Webalyt: Open Web Aalytics Platform

Lukáš Čegan[1], Petr Filip[2]
Faculty of Electrical Engineering and Informatics
University of Pardubice
Pardubice, Czech Republic
[1]lukas.cegan@upce.cz
[2]petr.filip1@upce.cz

***Abstract*—Web analytics tools offer important support for better recognition of the web user's behavior, identify bottlenecks and errors in user interface design, measure performance of web environment, monitor website availability or recommend appropriate website content. These tools are based on tracking techniques and sophisticated algorithms that process and evaluate large volumes of captured data. In this paper, we propose a new open web analytics platform called Webalyt that is designed with regard to ensuring availability and reliability under high traffic. A presented platform is formed by interconnected components, which apart from providing the basic functionality of tracking systems, also features advanced functionality for processing real-time data which is needed for recommendation systems. Furthermore, the platform has a scanning robot to search for information about users of third-party applications and marketplace for data interchange between trading partners. The platform is an open source with documented APIs for the development of expansion modules.**

Keywords—web analytics; website; monitoring; user behavior; error handling;

I. INTRODUCTION

The rapid pace of innovation cycles of information technology in recent years led to increased availability of Web services in the world. According to available statistics, nearly 3.5 billion people were connected to the Internet at the end of 2016 [1]. That is more than 40% of world population. This large number of users represents a huge business market. For successful enforcement on this global market (regardless of the fact whether it is an e-commerce application, social network or blog) it is necessary to be able to clearly understand web users. For this purpose, it is possible to use tracking tools in a digital world that can capture data expressing the user behavior and data representing the user performance environment. In addition, is also possible use any of the tools that can evaluate large amounts of data and present the results in an appropriate form.

The subjects of current scientific interest in this area are studies mainly focused on solving isolated problems such as segmentation of users in order to better sales strategies [2], or behavior profiling of users for user identification [3]. Further, many approaches in literature are aimed at improving performance of websites based on intelligent preloading of website resources [4] or based on predicting users' future movements using the Markov model [5][6]. Additional published papers are focused on customizing a website to the needs of specific users [7][8], developing new distributed recommender system supporting device adaptivity [9], developing agent-based personalized recommendation method [10] and a personalizing products based on a recommendation system [11]. All these papers are built on the same base; on the ability to capture and evaluate web users' behavior and web environment data.

In this paper we present a new innovation open platform for web analytics. The platform is called Webalyt (an acronym of WEB anALYTics). The platform is composed of six interconnected components, which are able to measure, collect, transform, analyze and report web data. A decomposition platform into components, allows better control over performance, security, availability, and its own development.

The paper is organized as follows. After introducing the objective of this paper, the web analytics field are presented in Section II. Section III described the proposal solution of Webalyt, new open web analytics platform. The developing of extension modules for Webalyt platform is described in Section IV. In Section V proposed solution is described. Finally, the last section gives conclusions and future research opportunities followed by references at the end.

II. WEB ANALYTICS OVERVIEW

Web analytics is a scientific field that encompasses many sub-disciplines such as measuring web traffic, collecting big data, analysing web performance, reporting processed outputs, business data mining, data visualization strategies, etc., There are a number of tools within these disciplines that greatly facilitate our work with captured data. However, for a proper grasp of these data, individual tools are not as important as where they come from and under what circumstances they were acquired.

