Towards the Automatic Restructuring of Structural Aesthetic Design of Android User Interfaces

Article *in* Computer Standards & Interfaces · November 2021

DOI: 10.1016/j.csi.2021.103598

Narjes Bessghaier
École de Technologie Supérieure
7 PUBLICATIONS **38** CITATIONS
SEE PROFILE

Makram Soui
Saudi Electronic University
57 PUBLICATIONS **386** CITATIONS
SEE PROFILE

Nadia Ghaibi
University of Gabès
3 PUBLICATIONS **12** CITATIONS
SEE PROFILE

Computer Standards & Interfaces 81 (2022) 103598

Contents lists available at ScienceDirect

Computer Standards & Interfaces

journal homepage: www.elsevier.com/locate/csi

Towards the automatic restructuring of structural aesthetic design of Android user interfaces

Narjes Bessghaier [*,a], Makram Soui [b], Nadia Ghaibi [c]

[a] Ecole de Technologie Superieure (ETS), Montreal, QC, Canada
[b] College of Computing and Informatics Saudi Electronic University, Riad, Saudi Arabia
[c] Artificial Intelligence Research Unit, University of Manouba, Tunisia

ARTICLE INFO

Keywords:
Mobile UI
Aesthetic quality
Design guidelines
Restructuring

ABSTRACT

End-user engagement heavily relies on the aesthetic design of the application's user interfaces. Designers are keen to create user interfaces that are usable and appealing. However, fundamental design issues , such as inconsistent padding and margins, cluttered user interfaces, and a high variation of element sizes, are too frequent in a UI design.. Prior studies provided ready-to-implement user interface alternatives. However, the aesthetic quality of these alternative designs is not guaranteed, and it limits the creativity of designers. Therefore, we present in this study an automated approach for restructuring a user interface structural design based on its data model. Our framework checks the violation of 13 generic structural design standards provided by Google Material Design. Then, the framework provides a set of recommendations for each violated guideline based on the specifications of the evaluated MUI. As a proof of concept, we used the tool ADDET to evaluate the quality of the original and restructured versions of 511 user interfaces in terms of the number of aesthetic defects and aesthetic properties. The results revealed a significant positive difference with a mean of 0.59 for the benefit of restructured user interfaces with an improved set of 7 quality metrics. We have also found that our approach could improve to an extent the accessibility of some designs by providing bigger element sizes.

1. Introduction

End-users interact with mobile applications via the Mobile User Interface (MUI). The MUI is a touch-sensitive display that recognizes the user's events and displays the results via graphical components. Therefore, the MUI should be usable and appealing to the end-users to increase their engagement with the application. For example, Mahajan and Shneiderman [1] discovered that inconsistent desktop GUIs (Graphical User Interface) could increase job completion time by 10% to 25%. At the same time, 60% of internal code defects occurred at the GUI level [2].

Aesthetic, which determines the level of the visual appeal of a design, is one of the critical criteria of usability [3,4]. Aesthetic refers to the MUI's perceived aesthetic level and how it engages the interest of users. The aesthetic quality encompasses the structural (a layout-related property dealing with the geometric placement of graphical elements) and the colorful aspects (the choice of harmonically appropriate colors). There are two types of aesthetic evaluation approaches. 1) *Qualitative/Subjective evaluation:* the emphasis is on determining whether the design follows a set of visual principles [1,5,6]. Experts or end-users subjectively perform this evaluation. 2) *Quantitative/Objective assessment:* this evaluation is based on the use of metrics to analyze the properties of a design component, such as cohesion, density, and balance. A rigorous metric assessment considers how good or bad a certain structural attribute is.

Existent studies have empirically demonstrated that design aesthetic impacts the user's engagement level [7–9], the perceived usability [10–12], and users loyalty [13]. According to Trkyilmaz et al. [14], end-users value design aesthetic as much as the operational side of an application. Several approaches are proposed to support MUI designing with early concept generation using prototyping tools [15,16], generating similar models of existing MUIs to choose the best design alternatives [17], and detecting aesthetic defects [18,19]. Other studies targeted restructuring an MUI design by providing recommendations on how to address a violated guideline manually [20].

The problem with the proposed solutions regards the non-guaranteed aesthetic quality of the proposed MUI design alternatives. Furthermore, there is no automated approach initiative that repairs structural design

* Corresponding author.
E-mail addresses: narjes.bessghaier.1@ens.etsmtl.ca (N. Bessghaier), m.soui@seu.edu.sa (M. Soui).

https://doi.org/10.1016/j.csi.2021.103598
Received 8 October 2020; Received in revised form 18 October 2021; Accepted 30 October 2021
Available online 5 November 2021

defects to the best of our knowledge.

To this end, we propose the *MUI Designer* framework to support developers to restructure Android applications' user interface structural aesthetic designs automatically. The MUIDesigner automatically reconstructs an MUI grid layout structure based on 13 structural Google MUI design guidelines, such as *colorfulness, typography, padding,* etc. First, the framework reads the data-model tree of the MUI to access the structural properties of each graphical element. Then, the framework generates a new data-model tree that best re-positions the visual elements in the MUI.

Mainly, the proposed framework starts with evaluating 15 structural properties and detecting five structural aesthetic defects [19]. The defects detection precision reached a score of 71%. Second, based on the collected design guidelines, the framework checks the violation of these guidelines and triggers a recommendation based on the specifications of the evaluated MUI. Finally, the framework automatically generates a new MUI tree model, including the restructured design properties. Developers and designers could easily locate the restructured elements in the new tree model as the changes are highlighted in red. It is also easy to read from or integrate the new tree model generated using the CSV format.

As a proof of concept, we address the following two preliminary questions to identify global design improvement.

- *PQ1: To what extent have the structural properties of the MUI been improved?*
  In this question, we select a set of eight structural aspects representing the main properties of our restructuring approach to determine if there is a global improvement in the layout.
- *PQ2: What type of layouts were most improved by the restructuring approach?*
  As Android application user interfaces could be designed using a variety of grid layouts, we aim to examine which ones were structurally improved the most by our approach. Furthermore, this question could pertain to the best layouts to use when conceiving an MUI design, as these layout types are supposed easy to redesign.

In the following two research questions, we evaluate and compare the quality of 15 aesthetic metrics between the original and restructured MUIs. We also examine whether the number of aesthetic defects has decreased.

- *RQ1: To what extent have the structural aesthetic metrics been improved?*
  The goal behind this question is to examine the impact of our restructuring approach on the quality of 15 structural aesthetic metrics.
- *RQ2: To what extent has the structural aesthetic defects number been reduced?* To complement the results found in RQ1, we also compute the number of detected aesthetic defects in the original and the restructured MUI designs.

Our qualitative and quantitative results demonstrate that the framework was successful in restructuring some of the structural properties of the MUI. The results reveal that our approach improved seven out of 12 metrics. The number of defects has also decreased in the restructured designs, with a mean difference of 0.59. Furthermore, we noticed that our approach could improve accessibility by providing bigger element sizes.

The rest of this paper is structured as follows: Section 2 reviews existing studies that are related to design defects. MUIDesigner architecture and workflow is detailed in Section 3. We evaluate the efficiency of the proposed approach in Section 4. We report the threats to validity in section 5, along with the conclusion in Section 6.

2. Related work

The evaluation of user interface quality dates back to the early ages, and it has been largely studied in the last years. Several modern engineering tools and approaches are used to evaluate the user interfaces, such as Multi-Objectives and image processing algorithms. Several studies are being conducted to assess different functional and non-functional user interface properties. In the following paragraphs, we provide studies relevant to evaluate the properties of user interfaces.

User interface evaluation: A wide range of usability evaluation tools and approaches have been proposed, including laboratory testing, surveys, and inspection methods. [21–23]. The inspection methods and questionnaires aim to evaluate desktop GUIs with different User eXperience (UX) needs than smartphones. For this reason, usability experts have attempted to adapt those principles to the mobile context. Yanez [24], for example, adjusted a desktop-centric checklist for the mobile interface to detect specific MUI issues and established independent OS guidelines. Kuparinen et al. [25] presented a comparison of generic and domain-specific heuristics for map-based applications. The findings revealed that the proposed domain-specific checklist aided users in identifying additional usability issues. These studies manually created heuristics to evaluate an MUI usability. Heuristics evaluation is one of the most widely used methods for assessing the usability of mobile apps during the early design phase. It puts an MUI to the test using a set of rules [26]. This method is effective, but it is also error-prone and subjective. Similarly, Quiñones et al. [27] have developed a formal methodology to create usability heuristics. In addition, the authors simplify the application of the proposed heuristics by providing steps on how to iterate and apply the heuristic properly.

Automated MUI evaluation: Ines et al. [18] utilized genetic programming in MUIs to create evaluation rules for automatic usability detection. The authors considered user profile parameters such as motivation, level of experience, and age when developing user-adapted detection rules. In addition, Riegler and Holzmann [28] proposed a set of eight complexity metrics (number of UI elements, smallness of element, misalignment, density, imbalance, color complexity, typography complexity, and inconsistency) to determine the visual appearance of mobile app user interfaces. The method is based on a computer vision technique that requires no parsing of codes. The critical weakness of the analysis is that metric thresholds are not tuned. Besides, no importance weights were given to the metrics. GUIEvaluator, a metric tool developed by Alemerien and Magel [29] automatically assesses the complexity of a GUI. The method measures five quantitative metrics: orientation, grouping, size, density, and balance. In addition, Soui et al. [30] have developed a plugin called PLAIN, which assesses the usability aesthetic quality of Android mobile app user interfaces. We discovered that the PLAIN parser considers the *ViewGroups* and the *layouts* as visual components when calculating metrics. Such a method may result in an incorrect number of visual graph components. Zen and Vanderdonckt proposed QUESTIM, a region-based method for measuring an MUI as a screenshot using metrics like density, balance, balance, symmetry, borderBalance, and borderDensity [31,32]. The user will access the website, upload a screenshot, and manually draw the regions. This method does not include automated metric threshold adjustment, as well as, the metrics computation could differ based on the extracted regions.

