VOL. 17, NO. 13, JULY 2022 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences

www.arpnjournals.com

APPLICATION OF CLUSTERING TECHNIQUES IN THE ANALYSIS OF MOUSE TRACKING TESTS

Gabriel E. Chanchí-Golondrino[1], Manuel A. Ospina-Alarcón[1] and Wilmar Y. Campo-Muñoz[2]
[1]Programa de Ingeniería de Sistemas, Facultad de Ingeniería, Universidad de Cartagena, Cartagena, Colombia
[2]Programa de Ingeniería Electrónica, Facultad de Ingeniería, Universidad del Quindío, Armenia, Quindío
E-Mail: gchanchig@unicartagena.edu.co

ABSTRACT

With the growth in the number of applications deployed in cloud app repositories and the increase in the number of users consuming them, usability has become a differentiating factor that promotes the competitiveness of software companies and user productivity. One of the usability tests that focuses on evaluating the layout and/or distribution of the elements in the graphic interface is the mouse tracking test, which has as a challenge the analysis of the mouse trace obtained in the test. In this paper we propose as a contribution the application of unsupervised learning techniques and specifically the use of clustering techniques from the K-Means algorithm in the analysis of the mouse trace obtained in a usability test under the mouse tracking approach. In this way, from the images with the mouse traces generated in these tests, the distribution of the mouse trace points around a set of centroids is obtained, with the purpose of determining the area of the screen in which there was more interaction in the test by the user and in order to make decisions on the layout and distribution of the interface components.

Keywords: clustering, mouse tracking, unsupervised learning, usability.

INTRODUCTION

With the increase in the number of applications developed and published in cloud app stores, as well as in the number of users that consume them, usability has become one of the key aspects that contribute to improving the competitiveness of companies and the productivity of end users[1]-[5]. According to ISO 9241, usability can be defined as the extent to which a software product can be used by specific users to accomplish their objectives effectively, efficiently and with satisfaction in a given context of use[6]-[9]. Similarly, according to ISO 9126-1 and ISO 25000, usability corresponds to one of the attributes that define software quality and can be defined as the ability of a software to be understood, learned and used in a simple and attractive way[10]-[12].

Usability can be involved in the software development process, both in the design phase and in the evaluation phase. In the design phase through the inclusion of heuristics and usability guidelines, while in the evaluation phase through the development of usability tests or user tests, in which a set of users perform a set of tasks on a given software in a controlled environment or usability laboratory[3], [13]-[15]. Among the usability tests, those based on the mouse tracking approach stand out, in which the added value is the capture of the mouse trace generated in the interaction of each user in the different tasks developed in the test[16], [17].

One of the existing challenges in usability testing based on mouse tracking is the analysis of the mouse trace captured from each user who performed the test in a controlled environment, in order to determine whether the interface controls of the evaluated software are properly distributed with respect to the utilization of the hierarchical zones [16]. In this sense, although there are tools that allow the capture of this data in the tests, there is no evidence of the application of machine learning models in the analysis of the mouse trace, so it is necessary to have tools that allow the analysis of the distribution on the screen of the points of the mouse trace and its relationship with the zones on the screen automatically using unsupervised learning algorithms or clustering. Machine learning clustering methods start from a set of examples, which are grouped or organized into clusters according to a notion of similarity that is generally determined by a distance function or metric [18]-[22].

In this paper we propose as a contribution, the application of unsupervised learning techniques and specifically the application of clustering algorithms in the analysis of mouse traces obtained in a usability test.For this purpose, a tool was developed in Python language and using the openCV and scikit-learn libraries, which receives as input the image with the mouse trace of a usability test and obtains as a result the distribution of the points of the trace around a set of centroids, in order to determine the areas of the screen or interface in which there is more interaction. Thus, the results obtained by the analysis tool allow the test coordinator and the development team of the evaluated software to make decisions regarding the distribution of the components in the interface.

The rest of the article is organized as follows: first, the different phases of the methodology used for the development of this research are described. Subsequently, the results obtained through this research are presented, including the description of the design and implementation of an automated tool that allows the application of clustering techniques on the mouse trace captured in a mouse tracking test. Likewise, this section includes the description of a case study developed from the use of the proposed tool on the mouse trace generated by a user in the interaction with the software for the creation of algorithms by means of flowcharts: DFD. Finally, the conclusions and future work derived from this research are presented.