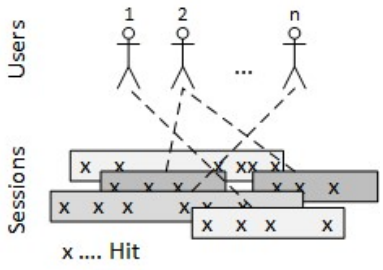

Fig. 1. Users, sessions and hits

The Fig 1. illustrated the basic data classification. A core group of data that is called Hit. Hit is any individual user's interaction with the page. For example, hit is page view, click the link, cursor movement, filling in the form, custom events, etc. A session is a set of Hits that have been captured during one user's continuous work with the website. Session identification is assured through a unique ID that is generated for each session by tracking system and it is added to each hit. Limitation of user tracking in a single session is the impossibility of forwarding session ID among the various websites that the user visits. This limitation is due to the safety rules of web browsers. For example, if a user views a website dedicated to a destination of his business trip and at the same time he views the weather forecasts on another website, from the perspective of tracking system they are two separate sessions. Each session belongs to a particular user. Decision of which session belongs to which user is not an easy task, because the HTTP protocol is stateless. The solution to this problem is the analysis of selected attributes of the captured data such as the IP address, digital fingerprint, or user identity. However, the results are not always entirely convincing, because, for example, the IP address can be shared or a digital fingerprint may not be strong enough.

Web analytics tools currently available on the market generally provide information on what volume of visitors are coming to our site, from which countries, what time of day they coming, on what content they spend most of their time, etc. Some of these tools offer advanced features including the implementation of test scenarios, A / B testing, error detection and evaluation of conversion. The most widely used web analytics tool is a product called Google Analytics. This tool is used to track traffic for more than 50% of all websites [12]. Google Analytics allows website owners to obtain statistical data about their users; like current and historic traffic, user behavior and their properties, sales conversions, etc. The tool is primarily focused on the evaluation in terms of sales and marketing. To lesser extent, it is applicable to analyzing errors in the design of the user interface and user experience. Google Analytics is available in two versions: free and paid. Limitations on the free version are mainly in the number of requests that the server processes in a certain time. An alternative to Google Analytics is Piwik. It is a free open-source analytics tool written in PHP. The advantage of the tool Piwik is able to achieve full control over the collected data. The terms of use does not require sharing data with anyone (including Google). The owner can control the privacy. The advantage is also the ability to customize and extend the tool by plugin. On the other hand, the disadvantage is less features and less user-friendliness. Besides the mentioned tools above, there are other commercial tools, which offer similar functionality: Mouseflow, SessionCam, UsabilityTools, Crazy Egg or Hotjar.

All tracking tools operate using JavaScript code that makes it very easy to block them by disabling JavaScript in the browser (this also disabled a variety of useful functions based on JavaScript), or by browser's plugin, which prevents sending tracking hits. From the user perspective blocking of tracking requests has advantages and disadvantages. The advantage is the privacy protection and reduction of the data flow (bandwidth savings). The disadvantage is affecting the correct functioning of certain systems that are dependent upon tracking, such as automatic filtration of large quantities of media content and pre-selection of only relevant content for the user (commercial product close to the user's interest).

III. WEBALYT – PROPOSAL SOLUTION

Analysis tools provide a range of functions through which it is possible to capture and analyze data as mentioned in Section 2. However, these tools allow analysts to work only with a limited set of hypotheses, because the breadth of statistical data is determined by internal limits of each application. In this paper a new open web analytics platform is presented. The platform has proposed new unconventional architecture that allows processing data from various sources, segmenting users in real time and extending the provided functionality through additional modules.