Colorfulness Several research studies have proven the existence of an effect of colours on emotions [33–35]. In particular, the choice of colors [36,37] along with the saturation and brightness have shown a strong effect on the user's emotional state [38]. Reinecke et al. [39] have demonstrated that the visual complexity and the colorfulness of web pages can take up to 48% in users' first impressions. As well, Miniukovich et al. [37] found that color depth, contrasts, and dominant colors have shown a strong correlation with aesthetics. Additionally, Dianne et al. accentuated how coloring fosters trust and satisfaction of websites [40]. However, existing studies on MUI colorfulness evaluation are limited to subjective end-user ratings. Hence, our approach builds on previous research by automatically analyzing the colorfulness of the MUI based on Google design principles and recommending the optimal color shades to be used. In addition, we specify the number of pixels that each color shade should cover to provide harmonically ordered hues.

Guidelines Conformance check: Mathur et al. [20] established a framework that detects violations of usability principles for Android and iOS applications. Based on code analysis, the framework verifies the adherence to the rules and makes recommendations for a failing standard. Another study focused on improving the accessibility and usability of Web interfaces [41]. For each detected accessibility problem, the authors offered an adaptive refactoring practice. Some of the problems are related to structural MUI difficulties, such as overloaded pages and complex interaction with elements. Similarly, Darejeh and Singh [42] proposed a list of design guidelines for the Ribbon UI design based on the identified usability issues in literature. The aim is to improve the usability of the Ribbon designs for the end-users with less computer literacy.To the best of our knowledge, our approach examines the violation of an increased number of design guidelines. Besides, our framework provides an appropriate recommendation for each violated guideline and automatically updates the entire MUI tree to include the recommended properties of the corresponding elements. With that being said, our framework also modifies other elements' properties to be harmonious with the changes.

Design similarity: Creating similar models of MUIs can assist designers in quickly formulating design concepts and overcoming existing design difficulties. In a similar vein, Deka et al. [17] mined over 9.7k Android apps to generate five types of data: design search, UI layout generation, UI code generation, user interaction modeling, and user perception prediction. The RICO dataset contains about 72k MUIs, allowing users to search for similar concepts based on the structural composition of the two compared MUIs. Other alternatives constructed the MUI's view hierarchy solely via non-dynamic mining [43–45]. Ulusoya et al. [46] proposed an approach to unify the user interface designs of the same service. Their study aims to present a device-independent UI development approach that generates a user interface design of the same service for different devices to reduce the UI development efforts. Having access to existing models enables the improvement of concepts through the use of best design practices. However, being surrounded by established design principles may limit the designer's creativity. To that end, we aim to provide designers/developers with automatically generated best practices for improving the structural aesthetic of their MUIs.

MUI prototyping: Prototyping or sketching is an iterative process of designing a concept until reaching a compromise between the graphical elements and convey all formal designing standards. It is an early design phase to aid practitioners in creating the appropriate mock-up before the implementation phase. Coyette et al. [15] have proposed the Sketch-iXML rapid prototyping tool to enable designers to sketch MUIs for different contexts of use. Elements of the sketch will transform into specifications to design several concepts until the final MUI design is reached. Kieffer et al. [16] has extended the study to sketch several design ideas, including manually drawn widgets representations with multiple fidelity levels for ease of shape recognition. It is improbable that the proper MUI will be designed from the first implemented version. Furthermore, end-user feedback continues to express dissatisfaction with some issues of the MUI's aesthetic. This work describes an automatic approach for assisting practitioners in repairing various MUI structural aesthetic qualities and improving the application of 13 non-functional design standards.

3. MUIDesigner framework

Redesigning a user interface entails changing various aspects, including structural properties (X, Y, H, W), cohesion and semantic links between elements, etc. In this study, we only cover some structural

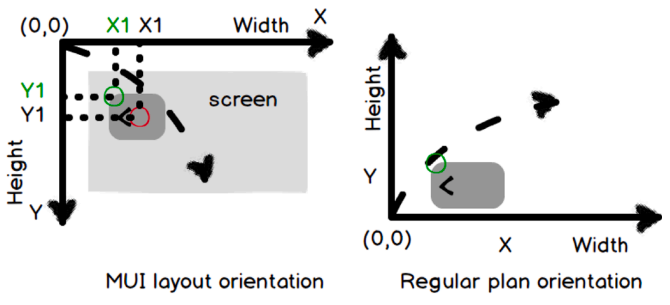


Fig. 1. MUI layout orientation vs Regular plan orientation.

features of redesigning an MUI grid layout. As a result, we define redesigning an MUI as properly providing the dimensions and placements of the graphical elements. The structural restructuring entails changing the X, Y, width, and height of each visual element. In addition, other metadata, such as the X-padding, Y-padding, and the four margins of each component, must be computed to recalculate the structural attributes appropriately.

3.1. Meta data computation

Columns and rows: An MUI layout orientation differs from the regular plan orientation. In a mobile app context, we start filling the layout cells from the very top-left point as shown in Fig. 1. The element with (0,0) coordinates is located in the first {row, column}. To generate the MUI tree, we need to extract the ID of elements found on each row and on each column (the couple {X, Y} represents the ID of each component). It should be noted that the {X, Y} coordinates of each item illustrate the very first top-left starting point of the element shape (green circle) and not the central point (red circle).

- *Row number:* The MUI tree is generated based on the number of rows. We sort the list of the elements' *Y* values from the minimum to maximum. Then, we extract the ID of items that share the same *Y* value.
- *Column number:* We sort the list of the elements' *X* values from the minimum to maximum. Then, we extract the ID of items that share the same *X* value.

Padding: It refers to elements on the same row or column. The X and Y distances help determine whether the same padding is utilized between the components. A considerable variation in padding indicates the presence of a structural defect. [3,4,47].

- *X-distance:* It is calculated between any two adjacent components on the same row, as seen on the left side of Figure 2. Thus, it aids in determining how many various distances exist between the cells. In addition, this structural data assesses the consistency of the elements' placement on the MUI. The following formula is used to calculate the X-distance:
  $x\text{-}distance\ (X_n, X_{n-1}) = X_n\text{-}(X_{n-1} + WidthCmp_{n-1})$.
- *Y-distance:* We compute the white space between every two adjacent elements on the same column for all elements on the Y-axis, as shown on the right side of Fig. 2. The following formula is used to calculate the Y-distance:
  $y\text{-}distance\ (Y_n, Y_{n-1}) = Y_n\text{-}(Y_{n-1} + HeightCmp_{n-1})$.

Margins: The margins represent the distance between the first and last elements on rows and columns, considering the frame resolution. Thus, each margin considers a specific set of components.

- *Margin Left Distance (MLD: first component on each row):* it computes how distant is the first x-point on each row from the Y-axis. MLD is

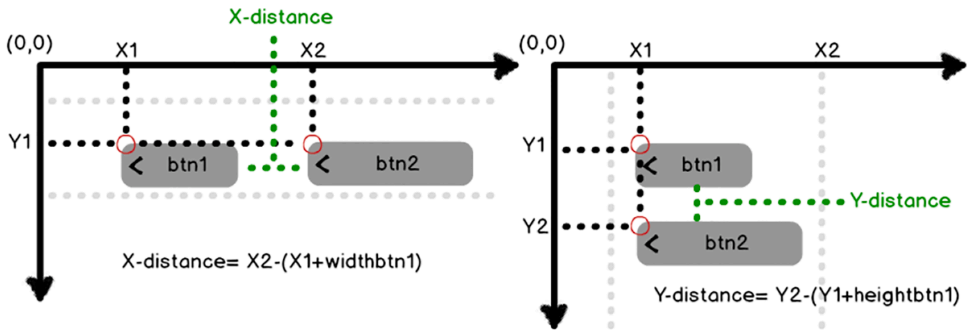


Fig. 2. Padding computation in rows and columns.

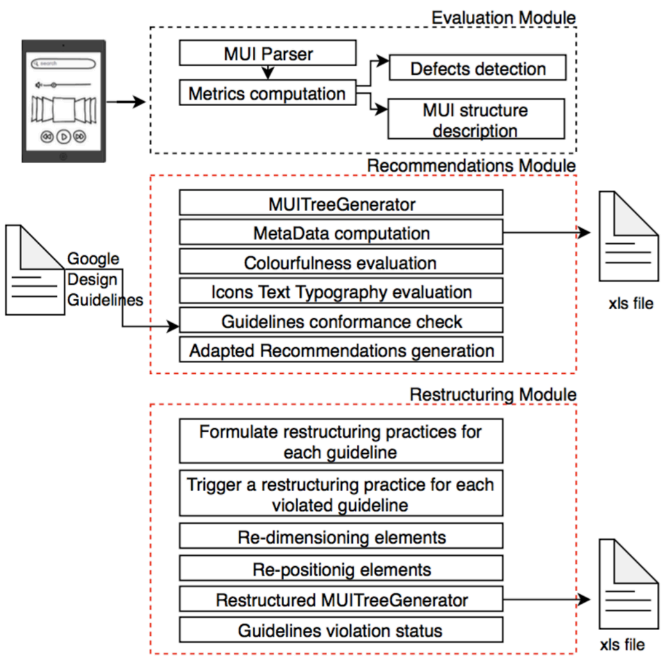

Evaluation Module
MUI Parser
Metrics computation
Defects detection
MUI structure description
Recommendations Module
MUITreeGenerator
MetaData computation
Colourfulness evaluation
Icons Text Typography evaluation
Guidelines conformance check
Adapted Recommendations generation
Google Design Guidelines
xls file
Restructuring Module
Formulate restructuring practices for each guideline
Trigger a restructuring practice for each violated guideline
Re-dimensioning elements
Re-positionig elements
Restructured MUITreeGenerator
Guidelines violation status
xls file

Fig. 3. The MUIDesigner framework modules.

computed as follows:

MLD_row_i=X1_row_i-0

- *Margin Right Distance (MRD: last component on each row):* it computes the distance between the last x-point on each row and the frame width. MRD is computed as follows:

  MRD_row_i=Framewidth- (X_n+width_n)

  with: n = number of last widget on the row.
- *Margin Bottom Distance (MBD: last component on each column):* it computes the distance between the last y-point on each column and the frame height. MBD is computed as follows:

  MBD_column_i=Frameheight- (Y_n+height_n)

  with: n = number of last widget on the column.
- *Margin Top Distance (MTD: first component on each column):* it computes how distant is the first y-point on each column from the X axis. The MTD is computed as follows:

  MTD_column_i=Y1_column_i-0

Note: The last x-point or last y-point represents the widget's top-left x-point (X_n) plus the widget's width (width_n) and top-left y-point (Y_n) plus the widget's height (height_n), respectively.

