested. Figure 7 shows the histogram of the number of tiles requested per pan operation. In most cases, the number of tiles is zero because a large number of small pans only affect parts of the tiles. Since these tiles have already been loaded, even when they do not appear on the screen, there is no new tile transfer. As the screen resolutions support at least three tiles on each axis, no pan generates a request of just 1 or 2 tiles. Currently, the most widely used screen resolution is 1366x768, as can be seen in the StatCounter Web page, which allows a visualization of 6 or 7 tiles on the X-axis and 3 or 4 tiles on the Y-axis. These are among the most common number of requested tiles per pan operation. The other values follow the same reasoning, *i.e.*, the screen resolution influences the distribution of tiles requested per pan operation. As expected, large pans have a low frequency, *e.g.*, pan operations that request more than 16 tiles are rare.

Figure 8 shows the histogram of the tiles requested per zoom operation. Again, the number of tiles is directly proportional to the screen resolution. As can be seen, the most requested numbers of tiles are 18, 24, and 28 and the reason for this is that they are related to the most commonly used screen resolution, 1366x768 px. Depending on how the tiles are arranged, this resolution can fit 6 or 7 tiles on the X-axis and 3 or 4 tiles on the Y-axis, which means the product of each pair corresponds to **18**, 21, **24**, and **28**. 40 and 45; they also have high frequencies for the same reason and correspond to the number of requested tiles for the second most commonly used resolution, which is 1920x1080 px.

In addition to identify the operations, the characterization of the user behavior also involves time issues. Figure 9 shows the PDF of the intervals between the user actions, while Table I shows the average, and some important percentiles of these intervals. Although there are long intervals, some higher than 1 minute, they are not representative. More than 95% of the intervals are shorter than 15 seconds, which is significant information about the think time of the users. Figure 10 shows the CDF of the durations of the user sessions. There are two distributions represented in the figure: one with all the sessions (**All Sessions**), and another in which the sessions only have intervals between consecutive actions smaller than 1 minute (**Engaged Users**). This latter distribution removes the sessions in which the user leaves the Google Maps application open, but embarks on another task. Although this kind of session has a long duration, it does not really have an engaged user. For example, in our collected data, there are some rare samples ($< 0.01\%$) of users that leave the Google Maps open for days. From the perspective of workload generation, this type of session has little interest, since the tile requests become very rare. Operationally, very long sessions can be broken up into smaller ones.

All the operations described in this section are commonly found in any Web mapping system. Although the distributions might be different in other systems, the characteristics remain and the

Table I. Mean and percentiles of the intervals between user actions.

Mean	25th Percentil	50th Percentil (Median)	75th Percentil	95th Percentil
4297 ms	1302 ms	2320 ms	4608 ms	14347.1 ms

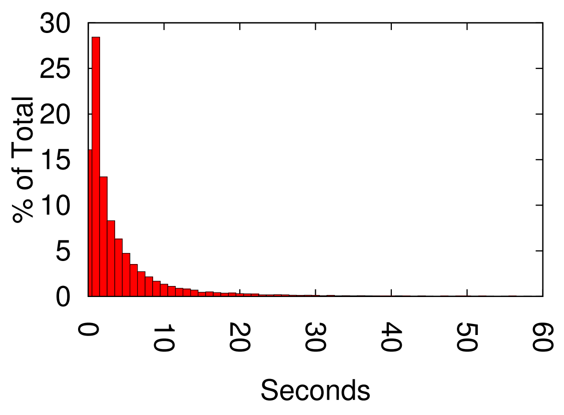

Fig. 9. Time interval between the user actions.

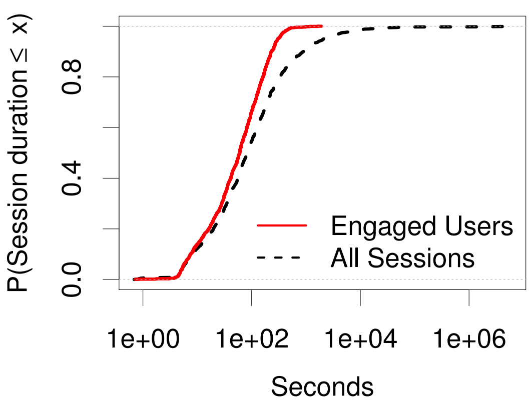

Fig. 10. CDF of the user session duration.

model can be applied to them. Thus, our model can be used as a starting point to create a workload generator and emulate real user features, such as the pan and zoom operations, and think time.

