Implemention On Web Analytics Process-Measurement And Optimisation

M D Naushad[1], Rakshith Sridharan[2], Raj Pratap[3], Shashikant S Singh[4] ,Rajeswari S[5]
[1, 2, 3, 4, 5] Dept of Information Science
[1, 2, 3, 4, 5] New Horizon College of Engineering, Karnataka, Bangalore, India

Abstract- *This paper we present a system that supports the design of web graphical user interface by finding the optimal placement of interactive elements. The definition of optimal placement is context specific; it aims at maximizing measurable aspects of the user experience, and it is derived using expert knowledge embedded in the system, which is based on HCI principles, user studies, and data analytics. We use a novel modeling technique to represent the layout of web user interfaces.*

Keywords- Web User Interfaces, Data Analytics, Expert Knowledge, Optimal placement

I. INTRODUCTION

Providing user-friendly systems is one of the key objectives for designers of interactive computer-based systems. As technology advances, novel interaction modalities emerge, providing new engagement opportunities as well as new challenges in the design process. Previous works on Human-Computer Interaction (HCI) have demonstrated that it is possible to influence the behavior of humans using computer systems. In the past, extensive user studies had to be carried out to collect sufficient data to identify effective strategies for influencing the users. Today, most personal devices are connected to the Internet, and user activity logs are often collected with anonymous profiles. Large Internet companies collect a significant amount of information and appropriate large data analytics can identify specific users' preferences and patterns. This enabled the definition of context specific HCI design guidelines to improve system usability. The application of this expert knowledge has opened opportunities in a wide spectrum of fields. In this paper, we focus on web interfaces because of their popularity, universal accessibility, and nearly ubiquitous presence.

II. PROPOSED SYSTEM

We propose a system to automate the embedding of expert knowledge in the design of website layout. We focus on web interfaces because of their popularity, universal accessibility, and nearly ubiquitous presence. Identifies the type of website. Builds a model of the website layout. Compares the computed model against optimal reference models, providing a quantitative evaluation for the current design. And provides a recommendation to improve the UI based on the expert rules

A. Website Capture Module:

Early The website capture module is the entry point to the system. It takes a website URL, and it fetches the HTML code of the website using an HTTP request. The code is then passed to the website classification module. We use a web driver (such as Selenium web driver) to render the HTML and obtain an image of the webpage. Then the web driver produces a screenshot of the webpage with a parametric geometry and passes it on to the following image processing module.

B. Web Classification Module:

The classification algorithm developed in this system is based on the bag-of words approach. This approach was chosen because it provides high accuracy with text based classification. Before applying the bag-of-words approach, the HTML text is cleaned from tags, scripts, and styles. It is then passed to a natural language processing toolkit to remove simple words such as "and", "or", "with", "is". These words are known as stop-words and are irrelevant for classification purposes. The bag-of-words approach generates a histogram of frequencies for each keyword. Based on this histogram, a score is given to each category based on the number of words assigned to each category.These categories were also chosen because the vast majority of e-commerce websites fall into one of these

C. Image Processing Module:

This component of the system processes the screenshot of the website (checkout page in particular) captured in the previous stage. At first, the algorithm segments the image using Felzenszwalb's algorithm. Then, each segment is analyzed separately to identify whether it contains any UI relevant element. Pattern matching is used to identify elements of unique layout, such as the Visa Checkout button which is used for validation. The matching process is based on SIFT to find key-points and on Random Sample Consensus (RANSAC), which uses Homography to find the most likely correct key-points. The module reiterates over each segment and tests the different templates. To pass the test, templates need at least 10 matched key-points and a statistically sufficient number of good matched key-points defined by Lowes ration test.

D.Model Generation Module:

After receiving the UI image elements from the image processing, this module generates a graph based model that describes the relation between the image elements. The model can be described as a set of nodes connected by links. Each node in the model will describe an element in the UI. The node structure will be as such:

- Node Type: (Checkout or Continue or Image)(text)
- Node ID: (number)
- Node Coordinates: (x, y) (number, number)
- Node Boundaries: (up, down, left, right)
- (number, number, number, number)
- Node Dimensions: (h, w) (number, number)
- Node Text: (text)
- Node colour: (number)

E. Scoring and Evaluation Module:

After computing the nodes and links of the model, the evaluation module processes it according to the website type and according to the provided expert rules. These rules are parsed according to the follow syntax:

- Colour intensity of X with respect to Y
- Location of X relative to Y (above, below, right of, left of)
- Text of X contains "text"
- Distance between X and Y
- X alignment with respect to Y (center align, right align, left align)
- Size of X with respect to Y (same, larger, smaller)

F. **Felzenszwalb's Algorithm***:*

The input is a graph $G = (V,E)$, with n vertices and m edges. The output is a segmentation of V into components $S = (C_1 \ldots C_r)$.