3.2. Framework architecture

Our framework comprises three basic modules, as shown in Figure 3.

Evaluation: The first module concerns evaluating the MUI, which evaluates 15 quality metrics and detects five structural aesthetic defects. The evaluation model uses the tool ADDET [19]. First, the user is required to extract and attach the MUI dump file[1]. Second, the user should enter the evaluated MUI frame and layout resolutions extracted from the dump file. In addition, the MUI image is necessary to assess the design's colorfulness. Following the computation of metrics and the detection of defects, the framework gives information on structural properties, such as element number repartition, element weight repartition, different element sizes, number of rows and columns, etc.

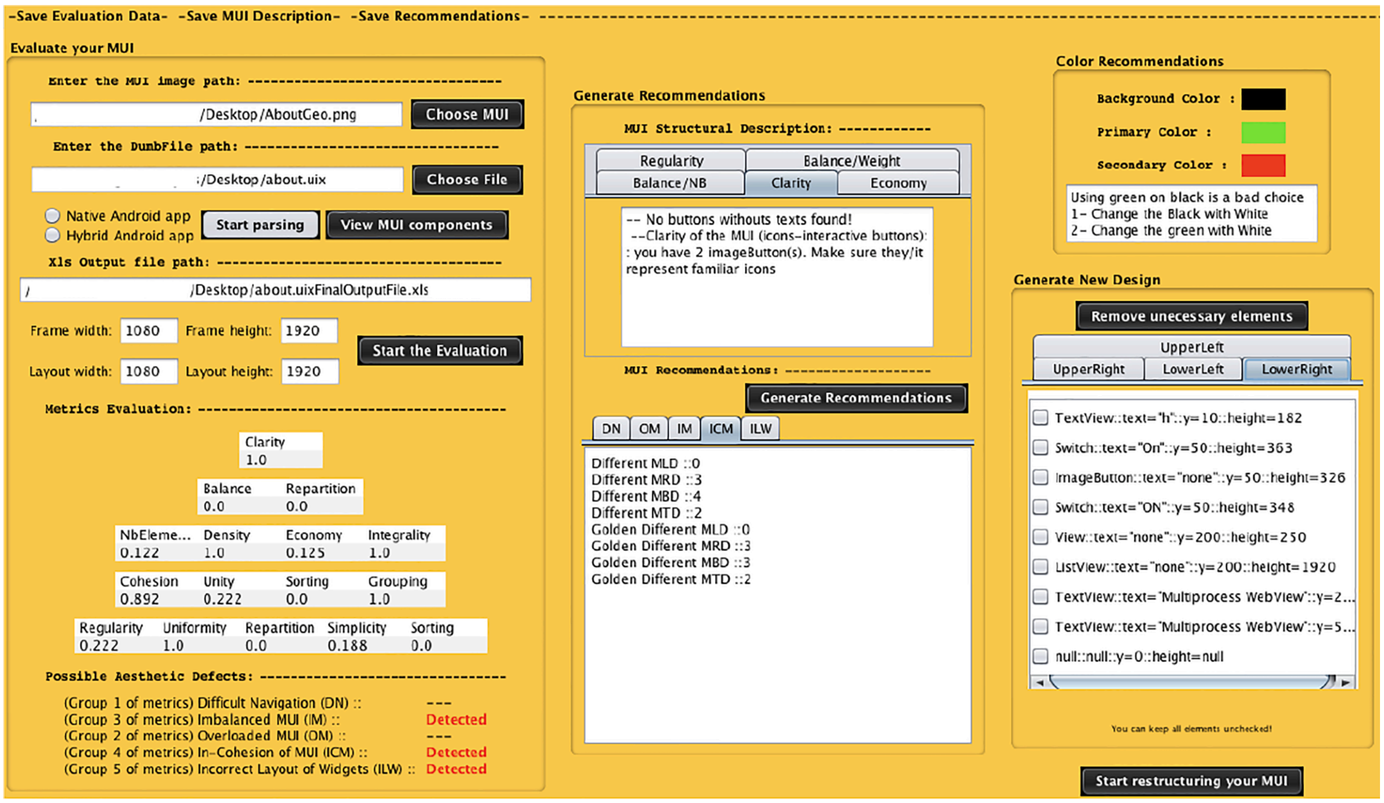
Restructuring: The restructuring approach evaluates the violation of design standards collected from the Google website for Android user interfaces design. A design guideline is a principle that provides the best dimensions of a specific design feature [48]. These guidelines are meant not to violate some quality measures such as accessibility and usability. We found two types of guidelines:

- *Non-Functional:* It is defined as a statement not related to the functionality of a given app, but it is indirectly associated with the usability aspect of an app. It is not possible to map these guidelines to code-level implementation. Let us consider a non-functional guideline from Google's guidelines *"Label button in a descriptive way"*. Checking conformance to such instruction is difficult via a tool and requires manual validation.
- *Functional:*It is defined based on behavior associated with the usability aspect, which can directly map to code-level implementation. Let us consider an applicable guideline from Google's guidelines *"Every TextField should have a label"*. This guideline can be achieved by looking at the element of type *TextField* with a non-null *content-desc* property.

The 13 non-functional design guidelines that we selected from Google material design official website[2] are in concordance with the quality metrics defined in the tool ADDET. We have chosen a set of guidelines to fix by modifying the structural properties or variables defined in the metrics formulas. Since ADDET, to the best of our knowledge, implements the highest number of quality metrics, we believe our approach covers the highest number of structural guidelines. Overall, we propose recommendations on how to address the violated guidelines. Before launching the restructuring module, we assess the clutter of the MUI and provide the user with the repartition of the elements on the four quadrants. Then, request the user to remove any unnecessary items manually via checking buttons. The checked elements will be automatically removed from the layout tree.

[1] A dump file with the extension (uix) is used to save the debuggable application's UI layout tree. It includes the MUI tree with all of the geometrical data of each element, as well as other properties such as *text, disabled, clickable*

[2] https://material.io/design, and https://learnui.design/blog/the-hsb-color-system-practicioners-primer.html.



```
Enter the MUI image path: ----------------------------------
/Desktop/AboutGeo.png
Enter the DumbFile path: ----------------------------------
:/Desktop/about.uix
Xls Output file path: ---------------------------------------
/Desktop/about.uixFinalOutputFile.xls
Frame width: 1080    Frame height: 1920
Layout width: 1080   Layout height: 1920
Metrics Evaluation: ----------------------------------------
```

Clarity 1.0				
Balance 0.0	Repartition 0.0			
NbEleme... 0.122	Density 1.0	Economy 0.125	Integrality 1.0	
Cohesion 0.892	Unity 0.222	Sorting 0.0	Grouping 1.0	
Regularity 0.222	Uniformity 1.0	Repartition 0.0	Simplicity 0.188	Sorting 0.0

```
Possible Aesthetic Defects: --------------------------------
(Group 1 of metrics) Difficult Navigation (DN) ::          ---
(Group 3 of metrics) Imbalanced MUI (IM) ::                Detected
(Group 2 of metrics) Overloaded MUI (OM) ::                ---
(Group 4 of metrics) In-Cohesion of MUI (ICM) ::           Detected
(Group 5 of metrics) Incorrect Layout of Widgets (ILW) ::  Detected
```

```
MUI Structural Description: ------------
-- No buttons withouts texts found!
 --Clarity of the MUI (icons-interactive buttons):
: you have 2 imageButton(s). Make sure they/it
represent familiar icons

MUI Recommendations: -------------------
Different MLD ::0
Different MRD ::3
Different MBD ::4
Different MTD ::2
Golden Different MLD ::0
Golden Different MRD ::3
Golden Different MBD ::3
Golden Different MTD ::2
```

```
Background Color : ■
Primary Color : 🟩
Secondary Color : 🟥
Using green on black is a bad choice
1- Change the Black with White
2- Change the green with White
```

```
☐ TextView::text="h"::y=10::height=182
☐ Switch::text="On"::y=50::height=363
☐ ImageButton::text="none"::y=50::height=326
☐ Switch::text="ON"::y=50::height=348
☐ View::text="none"::y=200::height=250
☐ ListView::text="none"::y=200::height=1920
☐ TextView::text="Multiprocess WebView"::y=2...
☐ TextView::text="Multiprocess WebView"::y=5...
☐ null::null::y=0::height=null
You can keep all elements unchecked!
```

Fig. 4. Main framework window.

Recommendation: Adapted recommendations are generated during the MUI restructuring. On the one hand, some restructuring practices are given in a textual form, requiring a manual application. Let us consider a guideline that cannot be automatically restructured: *"Do not truncate icons labels text"*. A possible recommendation to such a guideline is *"There is only 1 ImageButton that truncates text. ImageButon n#2: width=168dp, text=210dp. Consider removing seven characters"*. It is up to the developer/designer to resize the icon's label. On the other hand, some guidelines are automatically repaired, such as the wrong dimensions of elements.

Figure 4 presents the main window of the MUIDesigner. The restructuring results are provided in a separate window, including the initial MUI tree and the restructured MUI tree. Our framework is publicly available for extension purposes along with an illustrative demo [[51]] .

3.3. Evaluation module

We have chosen the tool ADDET [19] among the existing MUI aesthetic evaluation tools [30,32] as ADDET evaluates a higher number of structural aesthetic characteristics and detects five structural aesthetic defects. The tool also has a high precision of 71%, making it a reliable tool for evaluating the structural aspects of user interfaces. ADDET evaluates 15 aesthetic metrics collected from state-of-the-art studies [4,30,31]. The tool also detects the violation of five structural defects. 1) *Layout defects:* It is the type of defects that consider the structure of the layout, the existence of empty cells, the negative space between the widgets, and the margins distances. This type of defects includes:

- Imbalance of MUI (IM): It represents the unequal distribution of the elements on the four parts of an MUI layout.
- InCohesion of MUI (ICM): The lack of inter-relatedness between the MUI elements is lacking.
- Incorrect Layout of Widgets (ILW): From an end-user perspective, when the elements have a high number of different layouts, the ILW defect is detected. That is, the elements are placed in different rows and columns.

The second type of defect corresponds to 2) *Visual defects:* It concerns the defects that are not related to components positioning problems but rather to visual defects. This type encompasses two defects:

- Overloaded MUI (OM): From an end-user point of view, the overloaded MUI defect is detected when the user sees a high number of elements on the MUI.
- Difficult Navigation (DN): It entails the lack of descriptive labels associated with elements to describe its functionality.