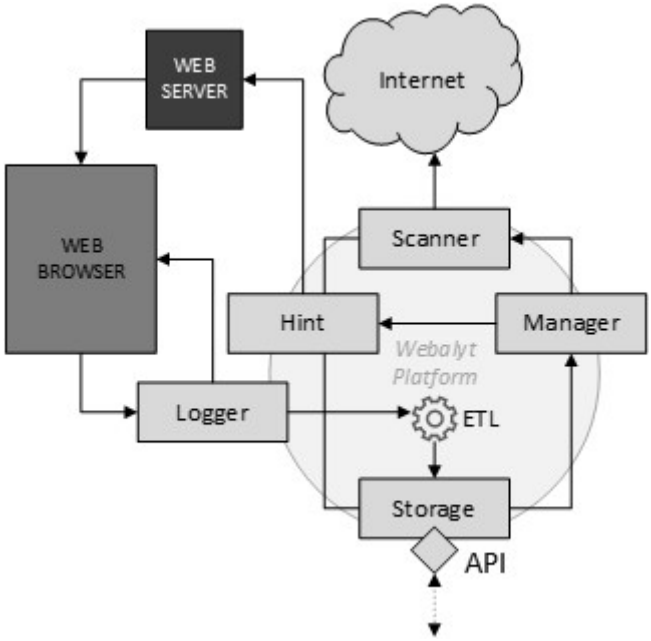

Fig. 2. Webalyt platform architecture

Fig. 2 shows a proposed architecture. Each component of the platform is described in the following section.

A. Webalyt Logger

Logger is the only component of the Webalyst platform, which a client's web browser directly communicates with. The rest of Webalyst component communicate only with each other or with the web server providing the web page to user. Communication among Web browser, Web server and *Webalyst logger* is showed in Fig. 3 using UML sequence diagram.

As is on the figure shown, communication starts with a user's web browser, which sends a HTTP request to the web server. The web server processes a request and then sends a response (HTML page). Next, the web browser parses the source code of the HTML page, and it starts loading the external resources, which blocks the rendering. One of these resources is a tracking script of Webalyt platform *webalyt.js* file. After the web browser receives the script, execution begins, which invokes loading another module scripts form Webalyt logger. A list of scripts is defined in the main configuration file *webalyt.json* that is located in the web server. Additionally, the main configuration file also contains parameters governing settings of each executed module. For each module are mandatory parameters: name, prefix, device. Optional parameters are given in params.

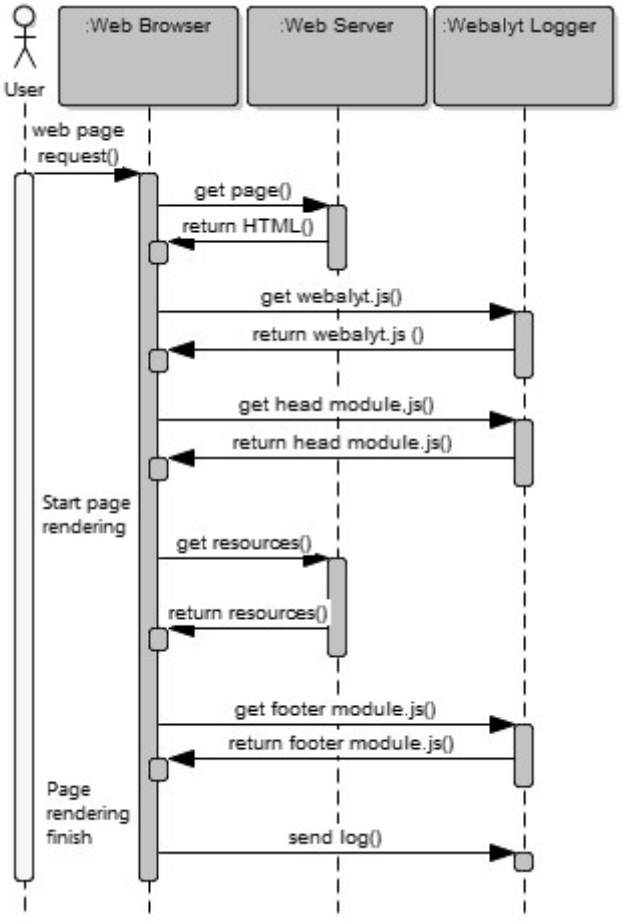

Fig. 3. Logger sequence diagram

The following Listing 1 provides an example configuration module "Link touch error", which is used to identify poorly targeted links on mobile devices. Optional parameters of this module are: count – the number of incorrectly focused link in a row; time – the period for which they are incorrectly targeted links; container – the size of the surrounding area of links; selectors – elements for implementing the measurement; pages - pages of website on which measurement is performed.

```
{
   "modules": [{
      "name": "link_touch_error",
      "prefix": "lte",
      "device": [
         "mobile", "tablet"
       ],
      "params": {
         "count": "3",
         "time": "5",
         "container": "15px",
         "selectors": [
            "a.btn","button.btn","input[type=submit]"
         ],
         "pages": "*"
      }
   }]
}
```