METHODOLOGY CONSIDERED

For the development of this research, 4 methodological phases were defined (see Figure-1): characterization of mouse tracking tests, design of the analysis tool, construction of the analysis tool and finally case study (see Figure-1).

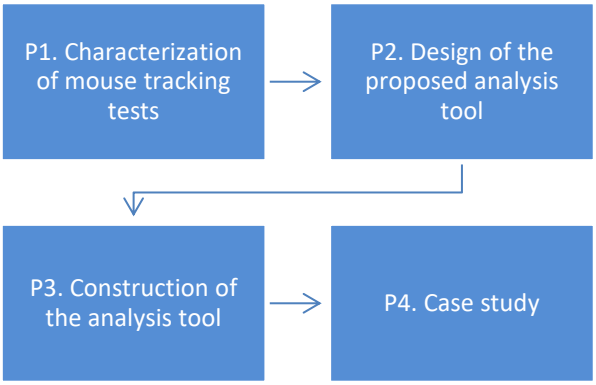

Figure-1. Methodology considered.

Phase 1 of the methodology identified the main characteristics of usability testing under the mouse tracking approach, in which while a group of users perform a set of tasks within an interactive application and in a controlled environment or usability laboratory, the mouse trace is captured using proprietary or free software to obtain as a final result an image with the mouse trace generated in the different tasks of the usability test. Once the mouse tracking tests were characterized in phase 1, phase 2 of the methodology focused on the design of the proposed analysis tool, which included the description of the functional modules that make up the tool, as well as the specification of the different processes developed in the functional modules and the generation of the high-level interfaces that allow compliance with them. Based on the design defined in phase 2, in phase 3 the techniques and technologies for processing the images with the mouse trace through the application of clustering techniques were selected. Once the above was done, the implementation of the analysis tool using the selected techniques and technologies was carried out. Thus, the Python language was chosen for the implementation of the tool, so that the Python openCV library was selected for the processing of the images with the mouse trace, while the Python scikit-learn library was used for the application of the clustering techniques. Finally, in phase 4 of the methodology, a case study was conducted, in which use was made of the tool implemented for the analysis of the data obtained in a mouse tracking test performed on the DFD flowchart creation tool.

RESULTS AND DISCUSSIONS

This section describes the results obtained from the development of this research, which includes the definition of the functional modules of the proposed analysis tool, the description through a flowchart of the processes developed by the tool for the application of clustering techniques on the images resulting from a mouse tracking test and finally, the description of the results obtained in the application of the proposed tool on the results of a case study developed in this research.

Accordingly, Figure-2 shows the functional modules that make up the mouse tracking test analysis tool, using clustering techniques.

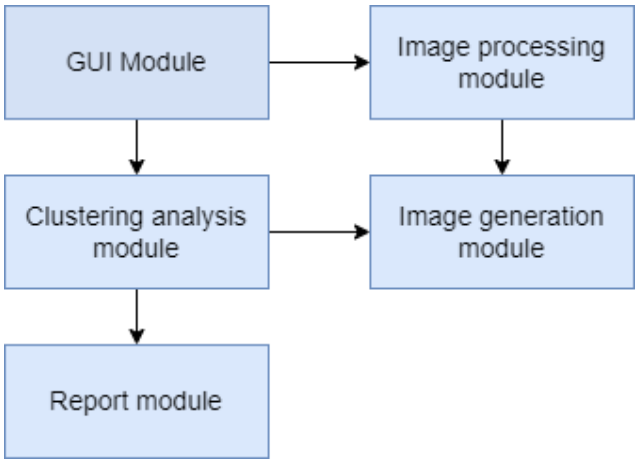

Figure-2. Block diagram of the tool.

The GUI module of the analysis tool is in charge of displaying the tool interface and managing the different components (labels, buttons, text areas, text boxes), as well as handling the different events that allow the end user (usability test coordinator) to interact with the interface. Thus, from the interface components generated by this module, the test coordinator can attach the image with the mouse trace generated in a mouse tracking test, in order to perform the clustering analysis and obtain as output a new image showing the centroids of each cluster obtained and its distribution in the 4 zones of the screen (upper left, upper right, lower left and lower right). The GUI module was implemented through the use of the Python Tkinter library. The image processing module, on the other hand, is responsible for transforming the input image to grayscale and iterating it in order to determine the coordinates of the mouse trace distributed along the image to store them in a numerical array. The image processing module was implemented through the use of the openCV computer vision library. On the other hand, the clustering analysis module is in charge of taking the numerical array of coordinates generated by the image processing module and obtaining a set of clusters with their associated centroids, as well as the number of instances or samples corresponding to each of the clusters. The clustering analysis module was implemented through the use of the scikit-learn library of the Python language. The image generation module is responsible for drawing on the image with the original mouse trace, the 4 zones of the screen and the clusters obtained by the cluster analysis module, in order to identify where the points of interest of the trace are concentrated. The image generation module was implemented using the Python openCV library.