3.4. Restructuring module

All of our chosen design guidelines, as well as their descriptions, adapted recommendations, and restructuring practices, are thoroughly detailed in Tables A.1 and A.2 in Appendix A. In the following subsections, we describe the guidelines that need significant setting.

3.4.1. Difficult Navigation restructuring practices

Two aspects are evaluated to aid in improving the MUI difficult navigation problem.

ColorfulnessThe two notable features that invoke a user's first impression of visual perception are colorfulness, and layout complexity [39]. We perceive the colorfulness of an MUI based on the HSB model attributes: Hue (the purity of color), Saturation (the intensity of a color), and Brightness (the visually perceived brightness). First, we extract all pixel's RGB (red, green, blue) colors. Then, we assess the HSB degree for all neighboring colors. For example, two different RGB color shades with an HSB degree between 41 and 55 invokes a yellow color with an HSB model of (60, 100, 100). This degree means that the area of the MUI having these pixels is quite bright. Colors perception is dependent on how colors shades are distributed on the MUI. Hence, there must be harmony among colors shades in adjacent cells. Zen et al. have evaluated the aesthetic of websites, and mobile apps user interfaces, including the colors palette [32]. However, the authors did not indicate the usefulness or the impact of these colors on the aesthetic design. Since RGB colors cannot indicate how harmonically colorful a MUI is, we evaluate the colorfulness of the user interfaces using the golden rule (6:3:1) or (60% + 30% + 10%). This rule uses only three primary colors, with 60% of the screen is filled by the first color, 30% by the second color, and only 10% by the third color. We applied this rule to the HSB model, assessing the top three colorful colors. We recall that an X color means different shades of colors included in the same HSB degree. This principle will allow user interfaces to reach a sorted balance among colors. Our framework supports 15 colors of the HSB model, with 256*256*256 shades in the RGB model. Therefore, we formulate two kinds of recommendations for the colorfulness of the MUIs.

1. We confirm the usage of the three brightest colors to the (6:3:1) rule. If the percentages are not respected, we provide the developers/designers with the number of pixels added to or removed from each color. Let's consider the case of: (white) 61%, (red) 18%, (black) 11%, (yellow) 6%, (green) 2%, (beige) 2%. The recommendation for such a situation would be: "*The secondary color (red) should be increased by 12%, which represents 3014 pixels. The following conditions must be met: white= − 1%, red=+12%, black= − 1%*". Following this recommendation, developers and designers should reduce the whitest area of the MUI by − 1, the reddish area should include more colors, and should reduce the darkest area.
2. We evaluate the contrast of the top two brightest colors based on the vibrating color combinations[3]. For every inappropriate contrast, we give the safest matching color whether by changing the primary or the secondary color. A recommendation example will be "*Using green on black is a bad choice.*
   1) change the black with white.
   2) change the green with white".

Typography This guideline is raised by the Visual Clarity (VC2) metric in Table A.1. It considers the icons labels to ensure the non-violation of the guideline: "*Do not truncate icons labels text*". An MUI is conceived using the DP unit. Hence, to compute how much a character is in the DP unit, we need the font family, font size, and screen dpi, which are automatically collected from the dump file. Then, we convert each text character to DP and compare the icon text width with the ImageButton width. We formulate the recommendation for such a guideline as follows: "*There is only 1 ImageButton that truncates text. ImageButon n#2: width= 168 dp, text= 210 dp. Consider removing 7.*

3.4.2. Overloaded MUI restructuring practices

To reduce MUI clutter, we target the restructuring of two properties.

A high number of elements. Cluttering user interfaces with interactive and non-interactive graphical components will not make task completion any easier. Seffah et al. have highlighted the *Minimal action* usability factor, which consists of not developing long-task models to reduce task-time completion in fewer steps [49]. Thus, we should conveniently achieve a trade-off between the number of elements and the number of MUIs for one single task. One solution is to remove all unnecessary items and keep only the most critical informing components. In this framework version, we leave the decision on which elements to withdraw to the end-users. The automatic removal of elements necessitates a study of the MUI's semantics, which is in our future directions. We provide the user with the list of components and their X, Y,

[3] https://webdesign.tutsplus.com/articles/why-you-should-avoid-vibrating-color-combinations–cms-25621

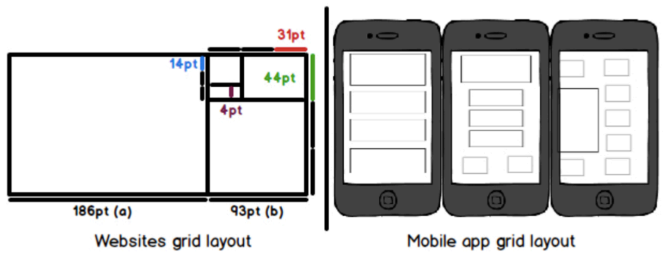


Fig. 5. Golden ratio in Web layout vs mobile apps layouts.

Width, and Height on each of the four quadrants via a checkbox list. The user can repeatedly re-design the MUI by removing and re-adding elements.

A high variation in element sizes. Using many disproportional sizes will not ease the user's eye movement. Besides, the MUI will not look organized nor sequenced. The Golden Ratio (so-called: the Golden Section, Golden Mean, or Divine Proportion) is a measure that correctly computes the proportional dimensions for a given geometrical shape. It is commonly approved that the Golden ratio fosters naturally sorted cuttings that are aesthetically pleasing to the eye.

The golden ratio is mainly used in web and graphic designs to cut the grid into equivalent cells. The golden ratio application results in embedded several boxes creating the shape of a snail's seashell. The process consists of producing a smaller proportional rectangle in every iteration applying Eq. (1).

$$isGolden(h,w) = \sum_{i=1}^{i=NBCmp} \frac{w_i + h_i}{w_i} - \frac{w_i}{h_i} <= epsilon \tag{1}$$

In case the view does not represent a golden rectangle, we compute the proportional height for its width by dividing the width on the ratio of 1.618.

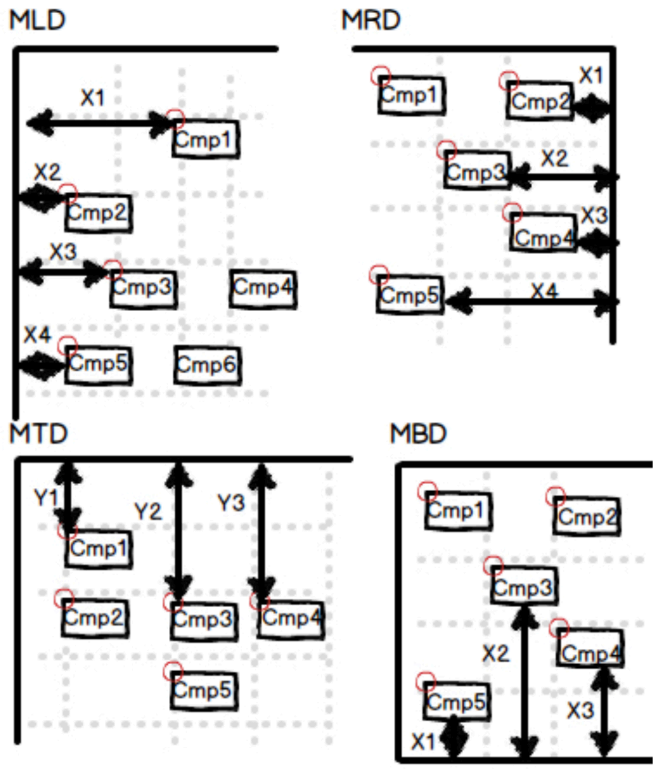


Fig. 6. Margins computation: MLD, MRD, MTD, and MBD.

$$'<'a\begin{cases} = 1 donothing \\ = 0 Golden(w) = \sum_{i=1}^{i=NBCmp} \frac{w_i}{1.618} = h_i \end{cases}$$

The application of the golden ratio in websites differs from that in mobile apps as the grid's structure changes significantly, as illustrated in Figure 5. Since a recursive application of the golden ratio formula is impossible in mobile user interfaces, we apply the golden ratio to shapes. Our goal is to create golden views, which reflect proportional height and width.

We use the golden ratio formula on each view to determine whether it resembles a golden rectangle (Eq. (1)). Otherwise, the golden ratio is used to obtain the golden height and width proportions. We include an epsilon of 0.1 in the evaluation to avoid harsh comparisons.

3.4.3. *Imbalanced MUI restructuring practices*

Bad sorting of elements. We evaluate the consistency of the four different margins of the MUI (left, right, top, bottom) based on the newly created golden proportions to guarantee that the elements are distributed in an ordered and structured manner. Following a seashell form of iterations, we read the first cells of the MUI until we reach the middle cells. Our margins evaluation method begins by reading the MUI from the left side, focusing on the user's cognitive abilities [50]. The margins restructuring approach illustration is available [51].

- *MLD:* We choose the first component on each row with the shortest X distance. Then, as illustrated in Fig. 6, we apply the minimum x distance to the other first components.
  (a) `min(X1,X2,X3,X4)`
  (b) `Apply_min(Cmp1,Cmp2,Cmp3,Cmp5)`
- *MTD:* As shown in Fig. 6, we choose the minimum Y for the first component on each column and adjust the minimum distance to the other components.
  (a) `min(Y1,Y2,Y3)`
  (b) `Apply_min(Cmp1,Cmp3,Cmp4)`
- *MRD:* Among all the components, we choose the smallest distance between the frame width and the ending X point, considering the components' width. We prioritize the MLD if there is just one component on the row as the users' cognition follows the left-right orientation. We compute the MRD as follows.
  (a) $X_i = FrameWidth - WidthCmp_i$
  (b) min(X_i)
  (c) `Apply_min(Cmp2,Cmp3,Cmp4)`
  (d) New$(X_i) = FrameWidth - (WidthCmp_i + min(X_i))$
- *MBD:* Among all the components, we chose the shortest distance between the frame height and the ending Y point, considering the height of the elements. We prioritize the MTD if there is only one component on the column. We compute the MBD as illustrated in Fig. 6.
  (a) $Y_i = FrameHeight - HeightCmp_i$
  (b) min(Y_i)
  (c) `Apply_min(Cmp2,Cmp3,Cmp4)`
  (d) New$(Y_i) = FrameHeight - (HeightCmp_i + min(Y_i))$

3.4.4. *InCohesion of MUI restructuring practices*

No consistent space between the elements and the frame area. Providing consistent spacing between elements is a crucial factor in improving the aesthetic quality of user interfaces. The spacing is represented by the negative or white space that is between the graphical elements. However, determining the right and consistent padding between the elements can be a time-consuming task. The recursive call of the Golden Ratio would create proportional squares where one should place the graphical components. This process ensures to compute the consistent spacing and proportions. However, we cannot create diagrams with the golden ratio in mobile apps, which means no consistent

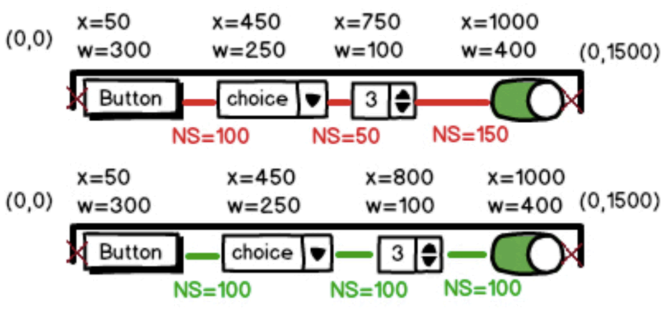


Fig. 7. Computation of the horizontal padding with X_1=0 and X_n=FrameWidth.

