Generation of Test Samples for Construction of Dashboard Design Guidelines: Impact of Color on Layout Balance

Olena Pastushenko, Jiří Hynek and Tomáš Hruška,

Brno University of Technology, Faculty of Information Technology,
IT4Innovations Centre of Excellence,
Božetěchova 1/2, 612 66 Brno, Czech Republic
{ipastushenko, ihynek, hruska}@fit.vut.cz

Abstract. The metric-based evaluation of user interfaces is a promising way to quickly evaluate their usability and other various design aspects. However, development of such metrics usually requires a sufficiently large training set of realistic-looking user interface samples, which might not be always easy to find. This paper describes a workflow of the preparation of such samples. It presents a configurable generator based on the composition of simple widgets into a screen according to a predefined model. It also describes a reusable library for simple creation of widgets using capabilities of the JavaScript framework Vue.js. The application of the implemented generator is then demonstrated on the generation of dashboard samples which are used to show the significance of color in the measuring of the layout balance.

Keywords: aesthetics, dashboard, generator, usability guidelines, user testing.

1 Introduction

It is a well-known fact, that evaluation of user interface (UI) usability is an important part of the design process [12]. A UI is the first (and usually the only) part of the system which is seen by users. Factors like aesthetics and even the first impression of user play important role in the usability and acceptability of the whole system [10], [17]. A usual way how to evaluate the usability is to let a suitable subset of users to use the interface and analyze their satisfaction and ability to perform selected tasks. A user testing with a representative group of users can provide sufficient initial feedback [15]. However, it can take time and effort to work with users and understand their needs (often based on subjective feelings). It can significantly increase expenses of the system. Thus – in the age of rapid evolution of artificial intelligence – it seems to be reasonable to try to automatize the evaluation process or at least some of its parts.

One of the approaches helping to do this is the metric-based approach analyzing various user interface aspects [3, 20]. Measured values can be used to detect usability problems according to quantitative guidelines or to calculate the overall level of a UI quality [5, 13]. However, the application of known guidelines can be tricky. There are different kinds of user interfaces with a different purpose and end users. They should be designed in accordance with conventions and knowledge of users [9]. Moreover, every user is a unique one with a subjective way of perception and thinking. Hence, guidelines can't be simply applied in a general way. They should be adjusted according to the design requirements of a chosen interface kind and requirements of users.

Searching for suitable guidelines is based on measuring particular interface aspects (e.g. layout, colors) and the comparison of the results with the reviews of users. It is needed to provide a sufficiently large training set of interface samples rated by users. A manual creation of this set can be elaborate and time-consuming. On the contrary, an automatic generation of samples seems to be a reasonable approach. There are several studies generating black & white screens composed of simple shapes (like squares or triangles) to evaluate the interface aesthetics [1, 16]. There has been an attempt to replace simple shapes by random images in order to simulate a real UIs [2].

In contrast to the screen comprised of simple shapes, there is a significantly higher amount of possibilities how the realistic-looking interfaces can vary from each other. Moreover, only a minor part of generated samples can be considered as well-designed ones. To generate sufficient number of "better-designed" samples, it is necessary to know how the well-designed samples should look like. However, designers don't know that since this information is the matter of the user reviews (Figure 1).

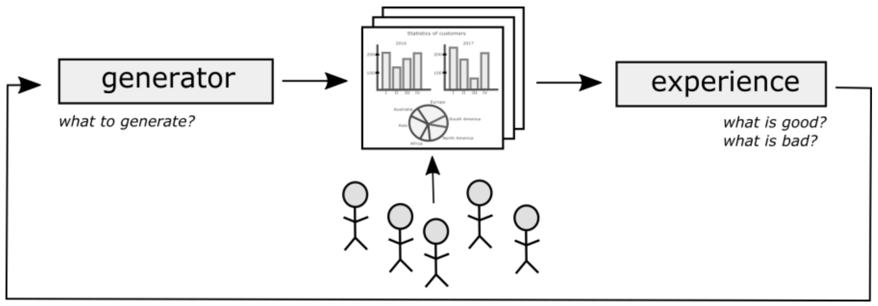

Fig. 1. The cyclic dependency. We need to have a user experience to generate an appropriate set of interface samples to get the user experience.