Finally, the reporting module is responsible for generating a .CSV file with the results of the clustering analysis, which includes the centroids of each cluster and the number of instances spatially distributed around each centroid. The reporting module was implemented through the use of the Python file class.

On the other hand, Figure-3 shows a flowchart representing the different functional processes developed by the analysis tool for the application of clustering techniques on the mouse trace images of a usability test.

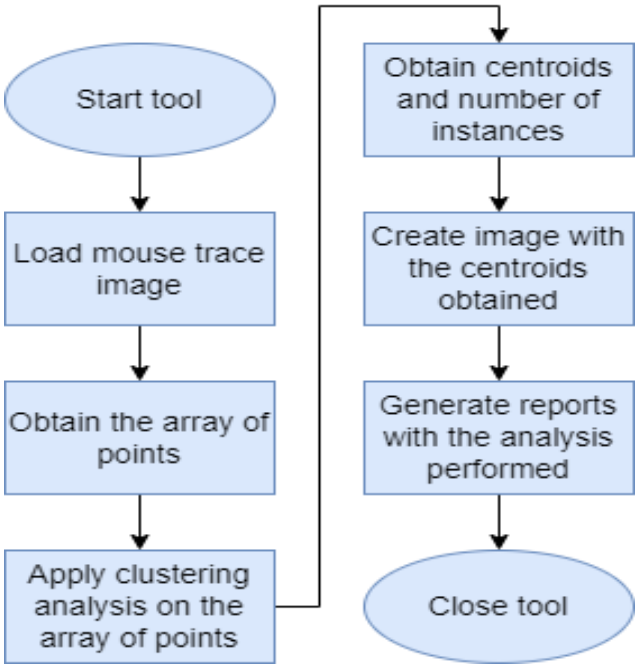

Figure-3. Tool flow chart.

Thus, once the analysis tool is started, the test coordinator proceeds to load the image with the mouse trace obtained in the mouse tracking test, which is normally generated by a tracking tool and includes the mouse trace in black on a white background. Thus, once this image is loaded, the tool iterates over it in order to identify the points or coordinates of the black pixels and store them in a numerical array. Once the array of coordinates has been obtained, the analysis tool applies the clustering model based on the K-Means algorithm, taking into account a given number of clusters. As a result, the tool obtains the coordinates of the centroids belonging to each cluster and the number of instances associated to each centroid. The coordinates obtained for the centroids are plotted on the original image, which also shows the four zones on the screen, so that it is possible to see the points of interest on the screen and the zone to which these belong. Finally, the tool allows generating a report in .CSV format with the results of the application of the clustering model on the evaluated mouse trace.

Once the functional processes developed by the tool have been described, the graphical interface of the tool is presented below, which is made up of 3 tabs: "Process Image", "Clustering Analysis" and "Graphical Analysis", as shown in Figure-4.

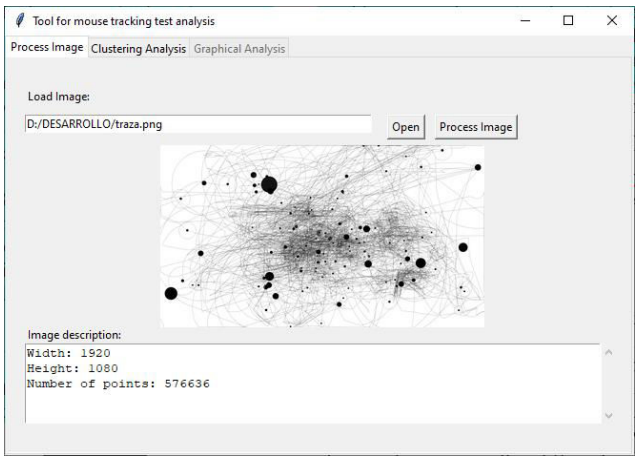

Figure-4. Analysis tool interface.