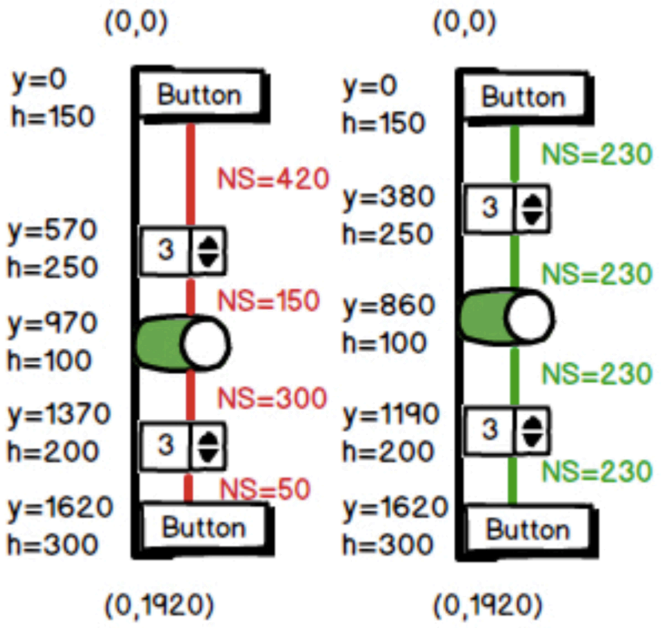


Fig. 8. Computation of the vertical padding with Y_1=0 and Y_n=FrameHeight.

spacing is guaranteed. Therefore, we are proposing our approach to elaborate the same padding between the elements on each row and column and noting that the padding can only be invoked after applying the margins restructuring and the golden ratio in Section 3.4.2.

- *Horizontal Padding*: As illustrated in Fig. 7, we compute the white distance (NS) for all rows having more or equal to three elements. We avoid dealing with two or fewer elements so as not to deteriorate the structure of listview MUIs. We evenly distribute the negative space between the elements considering the very first starting X point, the very last X point of the elements, and the frame width as illustrated in Figure 7. The horizontal padding is computed as follows.
  `Pre-condition: if NbCmp>=3`
  (a) NS= FrameWidth-$\sum(width_i)$
  (b) NS=NS-(X1)
  (c) NS=NS-(FrameWidth-$(X_n + width_n)$)
  (d) NS=$\frac{NS}{NbWidgets-1}$
  Modify X: $X_i + 1 = X_i + Width_i$
  (Note: In case the starting cmp has X!=0 or the ending cmp has X!=FrameWidth, the NS= NS/nbWidgets. In case the starting cmp has X!=0 and the ending cmp has X!=FrameWidth, the NS= NS/nbWidgets+1.)
- *Vertical Padding*: As illustrated in Fig. 8, we compute the white distance for all columns having more or equal to three elements. Then,

we equally distribute the negative space between the elements considering the first starting Y point, the last Y point of components, and the frame height. The vertical padding is computed as follows.

```
Pre-condition: if NbCmp>=3
(a) NS= FrameHeight-$\sum(height_i)$
(b) NS=NS-(Y1)
(c) NS=NS-(FrameHeight-($Y_n + height_n$))
(d) NS=$\frac{NS}{NbWidgets-1}$
    Modify Y: $Y_i + 1 = Y_i + height_i$
    (Note: If the first cmp has Y!=0 or the ending cmp has Y!=FrameHeight, the NS= NS/nbWidgets. In case the starting cmp has Y!=0 and the ending cmp has Y!=FrameHeight, the NS= NS/nbWidgets+1.)
```

In rare circumstances, once the padding is evenly distributed, the first and last components of each {row, column} could not have the same {X, Y} values. Thus, we re-execute the margin computation to guarantee that elements on all sides have the same margin distance. Following this procedure, we recompute the {X, Y} inner padding, which consists of computing the negative space between the first and last elements. In these cases, we do not consider the negative space between (0, $Cmp1_{x/y}$) and ($Cmpn_{x/y}$, FrameHeight/Width).

Different ratios of heights and widths between elements. The application of different sizes increases the complexity of perceiving the MUI. Based on the evaluation of 511 MUIs, we computed the number of different sizes after applying the golden ratio. Then, we considered the found average (12) of varying element sizes as the threshold. If we see more than 12 different sizes of elements in an MUI, we trigger a recommendation on how to reduce and make the element sizes as consistent s possible.

4. Validation

To determine the efficiency of our framework in improving the structural design of user interfaces, we used the tool ADDET [19] to evaluate and compare the aesthetic quality of original and restructured MUIs. We followed a four-step evaluation approach.

1. We evaluate the aesthetic metrics scores and detect the structural aesthetic defects of the original MUI by the tool ADDET;
2. We restructure the MUIs with our framework and generate a new restructured MUI tree;
3. We evaluate the restructured tree with the tool ADDET in terms of the metrics scores and the number of defects.

4.1. Research Questions

Our evaluation consists of satisfying two preliminary and research questions.

PQ1: To what extent have the structural properties of the MUI been improved? To satisfy this question, we considered eight structural attributes representing our restructuring approach's main structural aspects (#Rows, #Columns, #X_padding, #Y_padding, #MLD, #MRD, #MTD, #MBD). A score variable is computed for the original and restructured MUIs. Having many different values of a structural attribute means the MUI suffers from non-uniformity and in-cohesion in its structural grid cells properties (e.g., having different padding among elements on the same row) [4]. The closer the score is to 1, the better is the structure of the MUI. Our score variable is computed as follows:

$$Score = \frac{\#rows + \#columns + \#X\_padding + \#Y\_padding + \#MLD + \#MRD + \#MTD + \#MBD}{8} \tag{2}$$

Weight of each quadrant. After applying the golden ratio to the four-quadrant elements, we compute the total weight of components on each quadrant. A description of such a guideline could be *"Your MUI has a low to heavy widgets weight. Perfectly sorted out"* or *"The bigger weight is located in the middle of the MUI. Therefore, the topper and bottom sides are lighter in weight"*.

3.4.5. ILW restructuring practices

Simplicity: number of rows/columns. We check the number of rows and columns in the layout grid. Then, we provide the users with the coordination of the empty cells between every two adjacent cells on each row and column.

Visual layout consistency: Icons sizes. We search for icons (ImageButtons) that do not incorporate a text label (null "text" attribute). Then, we change their size to 36*72 pixels. The size of icons with text can be larger to fit the text label.

3.5. Recommendations module

The recommendations are a collection of non-functional restructuring practices. To generate adapted recommendations to each violated structural property, we examined the structural parameter of the metrics related to each defect and linked it to the appropriate guideline. A CSV file is generated, including the new MUI restructured tree and the new structural attributes (highlighted in red) for each element that violates a guideline. Please refer to Tables A.1 and A.2 in Appendix A for a detailed description of the violated guidelines recommendations.

The score is computed using the original and the restructured versions of 511 MUIs from 33 applications. For each app, we compute the global average of the accumulated scores values.

PQ2: What type of layouts were most improved by the restructuring approach? As presented in Fig. 5, Android applications MUIs might be conceived with different layouts. The choice of the layout is entirely dependent on the developer/designer's self-view on how to design the user interface. All layouts could be nested and incorporate different layouts into one MUI. We claim that the arrangement of layout cells could impact the efficiency of our restructuring approach. This research question aims to determine which layouts' combinations were positively improved by our restructuring approach and delivered better metrics values.

RQ1: To what extent have the structural aesthetic metrics been improved? The goal behind this question is to examine the impact of our restructuring approach on the quality of 12 structural aesthetic metrics. We evaluate the structural quality of the original and restructured version of the MUIs. We count the number of metrics that do not violate the correspondent structural property for the original and restructured version. Then, we perform the independent *t*-test to examine the difference of metrics means between the two MUIs populations. We test two hypotheses depending on the nature of the metric. We indicate a population (A') of restructured MUIs representing the metrics assumed to be greater in means and a population (B') of metrics with lower means. We indicate the hypothesis test (hypo-test) for each metric with the label `greater or less`. The *Clarity* and *Number of elements* metrics are not considered in the test as they require manual interference to be improved.

Table 1
Android applications involved in the study.

Category	Application	Nb-raters	User Rating	# of MUIs
Social	Instagram	+84.9m	4.5	58
	Facebook lite	+11.6m	4.3	34
	Pinterest	+5.3m	4.6	11
News & Magazines	Twitter	+13.4m	4.3	30
Communication	Gmail	+5.9m	4.3	9
	Messenger lite	+2.7m	4.4	19
	Skype	+10.8m	4.1	12
	Opera Mini	+6.1m	4.5	15
	Tinder	+3.4m	4.0	14
Tools	Google Translate	+6.4m	4.4	8
	Screen Recorder	+36k	4.7	5
	Clean Master	+44.4m	4.7	14
Books	AnyBooks	+104k	4.8	14
	Wattpad	+3.3m	4.6	20
Business	Business Card Maker	+50k	4.5	6
	Glassdoor	+142k	4.5	7
Productivity	Evernote	+1.5m	4.5	22
	WPS office	+1.7m	4.5	7
	Dropbox	+1.9m	4.4	10
Personalization	ZEDGE	+7.4m	4.6	14
Weather	Weather forecast	+388k	4.7	5
	AccuWeather	+2.3m	4.4	13
Sports	beIN SPORTS	+61k	4.4	12
	BeSoccer	+184k	4.5	16
Music&Audio	Shazam	+3.5m	4.4	11
	Anghami	+866k	4.5	31
	Spotify	+15.6m	4.6	8
Education	Duolingo	+7.8m	4.7	15
	TED	+192k	4.6	9
	Lumosity	+239k	4.2	18
	Math Tricks	+369k	4.5	11
Art&Design	Canva	+1.4m	4.7	15
	Floor Plan Creator	+47k	4.1	4
Total				511

- H1: The mean of the metrics of the restructured MUI of the population (A') are greater than population (A).
- H2: The mean of the metrics of the restructured MUI of the population (B') are greater than population (B).