1. Sort E into $\pi = (o_1 \ldots o_m)$, by non-decreasing edge weight.
2. Start with a segmentation S^0, where each vertex v_i is in its own component.
3. Repeat step 3 for $q = 1 \ldots m$.
4. Construct S^q given S^{q-1}as follows. Let v_i and v_j denote the vertices connected by the q-th edge in the ordering, i.e., $o_q = (v_i,v_j)$. If v_i and v_j are in disjoint components of S^{q-1} and $w(o_q)$ is small compared to the internal difference of both those components, then merge the two components otherwise do nothing.

More formally, let C^{q-1}_i be the component of S^{q-1}containing v_i and C_i^{q-1} the component containing v_j . If $C_i^{q-1} = C_j^{q-1}$ and $w(o_q) \leq MInt(C_i^{q-1},C_j^{q-1})$ the nS^q is obtained from S^{q-1} by merging C_i^{q-1} and C_j^{q-1} . Otherwise $S^q = S^{q-1}$.

5. Return $S = S^m$.

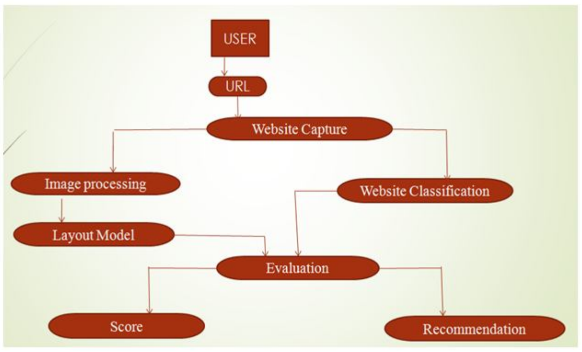
III. COMPRESSION TECHNOLOGY

Web page for the first time the load is in the browser without the cache from the server through the network to download the process of the client. There are many factors that affect the speed of the page, the page size is an important factor affecting the loading speed of the page. When the requested page is small, the amount of data is small and the time it takes to return is reduced. Compression is the simplest technology to reduce page size.

A. Gzip compression:

Gzip is an abbreviation for GNUzip, and the Gzip encoding on the HTTP protocol is a technique used to improve the performance of Web applications. Gzip compression technology is currently the most widely used a compression technology, it is a GUN free software file compression program. Gzip is now the most widely used a compression technology, patent rights and other aspects without any restrictions, and has become the RFC standard.

IV. DATA FLOW DIAGRAM



V. IMPLEMENTATION

The implementation phase involves the actual materialization of the ideas, which are expressed in the analysis document and developed in the design phase. Implementation should be perfect mapping of the design document in a suitable programming language in order to achieve the necessary final product. Often the product is ruined due to incorrect programming language chosen for implementation or unsuitable method of programming. It is better for the coding phase to be directly linked to the design phase in the sense if the design is in terms of object oriented terms then implementation should be preferably carried out in a object oriented way.

The implementation involves:

2. Careful planning.
3. Investigation of the current system and the constraints on implementation.
4. Training of staff in the newly developed system.

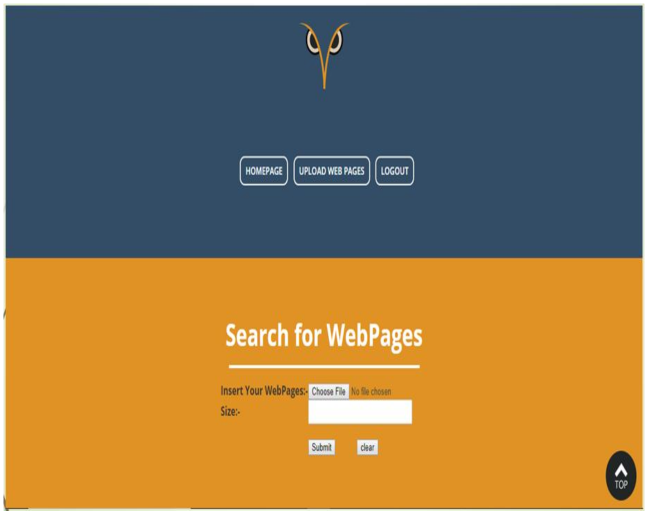


Fig1 Application

Implementation of any software is always preceded by important decisions regarding selection of the platform, the language used, etc. these decisions are often influenced by several factors such as real environment in which the system works, the speed that is required, the security concerns, and other implementation specific details. There are three major implementation decisions that have been made before the implementation of this project. They are as follows:

1. Selection of the platform (Operating System).
2. Selection of the programming language for development of the application.
3. Coding guideline to be followed.