The goal of this research is to provide a framework for the generation of interface samples used for user reviews. We present a model and a generator which generates samples based on values of the model. The problem of cyclic dependency is solved by iterative improvement of the model and incremental generation of better sets of samples. We demonstrate the applicability of the generator on creating the dashboard samples used for the evaluation of the impact of color on the aesthetic aspect – *layout balance* [11]. We consider the possibility to generalize this approach.

2 Guidelines for Dashboard Development

According to Stephen Few [4], the well-designed dashboard is *a visual display of the most important information needed to achieve one or more objectives*. He also adds that it should be *consolidated and arranged on a single screen so the information can be monitored at a glance*. The second part of the definition establishes the requirements. Few explains it that a user should be able to quickly get familiarized with a content of the dashboard and find something that deserves attention. Understanding the definition in such way makes the requirements vague, based rather on subjective judgment. It does not need to be completely clear how to fulfill them.

Few presents several examples of advice how to design dashboards based on knowledge of the field of data visualization and visual perception – e.g.:

- appropriate selection of charts and colors to emphasize relation between data and to highlight the important information (Figure 2),
- elimination of decorations to decrease the distraction [18],
- application of Gestalt principles to better recognize logical groups [19].

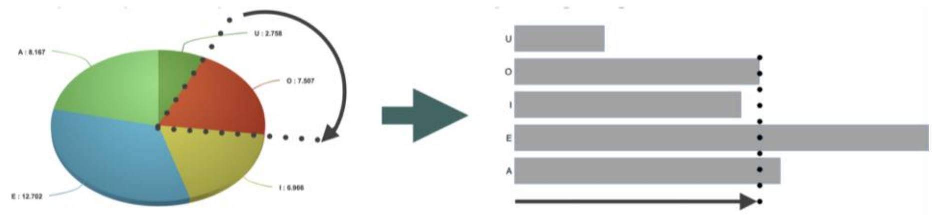

Fig. 2. An example of a design advice. It is easier to compare the values using the bar chart. It is also better to use one subtle color than several vivid colors which should be used to emphasize important information.

The problem of the guidelines is that they are usually described qualitatively and it is difficult to convert them into a set of strict rules. For instance, the selection of appropriate charts or colors usually depends on a situation and it cannot be completely generalized. Gestalt principles are difficult to formally describe as well [8]. Hence, the guidelines are usually simplified into basic rules with a limited usage.

Several dashboard design guidelines have been quantified and evaluated [7]. It has been shown that the well-designed dashboards can be distinguished by the measuring of their overall colorfulness or analysis of their histograms. The results are however based only on a small set of well-designed dashboards. We assume that the increase of the training set would allow to develop more advanced guidelines.

Another approach based on the analysis of interface objects (*widgets*) seems to be a promising way to improve the quality of guidelines [3]. It analyses logical groups of an interface, which is closer to the real interpretation of the interface by a human than the analysis of particular pixels. There have been presented several measures, mostly regarding aesthetics [11]. We believe that this approach could be also used for a deeper analysis of particular widgets (e.g. analysis of widget parts, colors or kinds). However, it is needed to have a higher training set of samples first.

This research works with aesthetic measure *Balance* [11]. We extend the knowledge about this metric by a study of the impact of color on it. We assume that the color of widgets and their dimensions have similarly important impact on the users' judgment of the aesthetic aspects (Figure 3). We use this research as an example to demonstrate the proposed framework for the generation of test samples of dashboards and construction of new guidelines.