RQ2: To what extent has the structural aesthetic defects number been reduced? The number of structural aesthetic defects is tightly related to the metrics values. Therefore, based on the results found in RQ1, we examine whether the number of defects has decreased, increased, or remained the same.

4.2. Selection of applications

We used a three-step selection procedure to assess the efficiency of our proposed restructuring approach. To guarantee that the apps exhibit well-engineered and user-satisfying projects, we first selected and downloaded 50 random applications from the Google Play store with a user rating $>=$ 4.0. Then, in each application, we went through several user interfaces to find the apps that best implemented Google Material Design. Afterward, we sorted the list of apps by the number of raters in the Google Play store. Then, we re-evaluated the apps, and around 500 MUIs were chosen. The majority of the selected MUIs represent the primary page of each activity in the application. We provide in Table 1 the characterisrics of the selected applications. We evaluated the MUIs on a frame resolution equal to 1280 x 720 portrait and 720*1280 landscape. Different frame resolutions will not alter the metrics values as we are dealing with adaptive user interfaces.

Table 2
Score variable results for the original and restructured MUIs.

App name	Nb MUIs	Old MUIs score	New MUIs score	Diff
Instagram	58	8.084	6.675	+1.409
Facebook lite	34	12.935	10.420	+2.515
Pinterest	11	10.170	7.954	+2.216
Twitter	30	9.181	7.633	+1.548
Gmail	9	9.375	7	+2.375
Messenger lite	19	11.458	7.583	+3.875
Skype	12	13	10.187	+2.813
Opera Mini	15	11.266	9.333	+1.933
Tinder	14	6.419	4.803	+1.616
Google translate	8	8.937	7.375	+1.562
Screen Recorder	5	9.325	6.6	+2.725
Clean Master	14	13.169	9.044	+4.125
AnyBooks	14	20.669	14.25	+6.1
Wattpad	20	9.181	7.168	+2.013
Business Card Maker	6	8.729	7.166	+1.563
Glassdoor	7	7.357	6.589	+0.786
Evernote	22	7.845	5.839	+2.006
WPS office	7	11.053	8.285	+2.768
Dropbox	10	7.675	6.662	+1.013
ZEDGE	14	11.471	9.201	+2.27
Weather forecast	5	21.9	18.25	+3.65
AccuWeather	13	9.259	5.884	+3.38
beIN SPORTS	12	15.885	12.916	+2.969
BeSoccer	16	17.843	15.210	+2.633
Shazam	11	7.579	5.943	+1.636
Anghami	31	9.566	7.275	+2.27
Spotify	8	8.078	6.968	+1.102
Duolingo	15	10.258	7.8	+2.458
TED	9	10.486	8.861	+1.625
Lumosity	18	6.236	4.770	+1.466
Math Tricks	11	7.693	5.931	+2.032
Canva	15	13.116	10.558	+2.558
Floor Plan Creator	4	12.312	11.218	+1.094

*Diff=Difference

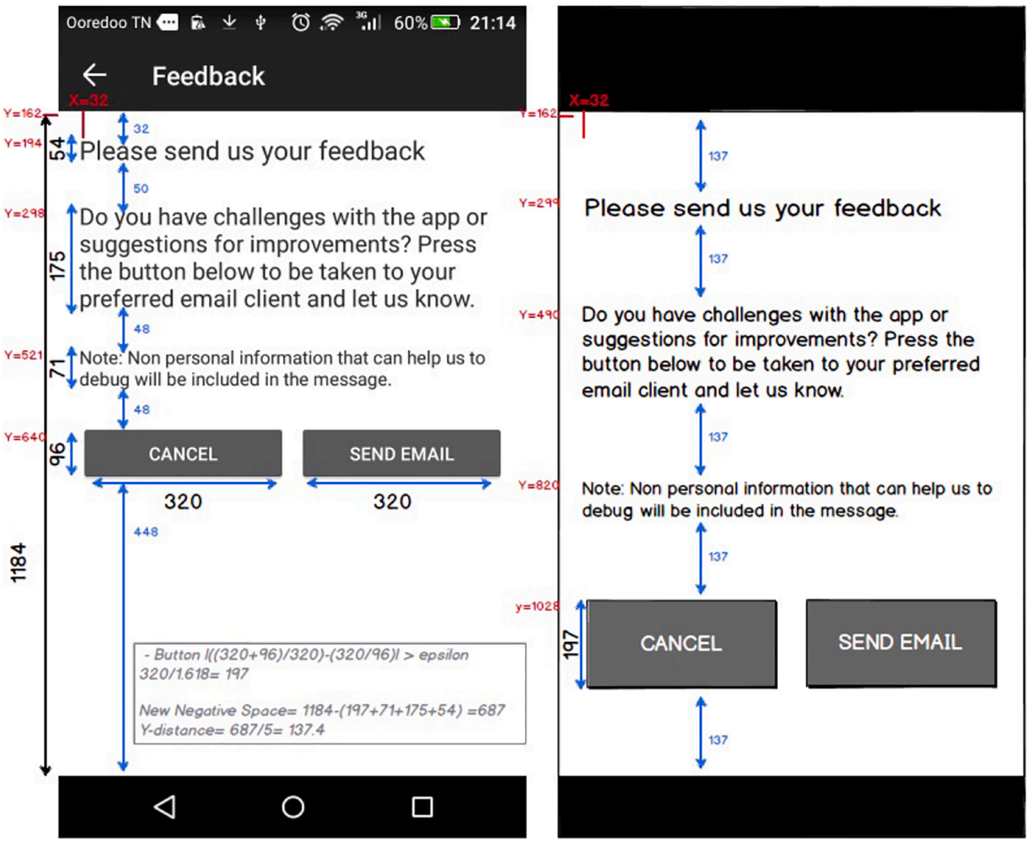
Table 3
Impact of the restructuring approach on the different types of layouts combinations.

Layouts	#	Combinations	#MUIs	Attributes
LinearLayout(LL)	4097	(LL,FR)	157	7.71
FrameLayout(FL)	3795	(LV,LL,FR)	69	7.53
RelativeLayout(RL)	1302	(LL,FR,RL)	206	8.52
GridView(GV)	3	(GV,LL,FR)	2	**3.31**
ListView(LV)	230	(GV,LL,FL,RL)	1	**6.34**
		(LV,LL,FR,RL)	76	9.33

4.3. Results and Discussion

PQ1: To what extent have the structural properties of the MUI been improved? In Table 2, we provide the accumulated *score* values for each application. The highest difference in score average is seen in the AnyBooks app, equal to 6.1. This latter implements most of its MUIs as list views, which increases the chance of having different right margins. All the other applications have a small score difference of 1 to 3, indicating that few structural attributes have been restructured. The low average score reflects how well-designed these apps are. Alternatively, the restructuring strategy has aided in the better restructuring of some structural attributes of these MUIs. The Y_padding, MLD, and the number of columns are the top three restructured characteristics. The MRD was ineffective in MUIs where some rows had $<$ 3 elements, such as in the BusinessCardMaker and Canva applications. As a result, only rows with $>=$ 3 elements will be restructured.

PQ2: What type of layouts were most improved by the restructuring approach?



Please send us your feedback

Do you have challenges with the app or suggestions for improvements? Press the button below to be taken to your preferred email client and let us know.

Note: Non personal information that can help us to debug will be included in the message.

- Button l((320+96)/320)-(320/96)l > epsilon
320/1.618= 197

New Negative Space= 1184-(197+71+175+54) =687
Y-distance= 687/5= 137.4

Please send us your feedback

Do you have challenges with the app or suggestions for improvements? Press the button below to be taken to your preferred email client and let us know.

Note: Non personal information that can help us to debug will be included in the message.

Fig. 9. Original vs restructured Feedback MUI of the ZEDGE app.

Table 4
Independent *t*-test results with df = 95.

Metrics	meanOld	meanNew	*t*-test	p-val	thresholds	hypo-test
Density	0.701	0.388	– 11.252	**2.2e-16**	0.701	less (H2)
Sequence	0.181	0.145	– 3.83	0.9999	0.181	greater (H1)
Balance	0.413	0.358	– 3.007	0.998	0.414	greater (H1)
Regularity	0.247	0.716	16.986	**2.2e-16**	0.248	greater (H1)
Economy	0.058	0.112	7.720	**5.005e-11**	0.058	greater (H1)
Simplicity	0.527	0.584	3.889	**0.000121**	0.528	greater (H1)
Layout uniformity	0.846	0.715	– 5.328	1	0.846	greater (H1)
Unity	0.262	0.355	15.925	**2.2e-16**	0.262	greater (H1)
Cohesion	0.642	0.678	1.381	**0.085**	0.643	greater (H1)
Integrality	0.862	0.308	– 19.001	1	0.883	greater (H1)
Homogeneity	0.007	0.005	– 0.672	0.748	0.008	greater (H1)
Grouping	0.135	0.192	– 4.847	**4.161e-06**	0.135	greater (H1)
Complexity	0.450	0.379				less (H2)
The complexity is the average of the metrics						

We parsed the dump files looking for different layouts declared in the tag ”class=”android.widget.LAYOUT”. Only five layout types were used in the 511 MUIs we tested (LinearLayout, FrameLayout, GridView, ListView, RelativeLayout). We explored alternative combinations of layouts as MUIs could incorporate different layouts. Then, we compared the mean of the eight structural features (PQ1) across the different MUI layouts. Six layout combinations were discovered with only three MUIs implementing the GridView, as shown in Table 3. In the entirety of the MUIs, the LinearLayout and the FrameLayout are the most commonly utilized layouts. Our restructuring approach had the most significant impact on MUIs with GridView, with means equal to 3.31 and 6.34, respectively. The GridView divides the grid into table cells with regular columns and rows, matching our restructuring method. With a score of 7.53, the LinearLayout, FrameLayout, and ListView tied for third place. These MUIs illustrate scrolling ListViews, with each item having its LinearLayout. A column/row model is used to position the ListViews. Overall, the results for all layout combinations were not that dissimilar. However, the restructuring approach benefited MUIs with a GridView, a ListView, and the Linear and Frame Layouts. As a result, the findings call for more empirical research into the properties of various layout types while designing a systematic MUI restructuring approach.