5. CASE STUDY: GOGEO

Stress benchmarks are suitable for comparing the performance between systems and understanding their behavior under saturation. However, they do not give a clear idea of how real users will interact with the system. For example, without a proper workload, developers are unable to determine the number of real users the system can support. Since the real workload is unknown until the system goes into production, a satisfactory strategy is to employ a synthetic workload that is as close as possible to the real one. In Section 4, we described some characteristics of the behavior of Web map users and discussed the importance of these characteristics on the design of the workload generators. In this section, we take one of the investigated characteristics – the intervals between actions or think time – to develop a proof-of-concept workload generator. By taking into account the think time, our workload better represents the frequency or the rate at which the users send requests to the server.

We employed the goGeo[3] infrastructure, including its Web tile server, to execute the tests. This illustrates how the data from a production system (Google Maps) can be used to evaluate a system that is still under development (goGeo). All the tests involved the same set of computers and a 151.3 MB database with more than 500,000 companies in Brazil. The set of requested tiles covers the entire territory of Brazil. The caching functionality was disabled in the server, which meant that all the requests required a new tile rendering. Each test ran for 10 minutes and the response time of each request was measured.

Initially, we executed a stress workload based on the MapLarge performance tests[4]. We generated 25 synthetic Web map clients to send requests for random tiles. The stress workload sends these requests as quickly as possible. The average response time of this workload was 105 ms. Following this, we created multiple workloads with the users' think time, starting with 25 clients and increasing this number until it reached a similar average response time as the stress workload. We used an empirical distribution, based on the distribution shown in Figure 9, to simulate the users' think time. With 800 clients the average response time was even smaller (71 ms), but with 1,000 clients it was larger (158 ms). This means that with a similar average response time, the system can support between 800 and 1,000 clients when the intervals between the requests are taken into account.

[3]goGeo – High Performance Maps Platform: http://www.gogeo.io
[4]MapLarge – Map Server Performance: http://maplarge.com/mapserverperformance

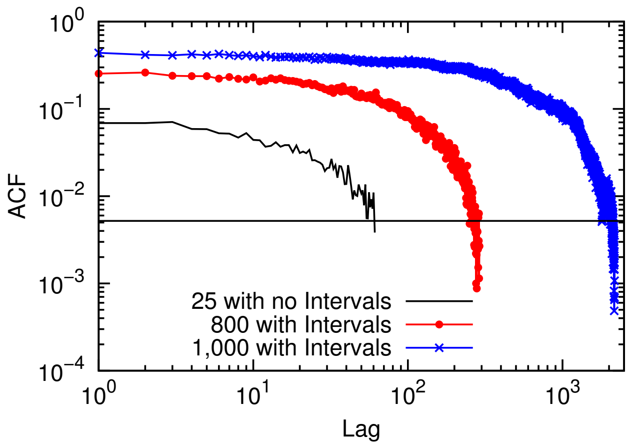

Fig. 11. Autocorrelation function (ACF).

As it would be expected, the two workloads present other differences in their statistical properties. For example, our workload has a higher autocorrelation in the response time function, as illustrated in Figure 11. The straight line parallel to the X-axis is the confidence bound at a significance level of 0.05. While the stress workload shows autocorrelation until a time lag of 6×10^1, the user-based workload reaches a lag of 2.5×10^2 and 1.8×10^3 for 800 and 1,000 clients, respectively. This means that, for a similar average response time, the time lag of the user-based workload is about one order of magnitude greater than the lag of the stress workload. Naturally, this difference was created only by the introduction of the think time, which suggests that other user characteristics could have a similar effect on the statistical properties of the workload.

In summary, although our proof-of-concept user-based workload generator does not represent many of the characteristics of real users, it was able to corroborate our initial hypothesis about the importance of adding real-user features. We confirmed this through the significant impact on the performance evaluation of a real-world GIS system.