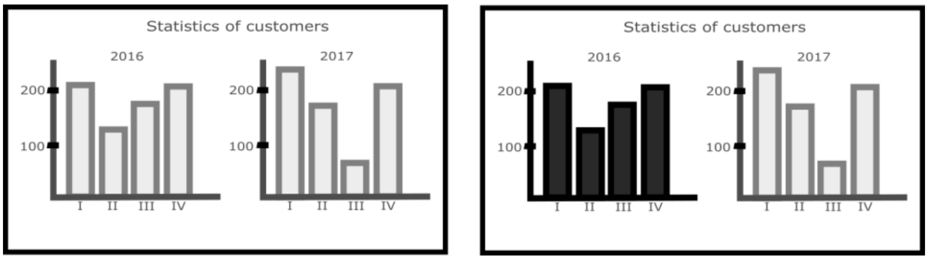

Fig. 3. Guideline hypothesis. The left screen can be considered as horizontally balanced in contrast to the right one.

3 Framework for Guidelines Construction

The framework for the construction and evaluation of guidelines can be characterized by the scheme in Figure 4. The problem of the cyclic dependency described in Figure 1 is solved by the specification of the implicit appearance of the generated interface. It defines the basic style of graphical elements and their data sets. Variability of generated samples is provided by a model which defines attributes of interface objects and by constraints of generation. At the beginning, the generator works with a simple model containing only dimensions of interface objects and type of interface objects. The rest of information is derived from the implicit appearance. The model is then extended iteratively according to a user experience. The more comprehensive the model is, the higher level of variability the generator provides. Constraints are then used to filter some kind of samples when we want to analyze a specific design aspect.

The second part of the framework is focused on the application of a user experience in the construction of design guidelines. Depending on a situation, guidelines can be represented by a simple threshold or an advanced classification algorithm. It is important that analyzed UIs can be represented in various formats (structured description like the web page, raster image etc.). For that reason, the interface needs to be converted to a simplified model which is recognized by the guidelines. It is not the matter of this research to discuss possible approaches of interface processing. An inspiration can be found for instance in [14].

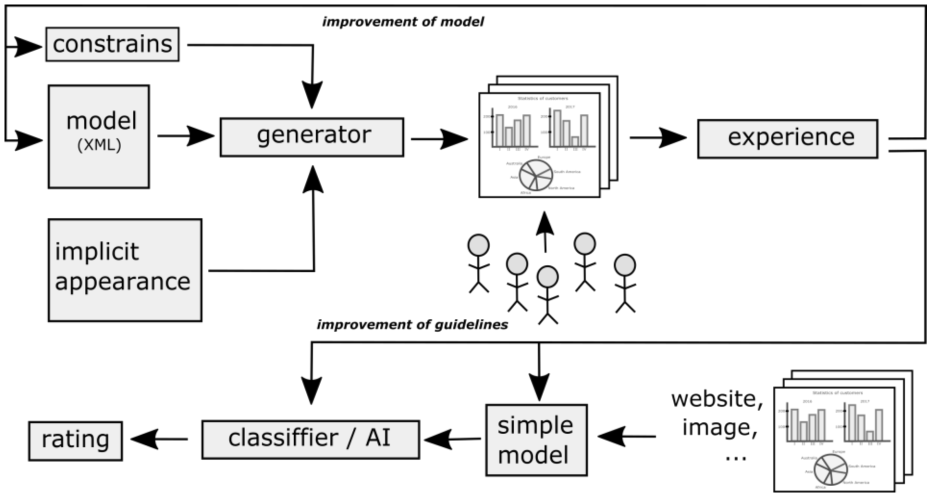

Fig. 4. Scheme of guidelines construction and evaluation. User experience used both for the improvement of model (for generation better samples) and the improvement of guidelines.