Listing 1. Example of module configuration in webalyt.json file

Data that are captured are sent from the web browser to Logger by AJAX on the background. It is possible to use GET and POST http methods. Despite the fact that the http itself does not impose any hard-coded limit on GET request length, it is necessary to reflect that browsers have limits, in the range 2kb - 8kb (depends on the browser type). The next Listing 2 shows the log format.

```
?prefix_p1=v1&prefix_p2=v2&prefix_p3=v3&…
```

Listing 2. Source code of log format

Log string contains variables and their values. The variable name has a prefix that identifies the module to avoid confusion variables with the same name between modules. The following Listing 3 shows a concrete example of a log chain module "Link touch error". It is a poorly targeted link with ID = button_ok on the page "shop/card.php".

```
?lte_id=button_ok&lte_page=shop/card.php
```

Listing 3. Source code of log format

Allowed MIME type of a log string is *application/x-www-form-urlencoded* for GET and POST requests. Moreover to the POST request it is possible to use a MIME type *text/json*. JSON format is suitable for use in the event that it is necessary to send large volumes of structured data. For example, measuring the performance of a web page by navigation timing API.

B. Webalyt ETL

Webalyt ETL is an extraction, transformation and loading component. Architecturally it is divided into two parts. The first part of ETL, extracts data from the priority queue, which is designed for real-time data processing. These type of data are processed immediately when there are created and then they are loaded to the data storage for post processing. Real-time data processing are necessary for real-time feedback component (*Webalyt hint*). The second part of ETL extracts data from the normal queue. This queue is not designed for real-time processing, the velocity and the quantity of the extracted data is dependent on the current load on the Logger server.

C. Webalyt Storage

Webalyt storage is a component, providing storage and retrieved of captured data. As a data repository, relational databases (Oracle, MySQL, PostgreSQL, MS SQL, etc.) and NoSQL databases (Hadoop) can be used. For developers appropriate DB drivers are ready. Moreover, it is possible to use cloud services like Amazon S3 as a data repository. The choice of database engine depends on the amount of data we capture and the required availability and reliability.

D. Webalyt Manager

Webalyt Manager is a component that falls within the category of business analytic tools. The purpose of this component is analyze data and extract only relevant information, thereby making it is possible to increase results, performance or user satisfaction. This component provides a set of basic visual elements (table, graph, etc.), which can be used for data visualization.

E. Webalyt Hint

Webalyt hint is the cornerstone component for building personalization applications. This component processes the captured data in real time and the results are forwarded to the web server. The purpose can be selecting the appropriate content relevant to the type of user, or dynamic adjustment of page layout. An example might be the module "Link touch error". If *Webalyt hint* detects whether a user has a problem with focusing buttons, it sends a message to a Web server, which increases the size of the buttons.

F. Webalyt Scanner

Webalyt scanner is a component, which provides tracing information from third-party applications. Architecture of this component is inspired by a classic web crawler. Basically, it is a small web bot that searches information according to a predetermined script. This data is highly advantageous for improving results in the segmentation of users.

IV. Module Development

The strength of the Webalyt platform is the ability to easily extend the functionality through additional modules. In the current commercial system, this is not possible, or only in a limited way. In this section a module development is described.

The core of the platform is created in PHP based on the Skeleton application for Zend Framework 3 (ZF3). Thus, the structure of each module is based on the defined code rules. Each module has its own directory that contains the manifest file *module.php* and these key folders:

- config – module specific configuration files,
- language – translation files,
- src – source files with module application logic,
- view – view scripts for presentation layer,
- public – images, javascript and css files.