The application is developed using Java, Net beans, Mysql and Navicat.

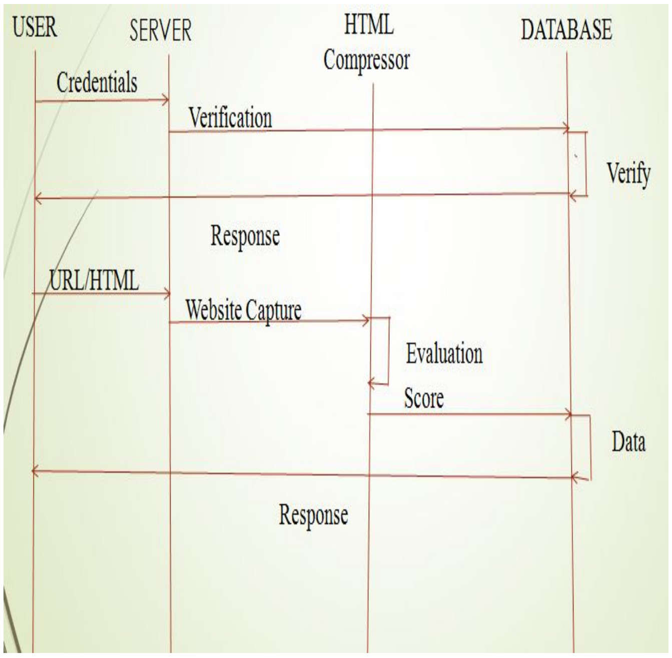
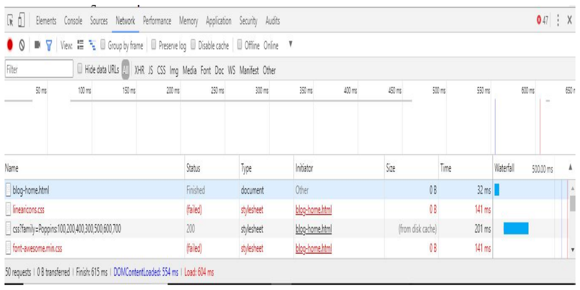


Fig .SEQUENCE DIAGRAM

VI. RESULTS AND DISCUSSION

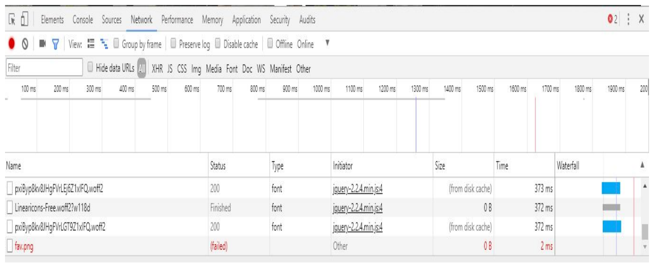
During the execution of the application the sample HTML file is uploaded into the application. Once the files are loaded it returns previous size and optimized size of the file. The path to optimized file is copied from the execution platform and file is opened to check for the optimized performance parameters. The same file is opened through the normal path of C drive , in similar manner the performance parameters are observed. The below figures illustrate the observed results.