4 Generator of Dashboards

There already exist a range of commercial tools for dashboards development. But mostly, they aren't suitable for purposes of this research. Existing tools do not allow to generate a whole set of interface samples which would differ in some specific characteristic (e.g. color or layout) at once. They also miss a possibility to export samples in a suitable format needed for further analyses. The developed generator of dashboards solves these issues.

The architecture of the generator is shown in Figure 5. The backend is developed as RESTful API, which allows easier scalability of the application and independent changes of the server and client sides. It also allows to extend the generator to support the construction of other interfaces types (e.g. mobile applications UI). The backend is based on Node.js environment and MongoDB database, which allows to store the model in JSON format [6]. The frontend is built with Vue.js framework using UXgraph library, which was primarily developed for the purposes of the generator. Another feature of the generator is that it may be extended to a hybrid mobile application, using Cordova wrapper.

The reason behind creating UXgraph (Vue.js extension for data visualization) is to have a predefined set of reusable widgets. They would use the same model but with the possibility of applying the different styles. All widgets of this extension were based on available qualitative design guidelines [4, 18], providing the implicit appearance of widgets. Beta-version of UXgraph is already published as an opensource npm (*Node Project Manager*) library[1], and is available for all Node.js-based projects under MIT License. In this extension, every widget is created as a separate Vue.js component, and it contains all code which can be repeated. These components then extend basic HTML tags with new custom ones. Every widget type component is located in its own single file template, which contains HTML, scripts for the declaration of properties, behavior, and styling.

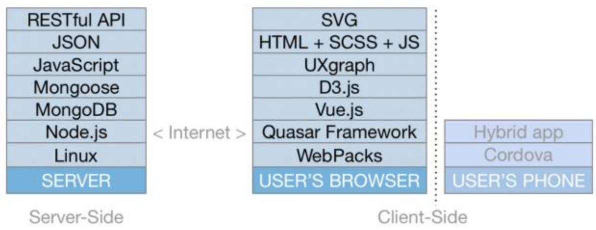

Fig. 5. Architecture of the generator. It illustrates technologies used for the backend and the frontend as independent applications. Frontend may be wrapped to a hybrid mobile application.

Another Vue.js feature which is used in UXgraph is a possibility to pass data to a component and set its properties. Since components are reusable and can be inserted basically in any place of a web page, it is important to keep them in their own isolated scope. Styling parameters from the frontend are passed to the widget components with the help of Vue.js properties (*props*). Figure 6 describes how the line chart widget can be generated using the implicit appearance (provided by the UXgraph library) and explicit appearance (defined by the XML model), which corresponds with the scheme described in Figure 4.

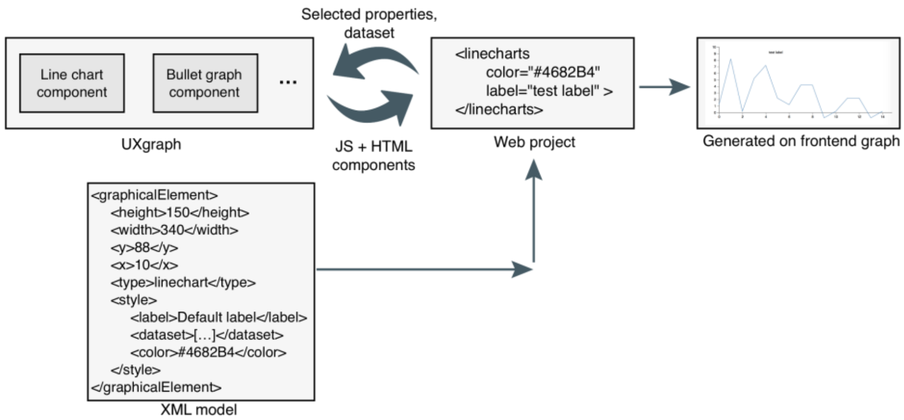

Fig. 6. Custom HTML tags. New HTML tag is defined for every widget type with the help of UXgraph.