6. CONCLUSION AND FUTURE WORK

The use of geographic information systems has increased in recent years, but there is still a lack of models for describing user behavior in this kind of application. This deficiency constrains the performance evaluation and capacity estimation of the geographic information systems. Without a suitable workload, developers may devote too much time to features of the system that are rarely used, while critical components are overlooked. In this article, we presented a descriptive model of the behavior of Web map users, which was developed on the basis of real-world data. We also implemented a workload generator based on the think time of the users and compared this with a stress benchmark. The results showed that adding user characteristics can significantly change the effective workload imposed on the server, and for example, lead to a different conclusion about the saturation of the system.

In future work, we intend to create a generative model that is able to comprehensively represent the behavior of the Web map user during a navigation session, including the relationships between the actions at each zoom level. This improved model will allow a more accurate workload generator to be developed. The investigation of the behavior of GIS users employing mobile devices is another key subject of research. There are important challenges related to the data collection in this context, such as capturing the user actions, saving the data locally and recovering the data without disturbing the user. In addition, the modeling may require the investigation of other factors about the user.

REFERENCES

Cibulka, D. Performance Testing of Web Map Services in Three Dimensions–X, Y, Scale. *Slovak Journal of Civil Engineering* 21 (1): 31–36, 2013.

Feitelson, D. G. Workload Modeling for Computer Systems Performance Evaluation – Version 1.0.3. http://www.cs.huji.ac.il/ feit/wlmod/, 2015.

Fiedler, M., Hossfeld, T., and Tran-Gia, P. A Generic Quantitative Relationship Between Quality of Experience and Quality of Service. *Network, IEEE* 24 (2): 36–41, 2010.

García, R., de Castro, J. P., Verdú, E., Verdú, M. J., and Regueras, L. M. Web Map Tile Services for Spatial Data Infrastructures: management and optimization. In C. Bateira (Ed.), *Cartography - A Tool for Spatial Analysis*. InTech, pp. 25–48, 2012.

Guan, X., Cheng, B., Song, A., and Wu, H. Modeling Users' Behavior for Testing the Performance of a Web Map Tile Service. *Transactions in GIS* vol. 18, pp. 109–125, 2014.

Kang, Y.-K., Kim, K.-C., and Kim, Y.-S. Probability-Based Tile Pre-fetching and Cache Replacement Algorithms for Web Geographical Information Systems. In *Proceedings of the 5th East European Conference on Advances in Databases and Information Systems*. London, UK, pp. 127–140, 2001.

Kefaloukos, P. K., Salles, M. V., and Zachariasen, M. TileHeat: a framework for tile selection. In *Proceedings of the 20th International Conference on Advances in Geographic Information Systems*. New York, NY, USA, pp. 349–358, 2012.

Masó, J., Pomakis, K., and Julià, N. OpenGIS Web Map Tile Service Implementation Standard – Version 1.0.0. `http://www.opengeospatial.org/standards/wmts`, 2010.

Quinn, S. and Gahegan, M. A Predictive Model for Frequently Viewed Tiles in a Web Map. *Transactions in GIS* 14 (2): 193–216, 2010.

Ray, S., Simion, B., and Brown, A. D. Jackpine: a benchmark to evaluate spatial database performance. In *Proceedings of the 2011 IEEE 27th International Conference on Data Engineering*. Washington, DC, USA, pp. 1139–1150, 2011.

Romoser, B., Fares, R., Janovics, P., Ruan, X., Qin, X., and Zong, Z. Global Workload Characterization of a Large Scale Satellite Image Distribution System. In *Proceedings of the 31st International Performance Computing and Communications Conference (IPCCC)*. Austin, TX, EUA, pp. 368–375, 2012.

Simion, B., Ray, S., and Brown, A. D. Surveying the Landscape: an in-depth analysis of spatial database workloads. In *Proceedings of the 20th International Conference on Advances in Geographic Information Systems*. New York, NY, USA, pp. 376–385, 2012.

Zhang, Q., Cherkasova, L., Mathews, G., Greene, W., and Smirni, E. R-Capriccio: a capacity planning and anomaly detection tool for enterprise services with live workloads. In *Proceedings of the ACM/IFIP/USENIX 2007 International Conference on Middleware*. New York, NY, USA, pp. 244–265, 2007.