The "Process Image" tab is shown in Figure-4 and is in charge of loading and visualizing the image with the mouse trace by pressing the "Open" button, as well as obtaining the number of points of the trace and their associated coordinates by pressing the "Process Image" button. These coordinates are stored in a numerical array, for use by the clustering models in the other tabs of the tool. The process of obtaining the coordinates of the mouse trace is performed through the advantages provided by the openCV Python library. Thus, as an example, in Figure-4 a mouse trace image with dimensions 1920x1080 was loaded, which was generated using the IoGraph tool [23] and whose number of points of the trace is 576636.

On the other hand, the "Clustering Analysis" tab is presented in Figure-5 and is responsible for the application of the clustering techniques on the array of points or coordinates obtained in the "Process Image" tab. Thus, by selecting in the "Clustering Analysis" tab the number of clusters to be determined and pressing the "Analyze" button, the centroids corresponding to each cluster and the number of instances that have been categorized in each cluster are obtained. In the same way, in this tab it is possible to generate a report in .CSV format by clicking on the "Report" button. The process of obtaining the clusters and their associated centroids is performed by the analysis tool through the use of the scikit-learn Python library. As an example, Figure-5 shows the results of the analysis with 3 clusters, performed on the image of the mouse trace shown in Figure-4, so that the following 3 centroids were obtained: C1={X=1469.14, Y=551.51, Number of Instances=183560, %Instances =31.833%}, C2={X=496.57, Y=564.308, Number of Instances=157847, %Instances=27.374%}, C3={X=963.952, Y=531.744, Number of Instances=235229, %Instances=40.793}. Thus, it can be seen that the cluster with centroid C3 is the one with the highest number of associated points with 40.793%, and it is located in zone 2 of the screen (upper right part). On the other hand, the cluster with the least number of instances

or points is cluster 2, which has 27.374% of the instances and is located in zone 3 of the screen (bottom left).

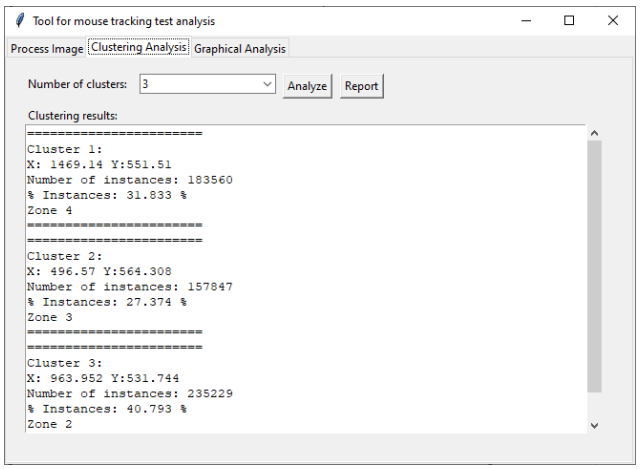

Figure-5. "Clustering Analysis" tab of the tool.

Finally, Figure-6 shows the "Graphical Analysis" tab, where it is possible to obtain graphically the representation of the centroids obtained in the original mouse trace image and their relationship with the 4 screen zones considered in this proposal. The new image with centroids and screen zones is generated using the advantages provided by the openCV library.

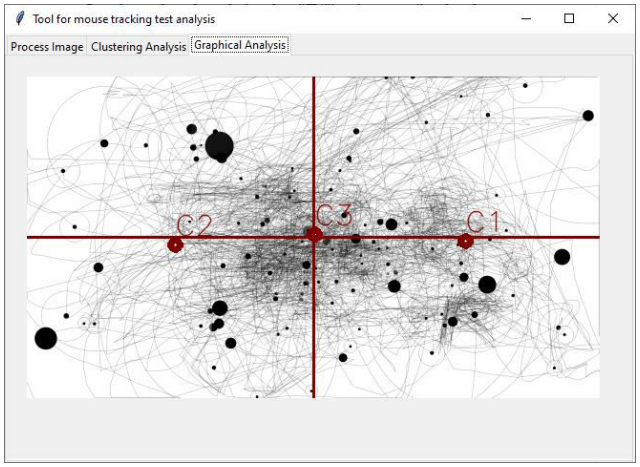

Figure-6. "Graphical Analysis" tab of the tool.