---

[1] https://github.com/lirael/vuejs-d3-uxgraph

Figure 7 shows the algorithm for generation of a set of the dashboards interfaces. A user needs to select several parameters, such as: number of samples, template, color scheme and layout density. Afterwards the generator makes a request to the database in order to get the current version of the needed template XML model, as well as its components XML models. Default properties are specified for all UXgraph components in order to make the inclusion of a new component easy. The user needs to specify a property only if he or she wants to use custom values. Based on the retrieved data and the defined parameters, a set of the interfaces is generated. The generator has several built-in templates which are used for the first iteration of samples creation. After a feedback from the reviewers is received, model of these templates can be changed accordingly.

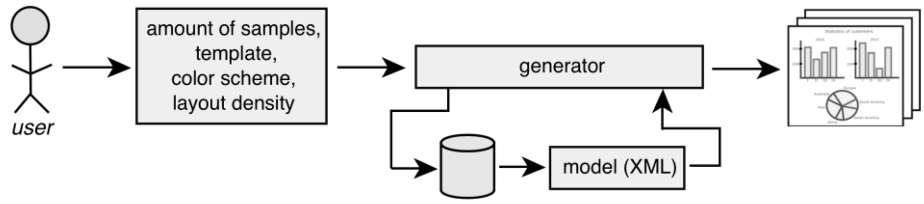

Fig. 7. Example of the generator workflow.

Along with automated algorithm of dashboards generation, the application allows to create new templates and single dashboards manually. In this case, a user has the possibility to add any number of widgets to the dashboards, select different color schemes for each of them, change their dimensions. They only requirement is to respect predefined constraints which help to quickly create realistic-looking samples (e.g. positions of widget need to respect grid layout). Every created configuration can be saved as an interface example and exported to the PNG or XML formats. Figure 8 describes the main possibilities of user's interaction with the application.

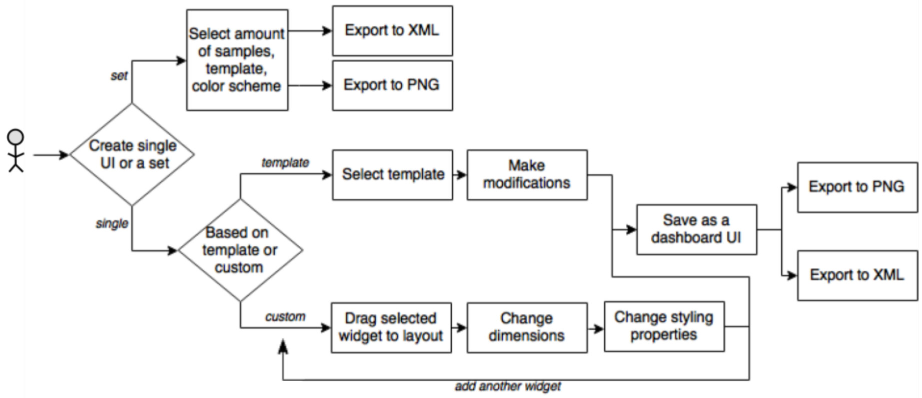

Fig. 8. User interaction possibilities with the developed generator application.

5 Evaluation and Results

The functionality of the generator was verified on a small-scale study evaluating the impact of color on the dashboard balance as described in the section 2. We created 2 reference dashboard layouts which can be considered as relatively balanced according to the formula presented by [11]. Then, we generated 5 realistic-looking dashboards for every layout. Generated dashboards differed in color (mainly by color intensity) and chart types (bar charts and line charts were chosen).

The evaluation was performed with 12 users (mainly with a technical background, but with completely different experience in UI design). They were asked to rate the horizontal and vertical balance and the first impression of the generated screen. They used an interactive form which let them quickly select locations of a perceived equilibriums (Figure 9).