Fig. 9 represents the original and restructured version of the Feedback MUI of the ZEDGE application. After using the framework to generate the restructured MUI tree, we manually created the restructured MUI’s views to present a real-world illustration of our approach. The old MUI suffers mainly from the IM (Imbalanced MUI) defect due to inconsistent vertical padding. After the application of the restructuring approach, the same Y-padding is employed between every two adjacent

Table 5
Results of the effect size between our metrics populations of original and restructured MUIs.

Metrics	effect size		
	d	CI	
Density	−**2.770** (large)	2.081	3.458
Sequence	0.944 (large)	0.426	1.463
Balance	0.740 (medium)	0.232	1.248
Regularity	−**4.181** (large)	− 5.059	− 3.303
Economy	−**1.900** (large)	− 2.493	− 1.308
Simplicity	−**0.957** (large)	− 1.476	− 0.438
Layout complexity	−**0.135** (negligible)	− 0.627	0.357
Layout uniformity	1.312 (large)	0.770	1.854
Unity	−**3.920** (large)	− 4.760	− 3.079
Cohesion	− **0.340**(small)	− 0.835	0.155
Integrality	4.677 (large)	3.727	5.628
Homogeneity	0.165 (negligible)	− 0.327	0.658
Grouping	−**1.193** (large)	0.659	1.727

cells. The golden ratio proposed a 320*197 instead of 320*96 for the buttons, which eases the user's interaction with the elements. The margins approach has resulted in no modifications, as all components share the same X value= 32, and all rows have > 3 elements. Overall, the MUI has gained more structural balance. The new design structure fosters the designers to improve accessibility by increasing the size of the TextViews.

RQ1: To what extent have the structural aesthetic metrics been improved? In the first step, an independent t-test was performed on the data to compare the means of the metrics values of the original and restructured MUIs with a 95% confidence interval (CI) for the mean difference. In the second step, we computed the effect size to entail the amount of improvement. In Table 4, we indicate the tested hypothesis for each metric. The results reveal that seven out of 12 metrics were improved compared to the thresholds (highlighted in bold). We recall that the clarity and the number of elements metrics have been removed from the test, as we assume their equality. Therefore, the layout complexity metric is removed as it computes the average of the involved 15 metrics. The hypothesis (H2) has been accepted for the density metric with a p-value $<.05$ with a difference of 0.313. The regularity, economy, simplicity, unity, cohesion, and grouping metrics of the restructured MUIs were significantly higher than the original population (A) with p-values $<.05$. The hypothesis is rejected for the sequence, balance, uniformity, integrality, and homogeneity metrics. However, these metrics values have slightly decreased in the restructured versions, which did not much deteriorate the MUI quality. The integrality metric has been highly decreased as it considers two inter-playing structural attributes, the a_{sc} (the cumulative area of components) and a_i (the area of the interactive object i). On average, in all the 511 MUIs, the golden ratio has resulted in smaller dimensions of components. Hence, a lower a_{sc} and a_i values are computed compared to the original MUIs. Thus, an evident decrease in the integrality metric value is expected. However, we have also found that the homogeneity and layout uniformity metrics have not remarkably changed after the restructuring approach. These two measurements examine how evenly elements are distributed across the four quadrants of the MUI. These metrics results indicate that our

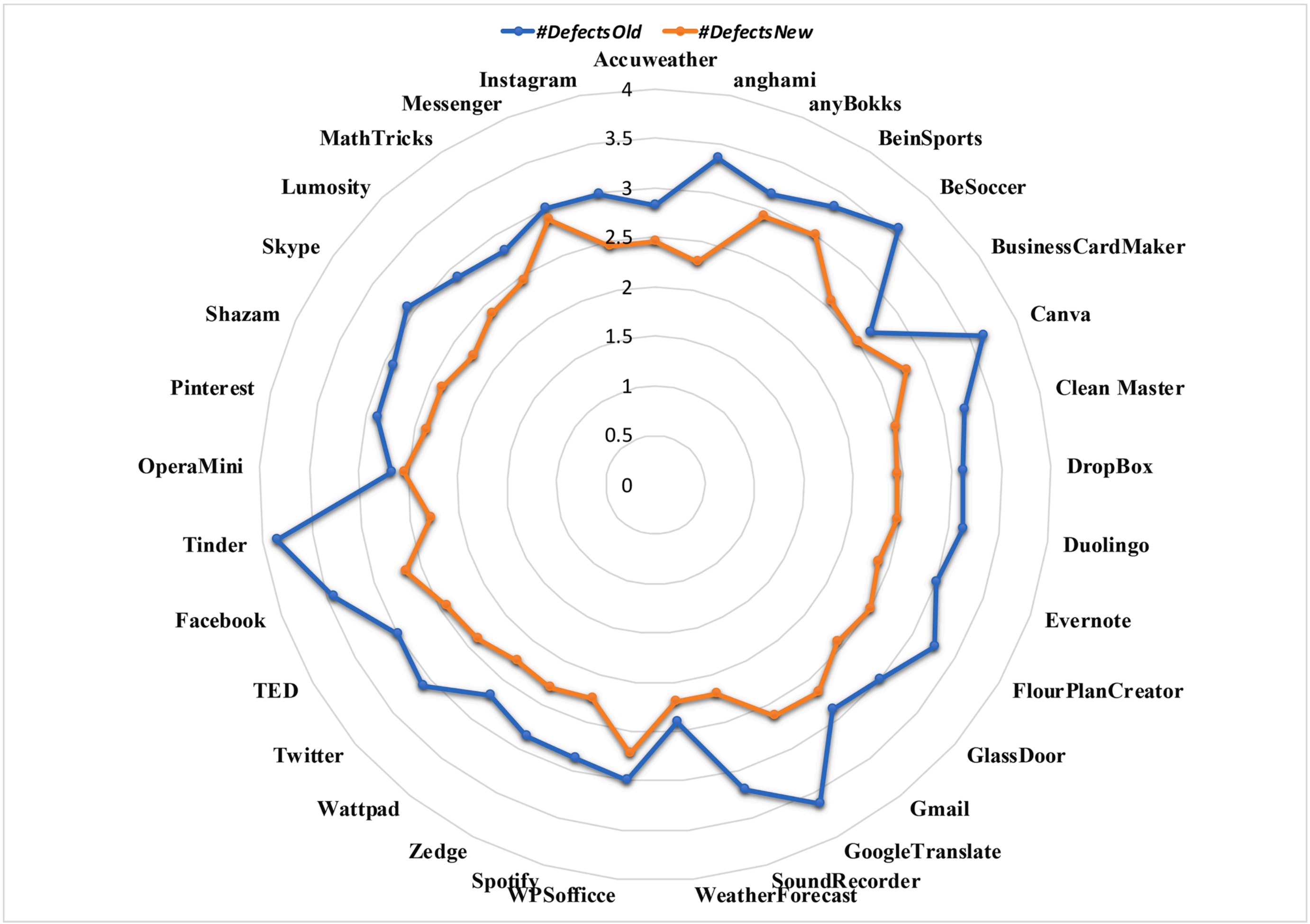


Fig. 10. Defects median number in the Original and Restructured MUIs.

approach kept more or less the same number of elements in each quadrant, guaranteeing that no semantically linked elements have been separated into different quadrants. Thus, we more or less guarantee the non-significant violation of semantics between elements groups that are still not covered in this study.

In the following Table 5, the cohen distance has been computed over the independent *t*-test results between the two populations of metrics. Thus, conducting the cohen-d test allows inspecting the effect size between our two samples (population A and population B) for the 33 apps. As stated by Cohen, a d=0.2 is considered a small effect size, 0.5 represents a medium effect size, and 0.8 is a substantial effect size. A negative cohen-d entails which population is significant. Table 5 presents the realized positive impact of our restructuring approach over the metrics values. A high positive impact (> 0.8) is witnessed in the density, regularity, economy, simplicity, unity, and grouping, indicating a significant positive effect of our approach on the quality level of the metrics and consequently on the MUI quality level. A small effect size (d=> 0.3, CI= − 0.835,0.155) in the cohesion metric indicating a modest improvement in the cohesion level. The overall layout complexity has been improved with a slight increase of 0.135.

RQ2: To what extent has the structural aesthetic defects number been reduced?

The Kiviat chart in Fig. 10 represents the average number of defects in each application in the original MUIs (blue dots) and the restructured MUIs (orange dots) ranging from 0 to 4 defects. We see a decrease in the number of defects in all the applications. Tinder and SoundRecorder apps have witnessed the most noticeable defect reduction from > 3.5 to < 2.5. Some applications have been improved by reducing a single defect, such as OperaMini, MathTricks, WeatherForecast, and BusinessCardMaker. Overall, our restructuring approach has succeeded in reducing the number of structural defects by a 0.59 difference in mean.

5. Threats to Validity

Construct threats to validity concerns errors in measurements. The computation of the quality metrics and the thresholds to detect the aesthetic defects are already validated in the tool ADDET study [19]. We have translated the textual guidelines into mathematical formulas regarding the framework guidelines applications, respecting all the given properties. Furthermore, other principles that do not provide constant measures, such as *"Material Design layouts encourage consistency by using uniform elements and spacing"*, are described in detail and discussed among the authors on how to provide a generic application.