As an example, it is possible to observe that regarding the analysis with 3 clusters for the image with a trace of 576636 presented in Figure-4, it is obtained that the centroid C1 of cluster 1 belongs to zone 4, the centroid C2 of cluster 2 belongs to zone 3, while the centroid C3 of cluster 3 belongs to zone 2 and is the one with the highest number of associated points.

Once the different tabs of the graphical interface of the analysis tool have been presented, a case study is described below, where the proposed tool was applied on the mouse trace resulting from a usability test in which a user interacted with the DFD software, which allows the creation of programming algorithms through flowcharts (see Figure-6).

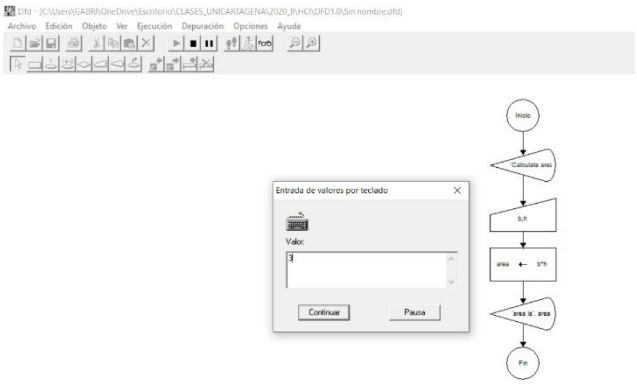

Figure-7. DFD tool.

Within this test, a user was requested to build a flowchart algorithm to calculate the area of a rectangle, while the IoGraph tool was used to capture the mouse trace of the test. Thus, by applying the analysis tool to the mouse trace of the test, which has a total of 84463 points, the results presented in Figure-8 were obtained for a total of 3 clusters.

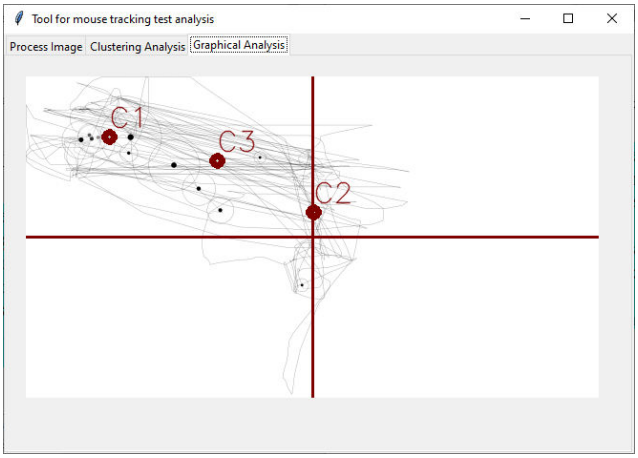

Figure-8. Results obtained in the case study.

It can be seen that the centroid C1 and centroid C3 are located in zone 1 of the screen (upper left part), while centroid C2 is located in zone 2 of the screen (upper right part). Similarly, with respect to the instances of each cluster, Table-1 shows the numerical and percentage distribution of the instances around the 3 defined centroids.

Table-1. Clusters and centroids defined.

Cluster	Centroids	Number of instances and percentage
1	C1={X =278.973, Y=203.354}	29954 35.464%
2	C2={X =963.554, Y = 456.442}	26426 31.287%
3	C3={X =640.174, Y=283.582}	28083 33.249%

Likewise, it is possible to see from Table 1 that cluster 1 is the cluster with the highest number of instances with a distribution percentage of 35.464%; however, the 3 clusters have a number of instances close to each other. Although both clusters 1 and 3 were classified in zone 1, it is important to note that 31.248% of the instances were classified in zone 2. This is explained by the fact that the evaluated tool displays the flowchart that is generated by the user in the center of the screen, while the icons or components of the flowchart are located in the upper left part of the screen, so that the path shown in the mouse trace can make the process of creating a flowchart inefficient. Thus, it is recommended that the DFD tool places the flowchart on the left and by default the tool window has a smaller width.