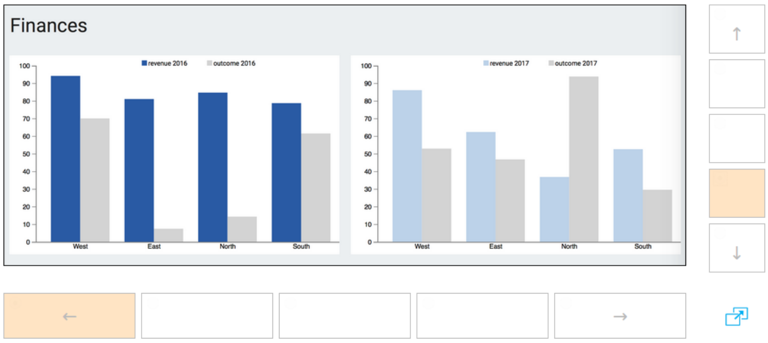

Fig. 9. Example of a dashboard balance rating using a generated dashboard sample. It is rated as horizontally unbalanced with the weight concentrated in the left side and partially vertically unbalanced. The example corresponds with the example presented in Figure 3.

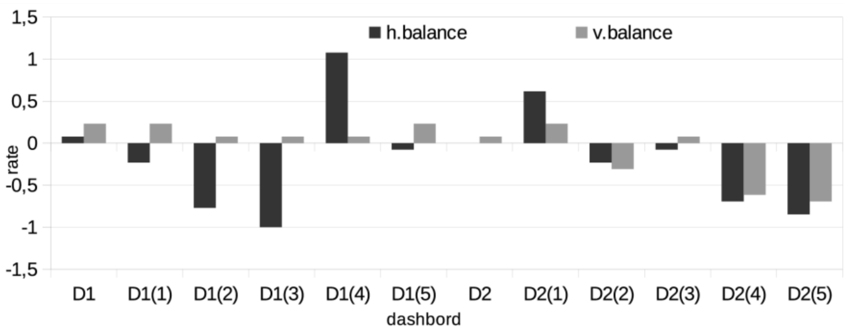

Fig. 10. Average values of horizontal and vertical balance for 2 layouts (D1 and D2) using different color and chart types.

The results of the user testing confirmed the hypothesis (Figure 10). The layout D1 tests only the change of the horizontal balance. The layout D2 tests both axes. Since the reference layouts were composed only from gray rectangles, they were perceived as relatively balanced. Using real widgets didn't dramatically changed the level of Balance (D1(1)). However, even a low change of color intensity of chart in one side caused a high deviation from the equilibrium (D1(2)). Increase of the intensity change made the unbalance stronger. On the contrary, color hue had a low impact on perceived balance (D1(5)). Similar results were observed for the layout D2.[2]

6 Discussion

The small number of samples and users represents the main limitation of the presented study. However, we successfully demonstrated the applicability of the designed workflow and generator, which was the main purpose of the study. The hypothesis was confirmed and the experience of users provided us new suggestions for further evaluations. For instance, we assume that the vertical balance may have a different impact on the overall balance than the horizontal balance (which would contradict the existing formula of Balance [11]).

There are several ways to extend the described generator. New widgets and graphical elements may be added, extending the UXgraph library. The XML model itself may be extended to support more different parameters for each widget. The algorithm for generating test samples could be scaled to change the properties within some range automatically, and not only to change the data (as it does now).

Finally, the model and generator may be extended to support other types of interfaces. A good example would be the support of mobile interfaces since there are similar restrictions (perception of the most important information at the first glance and display on one screen (without scrolling). These limitations are fully supported by the developed application. Plus, it doesn't allow some other mistakes in the interfaces, like overlapping of the elements or making them too small.