Src directory includes seven directories that divides the application logic for each component of Webalyt platform: Core, Logger, ETL, Storage, Hint, Scanner and Manager. The module may not contain all the directories except Core, Manager and Logger. These three components are required for each module. Distribution of the source code to these directories is because of the flexible deployment process. Webalyt platform in fact, can run on a single server (such as Piwik) or be deployed on multiple servers due to achieving optimal performance.

Src/Core directory contains shared source codes among all components of a module. The purpose of the rest of the directories is determined by their name. The component Manager is built using the MVC design pattern. For this reason, the *src/Manager* directory contains a Controller directory, which contains each controller with activities.

Public directory is used for storing a module's static assets like JavaScript files, CSS files, images, etc. However, this type of sources are in FZ3 skeleton stored in the public directory with a central access control. In order to develop high quality and well tested modules Webalyt has inside its core a plugin AssetLoader, which solves the problem of delivery assets with modules themselves. Plugin is built on the idea that every request which cannot be routed to the controller is redirected to a */public* directory and an asset is eventually served from there.

The deployment of Webalyt can be done in manual ways or by using the command line tool WebalytDeploy. This tool is located in */bin* directory and it helps administrators to deploy each component of the Webalyt platform to individual servers. The configuration of the deployment process is determined in */config/deployment.config.php* file. WebalytDeploy supports a two steps deployment process: staging and production environment. Staging environment is designed for verification that the application works as intended. After the application successfully passed all tests in the staging environment, it can be released into the production environment.

V. Performance Evaluation

The presented Webalyt platform was compared with an open source analytic tool Piwik. Comparisons were performed using the Apache JMeter 3.1. The experiment focused on the evaluation of tracing the script of each tool, which processes the hints on the server. The measurement of each script consisted of 50 concurrent users (threads) that called the script in the cycle for 60 seconds and the ramp-up period was 5 seconds. The query string of each request contained 28 parameters. In each measurement median time of server response, allocated memory and CPU load were monitored. Both measurements were performed on the same hardware and the same web server configuration.

Fig. 4 and Fig. 5 show the results graph of Piwik and Webayt tracking script respectively. Each line of graph represents the current average of all samples (blue), the median (purple), the current standard deviation (red), and the current throughput rate (green) in milliseconds. The median of all samples of Piwik was 2793 milliseconds. During the measurement, the web server has allocated 475 MB of memory and the processor was busy at 80-100%.