CONCLUSIONS

Taking into consideration the need for methods, techniques and tools that support the analysis of the results obtained in a usability test under the mouse-tracking approach, in this article we propose as a contribution the application of unsupervised learning techniques and specifically of clustering models for the analysis of the mouse trace resulting from a mouse-tracking test. Thus, for the application of these methods, a tool was developed in the Python language and with the support of the scikit-learn library, which from the array of coordinates of the mouse trace allows obtaining a set of clusters and their associated centroids, as well as their spatial distribution with respect to the coordinates of the trace.

The method employed based on the use of the K-Means algorithm and the analysis tool proposed to implement it are intended to support the coordinator of a usability test in terms of identifying the areas of the screen where the user focuses his attention, in order to facilitate decision-making regarding the improvement of the layout of the interface components. In this sense, the proposed tool has the advantage of generating a new image from the image with the original mouse trace, in which the defined centroids and their location with respect to the four defined screen zones are presented.

The open software tools and/or libraries used and belonging to the Python language proved to be suitable for the application of clustering techniques on the mouse trace obtained in a usability test. Thus, the Python Tkinter library was used to generate the tool's graphical interface and to manage the events associated with its different components. The openCV library compatible with the Python language was used to obtain the points or coordinates that make up the mouse trace and to generate the new image of the mouse trace in which the centroids obtained in the clustering analysis were included. Finally, the Python library scikit-learn was used to apply the clustering models on the points or coordinates belonging to the analyzed mouse trace.

Through the case study developed on the DFD tool from the analysis tool, it was possible to determine that the three clusters obtained have a similar number of instances, which is represented by a distribution of more than 30% for each cluster. Likewise, two of these clusters were classified in zone 1 of the screen (upper left), while the remaining cluster was classified in zone 2 of the screen (upper right). This led to the conclusion that the location of the flowchart within the DFD tool requires the user to make long mouse traces from the upper left side where the components of the diagram are located, to the center of the screen where the diagram is being generated. In this sense and in order to make the interaction with the user more efficient, it is necessary that the designers and developers of the tool place the flowchart generated by the tool on the left side of the screen and below the icons with the components of the flowchart.

Finally, this proposal is intended to be fed back by including other clustering methods, such as those based on the DBSCAN algorithm. It is also intended to extrapolate this proposal to the mobile device environment, in order to evaluate the distribution of the interface components in these contexts.

ACKNOWLEDGMENTS

The authors would like to thank the Universidad de Cartagena-Colombia and the Universidad del Quindío-Colombia for their support in the development of this research.

REFERENCES

[1] D. Hering, T. Schwartz, A. Boden and V. Wulf. 2015. Integrating usability-engineering into the software developing processes of SME: A case study of software developing SME in Germany. Proc. - 8th Int. Work. Coop. Hum. Asp. Softw. Eng. CHASE 2015, pp. 121–122, doi: 10.1109/CHASE.2015.22.

[2] C. Dinkel, D. Billenstein, D. Goller and F. Rieg. 2018. User-oriented optimization of the GUI of a finite element programme to enhance the usability of simulation tools. doi: 10.23919/SEEDA-CECNSM.2018.8544936.

[3] D. M. Delgado Agudelo, D. F. Girón Timaná, G. E. Chanchí Golondrino and K. Márceles Villalba. 2018. Propuesta de una herramienta para la estimación de la satisfacción en pruebas de usuario, a partir del análisis de expresión facial," Rev. Colomb. Comput., 19(2): 6-15, doi: 10.29375/25392115.3438.

[4] K. Radle and S. Young. 2001. Partnering usability with development: how three organizations succeeded. IEEE Softw., 18(1): 38-45, Jan. 2001, doi: 10.1109/52.903164.

[5] M. Monroy-Rios, G. Chanchí-Golondrino and P. Acosta-Vargas. 2020. Sistema automatizado para la conducción de inspecciones de usabilidad y

accesibilidad – SIUSA. Rev. Ibérica Sist. e Tecnol. Informação, no. E32, pp. 50-63.

[6] N. Kerzazi and M. Lavallee. 2011. Inquiry on usability of two software process modeling systems using ISO/IEC 9241. In Canadian Conference on Electrical and Computer Engineering, pp. 000773–000776, doi: 10.1109/CCECE.2011.6030560.

[7] P. Weichbroth. 2020. Usability of mobile applications: A systematic literature study. IEEE Access, 8: 55563-55577, doi: 10.1109/ACCESS.2020.2981892.

[8] V. F. Martins, M. De Paiva Guimaraes, and A. G. Correa. 2013. Usability test for Augmented Reality applications. doi: 10.1109/CLEI.2013.6670668.