Name	Status	Type	Initiator	Size	Time
blog-home.html	Finished	document	Other	0 B	32 ms
linearicons.css	(failed)	stylesheet	blog-home.html	0 B	141 ms
css?family=Poppins:100,200,400,300,500,600,700	200	stylesheet	blog-home.html	(from disk cache)	201 ms
font-awesome.min.css	(failed)	stylesheet	blog-home.html	0 B	141 ms

50 requests | 0 B transferred | Finish: 615 ms | DOMContentLoaded: 554 ms | Load: 604 ms

Fig2a optimized performance



Name	Status	Type	Initiator	Size	Time
pxiByp8kv8JHgFVrLEj6Z1xlFQ.woff2	200	font	jquery-2.2.4.min.js:4	(from disk cache)	373 ms
Linearicons-Free.woff2?w118d	Finished	font	jquery-2.2.4.min.js:4	0 B	372 ms
pxiByp8kv8JHgFVrLGT9Z1xlFQ.woff2	200	font	jquery-2.2.4.min.js:4	(from disk cache)	372 ms
fav.png	(failed)		Other	0 B	2 ms

Fig2b un-optimized performance

As we can see from the above two figures the optimized webpage takes only 32ms to load whereas when the page is loaded from C drive path it takes around 373ms fonts and Jquery of the webpage. This results can be shown with the help of load time graph as follows.

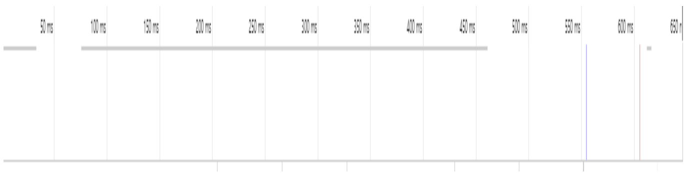


Fig3a optimized differential load time

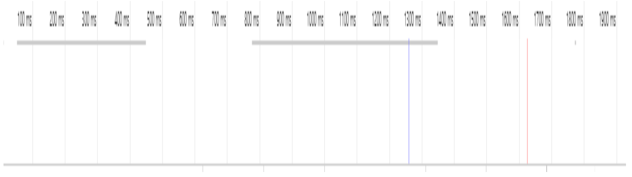


Fig3b un-optimized differential load time

VII. SUMMARY

We present a generic method to optimize the design of web interfaces by finding the placement of interactive elements that maximize measurable usability parameters. The method has been implemented as an opensource software and customized to automatically optimize the position of the checkout button in several types of ecommerce websites using Visa Checkout guidelines. The algorithm is based on the combination of several image processing techniques and a novel perceptually-relevant visual model of the website's layout. Results have shown that the system operates at a high degree of accuracy.

REFERENCES

[1] Yanchun Sun, Dejian Chen, Chao Xin, Wenpin Jiao,"Automating Repetitive Tasks on Web-based IDEs via an Editable and Reusable Capture-Replay Technique",2015 IEEE 39th Annual International Computers, Software & Applications Conference,pp 666-675.

[2] Steve Harrison, Phoebe Sengers, Deborah Tatar,"The Three Paradigms of HCI",CHI 2007, April 28 – May 3, 2007, San Jose, USA,pp 1-18.

[3] Vinayak B. Kadam, Ganesh K. Pakle,"A Survey on HTML Structure Aware and Tree Based Web Data Scraping Technique", International Journal of Computer Science and Information Technologies, Vol. 5 (2) , 2014, 1655-1658.

[4] K. Simon and G. Lausen, "Viper: augmenting automatic information extraction with visual perceptions," In CIKM Conference, pages 381–388, New York, NY, USA, 2005

[5] A. Dulac, D. Pellier, H. Fiorino, and D. Janiszek, "Learning Useful Macro-actions for Planning with N-Grams," in
proceedings of the 25th IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2013), 2013, pp.803– 810.

[6] Yun Young Lee, N. Chen, and R. E. Johnson, "Drag-and-drop refactoring: Intuitive and efficient program transformation," In proceedings of the 35th International Conference on Software Engineering (ICSE'13), 2013, pages 23-32.

[7] Grudin, J. Three Faces of Human-Computer Interaction." IEEE Annals of the History of Computing. Vol.
27, no. 4, Oct.-Dec. 2005. pp 46 - 62. [8] V. A. Saiiuguet, and F. Azavant, "Building intelligent Web applications using lightweight wrappers," Data and Knowledge
Engineering 36(3): 283-316, 2001.

[8] F. Sebastiani, "Machine Learning in Automated Text Categorization," ACM Computing Surveys, Oct. 2001.

[9] Y. Zhang, R. Jin, and Z.-H. Zhou, "Understanding bag-of-words model: a statistical framework," International Journal of Machine Learning and Cybernetics, vol. 1, no. 1–4, pp. 43–52, 2010.

[10] J. D. Gould and C. Lewis, "Designing for usability: key principles and what designers think," Communications of the ACM, vol. 28, no. 3, pp. 300–311, 1985.