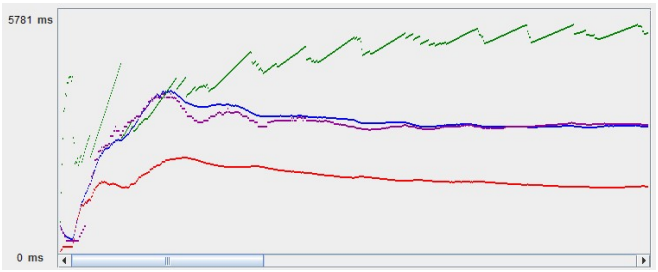

Fig. 4. Results graph of measurement of Piwik tracker script

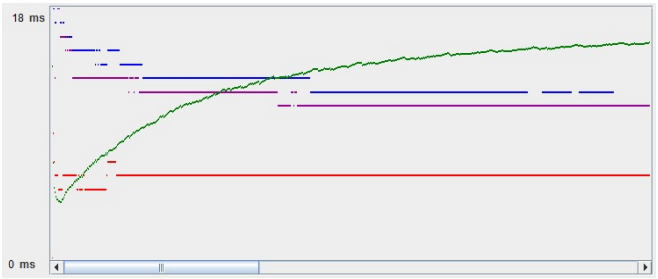

Fig. 5. Results graph of measurement of Webalyt logger script

The median of all samples of Webalyt was 9 milliseconds. During the measurement, the Web server has allocated 187 MB of memory and the processor was busy at 5-8%. This dramatic difference in the performance of scripts is given by way of their processing. The Piwik tool has implemented a FrontController design pattern, which means that all HTTP requests are directed to one input point and then the request is processed by the routing rules. Further, the tracking data are processed immediately. The all of this significantly increases processing time of the script. Platform Webalyt has strictly separated data logging and data processing mechanisms. Data processing occurs subsequently, where the actual server load is not high, which allows optimal usage of system resources. In case it is necessary to process tracking data in real time, then the server load is proportional to the number of parameters that need to be processed and the number of Webalyt modules that are deployed. Comparison with other web analytical tools like Google analytics, Adobe analytics, Hotjar, etc. is not possible, because the rest of the tools have a proprietary source code and it is not possible to carry out comparative performance tests.

VI. Conclusion

This paper presents a new open web analytics platform Webalyt from an architectural perspective. This platform is designed with regard to ensuring the availability and reliability under high traffic. The entire platform is split into six components: Logger, ETL, Storage, Scanner, Hint and Manager. These components provide the basic functionality of tracking systems, storing and processing big data (including real-time processing as a part or advance recommendation systems), obtaining information from third-party applications, a reporting system and a variety of other functionality for work with the captured data. This paper also presents development of an additional module, through which it is possible to expand the basic functionality of the Webalyt platform. Currently, the Webalyt platform is in the alpha phase of the release life cycle and therefore it is not yet a published source.

Acknowledgment

This work is published thanks to the financial support Faculty of Electrical Engineering and Informatics, University of Pardubice under grant SGS_2017_025 “Active monitoring of web users’ behavior”.

References

[1] Internet Live Stats (2017, Jan. 22) *Internet Users* [Online]. Available: http://www.internetlivestats.com/internet-users/

[2] S. Tonkin, C. Whitmore and J. Cutroni. *Performance marketing with Google Analytics: Strategies and techniques for maximizing online ROI.* John Wiley and Sons, 2011.

[3] Y. Yang, “Web user behavioral profiling for user identification” in *Decision Support Systems*. 2010, pp. 261-271.

[4] L. Cegan, “Intelligent Preloading of Websites Resources Based on Clustering Web User Sessions” in *IT Convergence and Security*, 2015, pp. 1- 4.

[5] X. Dongshan and S. Juni, “A New Markov Model for Web Access Prediction” in *IEEE Computing in Science and Eng.,* vol. 4, 2002, pp. 34-39.

[6] M. Deshpande and G. Karypis, “Selective Markov Models for Predicting Web Page Accesses” in *ACM Trans. Internet Technology,* vol. 4, 2004, pp 163-184.

[7] M. Eirinaki and M. Vazirgiannis, "Web Mining for Web Personalization" in *ACM Transactions on Internet Technology,* vol. 3, *No. 1,* 2003, pp. 1-27.

[8] R. Baraglia and F. Silvestri, "Dynamic personalization of web sites without user intervention" in *Communication of the ACM*, vol. 50, 2007, pp. 63-67.

[9] D. Rosaci, G. M. L. Sarné and S. Garruzzo, “MUADDIB: A distributed recommender system supporting device adaptivity” in *ACM Transactions on Information Systems*, 2009, pp. 1-41.

[10] T. Cheng, W. Han, H. Wang, Y. Zhou, B. Xu and B. Zang, “Content Recommendation System Based on Private Dynamic User Profile” in *International Conference on Machine Learning and Cybernetics*, 2007, pp. 2112-2118.

[11] S. Y. Sneha, G. Mahadevan, M. Prakash, “A Personalized Product Based Recommendation System Using Web Usage Mining and Semantic Web” in *International Journal of Computer Theory and Engineering*; 2012, pp. 202-205.

[12] W3Techs (2017, Jan. 22) *Usage of traffic analysis tools for websites*. [Online]. Available: https://w3techs.com/toverview/traffic_analysis/all