[9] L. Martin and M. Dudda. 2017. Usability reasoning using OWL 2 RL in Proceedings - 2016 10th International Conference on the Quality of Information and Communications Technology, QUATIC 2016, pp. 155-159, doi: 10.1109/QUATIC.2016.040.

[10] S. Rochimah, H. I. Rahmani and U. L. Yuhana. 2015. Usability characteristic evaluation on administration module of Academic Information System using ISO/IEC 9126 quality model. 2015 Int. Semin. Intell. Technol. Its Appl. ISITIA 2015 - Proceeding, pp. 363-368, doi: 10.1109/ISITIA.2015.7220007.

[11] C. Santos, T. Novais, M. Ferreira, C. Albuquerque, I. H. De Farias and A. P. C. Furtado. 2016. Metrics focused on usability ISO 9126 based. in Iberian Conference on Information Systems and Technologies, CISTI, vol. 2016-July, doi: 10.1109/CISTI.2016.7521437.

[12] M. D. Dzulfiqar, D. Khairani and L. K. Wardhani. 2019. The Development of University Website using User Centered Design Method with ISO 9126 Standard. doi: 10.1109/CITSM.2018.8674325.

[13] G. E. Chanchí, M. Sánchez, and W. Y. Campo. 2018. Sistema software para el análisis del estrés mental en test de usuarios. Campus Virtuales, 7(2): 105-114, [Online]. Available: www.revistacampusvirtuales.es.

[14] G. E. Chanchi, M. A. Ospina and J. L. Pérez. 2020. Sistema IoT para la monitorización de la variabilidad del ritmo cardiaco en pruebas de usabilidad. Rev. Espac., 41(25): 2020, [Online]. Available: https://www.revistaespacios.com.

[15] G. E. Chanchí-Golondrino, M. A. Ospina-Alarcón and W. Y. Campo-Muñoz. 2021. Herramienta para el Análisis de Evaluaciones Heurísticas de Usabilidad Mediante Lógica Difusa. Ing. Y Compet., 24(1), doi: 10.25100/iyc.v24i1.11095.

[16] D. A. Albornoz, S. A. Moncayo, S. Ruano-Hoyos, G. E. Chanchí-Golondrino and K. Márceles-Villalba. 2019. Sistema software para la ejecución de pruebas de usabilidad bajo el enfoque de mouse tracking. TecnoLógicas, 22: 19-31, doi: 10.22430/22565337.1511.

[17] C.-F. Peng and W.-H. Liao. 2017. Evaluation of Interactive Data Visualization Tools Based on Gaze and Mouse Tracking. in 2016 IEEE International Symposium on Multimedia (ISM), pp. 431-434, doi: 10.1109/ISM.2016.0099.

[18] X. Chu, J. Lei, X. Liu and Z. Wang. 2020. KMEANS Algorithm Clustering for Massive AIS Data Based on the Spark Platform. 2020 5th Int. Conf. Control. Robot. Cybern. CRC 2020, pp. 36-39, doi: 10.1109/CRC51253.2020.9253451.

[19] S. V. Gajbhiye and G. B. Malode. 2018. Enhancing pattern recognition in social networking dataset by using bisecting KMean. in Proceedings of 2017 International Conference on Intelligent Computing and Control, I2C2 2017, 2018-Janua: 1-5, doi: 10.1109/I2C2.2017.8321776.

[20] J. H. Xu and H. Liu. 2010. Web user clustering analysis based on KMeans algorithm. in ICINA 2010 - 2010 International Conference on Information, Networking and Automation, Proceedings, vol. 2, doi: 10.1109/ICINA.2010.5636772.

[21] I. Pérez-Verona and L. Arco-García. 2016. Una revisión sobre aprendizaje no supervisado de métricas de distancia. Rev. Cuba. Ciencias Informáticas, 10(4): 43–67, [Online]. Available: https://www.redalyc.org/pdf/3783/378349316004.pdf.

[22] A. Liu, J. Lu and G. Zhang. 2021. Concept Drift Detection via Equal Intensity k-Means Space Partitioning. IEEE Trans. Cybern., 51(6): 3198-3211, Jun. 2021, doi: 10.1109/TCYB.2020.2983962.

[23] IOGraphica. 2022. IoGraph. [Online]. Available: https://iographica.com/.