7 Conclusion

This paper described the problem of generation realistic-looking interfaces used for the design of quantitative usability and quality guidelines. For this purpose, we created the workflow of samples generation and evaluation and designed the generator which can be extended according to actual requirements. The library used for the creation of interface widgets (used by the generator) can be used as an independent library. The example of generator application was demonstrated on the small-scale study which shows the impact of color on the perception of the dashboard layout balance, which plays role in aesthetics [11]. This experience can be used for the improvement of existing metric of interface balance. Further user testing can be performed to specify a guideline for recommended balance level in a dashboard.

[2] The full form with the samples: http://www.fit.vutbr.cz/~ipastushenko/research/balance.php

Acknowledgments. This work was supported by The Ministry of Education, Youth and Sports from the National Programme of Sustainability (NPU II) project "IT4Innovations excellence in science - LQ1602".

References

1. Altaboli, A., & Lin, Y. (2011). Investigating effects of screen layout elements on interface and screen design aesthetics. Advances in Human-Computer Interaction, 2011, 5.
2. Bauerly, M., & Liu, Y. (2008). Effects of symmetry and number of compositional elements on interface and design aesthetics. Intl. Journal of HCI, 24(3), 275-287.
3. Charfi, S., Trabelsi, A., Ezzedine, H., & Kolski, C. (2014). Widgets dedicated to user interface evaluation. International Journal of HCI, 30(5), 408-421.
4. Few, S. (2006). Information dashboard design.
5. Hollingsed, T., & Novick, D. G. (2007, October). Usability inspection methods after 15 years of research and practice. In Proceedings of the 25th annual ACM international conference on Design of communication (pp. 249-255). ACM.
6. Holmes, S. (2015). Getting MEAN with Mongo, Express, Angular, and Node. Manning Publications Co.
7. Hynek, J., & Hruška, T. (2016, August). Pixel-Based Analysis of Information Dashboard Attributes. In East European Conference on Advances in Databases and Information Systems (pp. 29-36). Springer International Publishing.
8. Jäkel, F., Singh, M., Wichmann, F. A., & Herzog, M. H. (2016). An overview of quantitative approaches in Gestalt perception. Vision research, 126, 3-8.
9. Johnson, J. (2013). Designing with the mind in mind: simple guide to understanding user interface design guidelines. Elsevier.
10. Lindgaard, G., Fernandes, G., Dudek, C., & Brown, J. (2006). Attention web designers: You have 50 milliseconds to make a good first impression! Behaviour & information technology, 25(2), 115-126.
11. Ngo, D. C. L., Teo, L. S., & Byrne, J. G. (2003). Modelling interface aesthetics. Information Sciences, 152, 25-46.
12. Nielsen, J. (1994). Usability engineering. Elsevier.
13. Nielsen, J. (1994, April). Usability inspection methods. In Conference companion on Human factors in computing systems (pp. 413-414). ACM.
14. Reinecke, K., Yeh, T., Miratrix, L., Mardiko, R., Zhao, Y., Liu, J., & Gajos, K. Z. (2013, April). Predicting users' first impressions of website aesthetics with a quantification of perceived visual complexity and colorfulness. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (pp. 2049-2058). ACM.
15. Rogers, Y., Sharp, H., & Preece, J. (2011). Interaction design: beyond human-computer interaction. John Wiley & Sons.
16. Salimun, C., Purchase, H. C., Simmons, D. R., & Brewster, S. (2010, September). Preference ranking of screen layout principles. In Proceedings of the 24th bcs interaction specialist group conference (pp. 81-87). British Computer Society.
17. Tractinsky, N., Katz, A. S., & Ikar, D. (2000). What is beautiful is usable. Interacting with computers, 13(2), 127-145.
18. Tufte, E. R. (1983). The visual display of quantitative information. Graphic Press.
19. Ware, C. (2012). Information visualization: perception for design. Elsevier.
20. Zen, M., & Vanderdonckt, J. (2014, May). Towards an evaluation of graphical user interfaces aesthetics based on metrics. In Research Challenges in Information Science (RCIS), 2014 IEEE Eighth International Conference on (pp. 1-12). IEEE.