Conclusion threats to validity could be related to the data analysis to draw our conclusions. We chose a set of 511 MUIs from various application domains to evaluate the efficacy of our framework in reforming MUI designs. Our goal was to test our approach on as many different MUI trees and layouts as possible. We discovered promising results based on the examination of the 511 MUIs, suggesting the efficacy of our approach in improving seven aesthetic properties and reducing the number of structural defects. However, with the incorporation of new design guidelines in our future directions, we will need to increase the number of tested samples to generalize our approach's reliability. Another possible threat to validity is using the same tool ADDET to evaluate the original and restructured MUIs. We first used ADDET to assess the original designs. Then, we executed the restructuring tool and the recommender to generate a new re-designed MUI tree. The new MUI design quality is evaluated using ADDET to identify the improved and deteriorated properties compared to the original MUI designs. It is essential to accentuate that ADDET implements the highest number of quality metrics to the best of our knowledge. The only existing tool that identifies aesthetic defects [30] over-computes the number of visual widgets (considers the layouts as visual elements), which could lead to incorrect evaluation.

Internal threats to validity concern the factors that could restrict the applicability of our observations or affect our conclusions. Our approach does not provide a complete MUI design restructuring solution. Other design principles are yet unaddressed in this first version of our framework. Some elements may, for example, be placed in a semantically incorrect cell. Independent of the cohesion and semantic order of the components, our restructuring approach gives the ideal grid cell positioning in the function of the newly created widgets sizes. As a result, it is up to the designers to decide which element goes in which cell. Another possible internal threat is that our *score* metric gives a preliminary idea of the quality of an MUI using only eight structural properties. Other structural aspects such as the padding between the elements groups layouts are not yet considered. We have selected the main structural properties included in the aesthetic metrics in the tool ADDET that our approach aims to adjust. Other structural properties would be included when new design guidelines are covered.

External threats to validity concern the generalization of our findings. To the best of our knowledge, this study is the first to present an automated method for re-designing MUI graphical elements structurally. However, other non-functional design requirements, such as accessibility, must still be addressed to provide a complete restructuring strategy.

6. Conclusion

We proposed in this paper the MUIDesigner framework that aids in automatically restructure an Android mobile user interface design and provides adaptive recommendations for issues that necessitate manual interference. Our approach evaluates 15 structural aesthetic properties of the MUI and detects five structural defects. The restructuring module evaluates 13 structural design rules collected from Google's material design guidelines and provides users with the violations. Then, the framework automatically applies the recommendations to fix guidelines violations. We evaluated the efficiency of our approach by considering the old and newly restructuring versions of an MUI tree. The findings indicate that seven quality metrics were improved, and the number of defects has been reduced. Overall, the complexity of the restructured MUI trees was decreased. Thus, to the best of our knowledge, we propose the first automated MUI restructuring approach. Besides, users can see their design restructured with different elements by removing/re-adding items in each chosen quadrant on the MUI. The purpose of any such tool is to assist developers/designers in enhancing the look of the design structure and provide recommendations on how to fix some of the characteristics that cannot be automatically fixed. In our future directions, we plan to 1) empirically investigate the specifications of each layout type on the restructuring of the MUI. 2) The Automatic removal of elements by analyzing the task dependency between the component and the task using a set of semantic metrics. 3) Considering (0,0) components coordinates and the one component per row for better balance and weight equilibrium. We also plan to 4) consider the case where golden dimensions and the negative space on each row and column exceed the FrameHeight and FrameWidth. Thus, a scale-down approach of widgets' golden sizes would be proposed.

CRediT authorship contribution statement

Narjes Bessghaier: Conceptualization, Formal analysis, Methodology, Validation, Writing – review & editing. **Makram Soui:** Methodology, Supervision, Writing – review & editing. **Nadia Ghaibi:** Supervision, Writing – review & editing.

Declaration of Competing Interest

The authors of this paper confirm that we have no conflicts of interest.

Appendix A. Selected design guidelines

Table A.1
Selected guidelines from Google Material Design.

Defect	Metric	Parameter	Guideline	Description	Practice
DN	Clarity (C1)	Lack of icons labels	*"It's best to pair icons with text labels, especially if the icon does not have obvious meaning"*	Use user-friendly or labeled icons not to confuse the user	Check if ImagesButtons are associated with texts.
	Clarity (C2)	Lack of TextField labels	"*Every TextField should have a label*"	Add labels to interactive components	Look for the coordinates of the widgets that lack textual labels.
	Visual Clarity (VC1)	—	*"Combining bottom navigation and tabs may cause confusion"*	The MUI must incorporate either bottom navigation or tabs	Check if the MUI includes both bottom navigation and tab.
	Visual Clarity (VC2)	—	*"Don't truncate icons labels text"*	keep short and concise	Get icons text width and compare it with the ImageButton width, by converting each character to dp unit based on screen dpi, font family and font size.
	Colorfulness	Number and contrast of colors	*"The Rule of 3 Colors: Choose primary, secondary, and accent colors for your app that support usability"*	Pick one primary color shades and two other complementary colors shades.	The color contrast test is based on the Vibrating-Color Headache combinations.
OM	number of elements	Number of elements	*"The display must be reduced to only the necessary components for the current tasks"*	Remove unnecessary elements	Manual removal: We provide the user with a list of elements types on each quadrant of the MUI to select the elements that could be removed.
	Density	Area of components and frame area	*"Layouts should use a consistent grid, keylines, and padding"*	Add a negative space between the widgets so that the area of components decreases.	Add same spacing among elements located on same row and column.
	Economy	Different used widgets sizes	*"Material Design layouts encourage consistency by using uniform elements and spacing"*	Provide as much as possible consistent elements sizes	The Golden Ratio provides proportional dimensions of elements.
	Integrality	Economy+Density			
IM	Balance	Different sizes of widgets on four quadrants	*"Layouts should be visually balanced"*	Try to equally balance the weight of elements on the four quadrants.	Based on the new generated Golden proportions, we check the consistency of the four different margins to ensure that the group of widgets are in the middle of the MUI.
	Homogeneity	Different Number of components on four quadrants	*"Layouts should be visually balanced"*	Try to equally balance the number of elements on the four quadrants.	1- Look for the number of the components on each quadrant. 2- Manual removal: Provide the user with a list of elements types on each quadrant of the MUI to select the items that could be removed.
ICM	Cohesion	Different ratios of height and width between all elements	*"Layouts should use a consistent grid, keylines, and padding"*	Keep same size ratios among elements	1-Check how many different resolution of elements after the application of the Golden Ratio 2: We computed the average of different sizes for 511 MUIs.
	Sorting	Weight of each quadrant	*"Layouts should be visually balanced"*	Keep same weight balance	Check if the MUI has a descendant or ascendant order of widgets weights (heavy to light/ light to heavy).

Table A.2
Continuity: Selected guidelines from Google Material Design.

Defect	Metric	Parameter	Guideline	Description	Practice
	Unity	Number of different types of widgets	No related guideline found	–	–
	Grouping	Different negative spaces between the widgets	*"Layouts should use a consistent grid, keylines, and padding"*	Keep a cohesive negative space between layout cells	Add uniform padding among the elements of each row and column.
ILW	Regularity	Different distances between rows and columns elements	*"Layouts should use a consistent grid, keylines, and padding"*	Keep a cohesive negative space between layout cells	Add regular padding among the elements of each row and column.
	Layout Uniformity	Different height and width and number of elements on the 4 quadrants	*"Layouts should use a consistent grid, keylines, and padding"*	Keep same weight ratios	1-Check how many different resolution of elements after the application of the Golden Ratio 2: We computed the average of various sizes for 511 MUIs.
	Homogeneity	the Different number of components on four quadrants	*"Layouts should be visually balanced"*	Try to equally balance the number of elements on the four quadrants.	1- Look for the number of the components on each side. 2- Manual removal: Provide the user with a list of elements types on each quadrant of the MUI to select the items that could be removed.
	Simplicity	number of columns and rows	*Alignment: Align the text, images, and buttons to show users how information is related.*	Try to reduce the number of used rows and columns	1- Check how many rows and columns in the layout grid. 2-Provide the user with the coordination of the empty cells on each row by computing the difference between the number of elements on each row.
	Sorting	Weight of each quadrant	*"Layouts should be visually balanced"*	Keep same weight balance	Check if MUI has a descendant or ascendant order of widgets weights (heavy to light/ light to heavy)
	Visual layout consistency	Icons sizes	*"Icons Button Touch target minimum of 36*72 dp"*	Icons should be at least of size 36*72 pixels	We search for icons (ImageButtons) that do not incorporate a text label. These latter represent icons with images and change their size to 36*72 px. Icons with text can be larger to fit the text label.
		Navigation tabs	*"Don't use more than five destinations and fewer than three destinations"*	Try tabs or a navigation drawer	Check if the number of ImagesButton exceeds 5 or less than 3.
		Views sizes	*"Don't shrink text size to fit a text label on a single line"*	All TexViews should be consistently aligned and spaced one of the other	Test if all TextViews have the same width.

References

[1] R. Mahajan, B. Shneiderman, Visual and textual consistency checking tools for graphical user interfaces, IEEE Trans. Softw. Eng. 23 (11) (1997) 722–735.

[2] B. Robinson, P. Brooks, An initial study of customer-reported GUI defects. ICSTW'09. International Conference, IEEE, 2009, pp. 267–274.

[3] D. Ngo, L.S. Teo, J.G. Byrne, Formalising guidelines for the design of screen layouts, Displays 21 (1) (2000) 3–15.

[4] D.C.L. Ngo, A. Samsudin, R. Abdullah, Aesthetic measures for assessing graphic screens, J. Inf. Sci. Eng. 16 (1) (2000) 97–116.

[5] K. Mullet, D. Sano, Designing visual interfaces: communication oriented techniques vol. 2550, SunSoft Press Englewood Cliffs (NJ), 1995.

[6] F. Montero, V. López-Jaquero, Guilayout++: supporting prototype creation and quality evaluation for abstract user interface generation. Workshop of ACM SIGCHI Symposium on Engineering Interactive Computing Systems, Berlin, Germany, 2010.

[7] E.N. Wiebe, A. Lamb, M. Hardy, D. Sharek, Measuring engagement in video game-based environments: investigation of the user engagement scale, Comput. Hum. Behav. 32 (2014) 123–132.

[8] H.L. O'Brien, E.G. Toms, The development and evaluation of a survey to measure user engagement, J. Am. Soc. Inf.Sci. Technol. 61 (1) (2010) 50–69.

[9] H.L. O'Brien, E.G. Toms, What is user engagement? A conceptual framework for defining user engagement with technology, J. Am. Soc. Inf.Sci. Technol. 59 (6) (2008) 938–955.

[10] G.F. Tondello, R.R. Wehbe, R. Orji, G. Ribeiro, L.E. Nacke, A framework and taxonomy of videogame playing preferences. Proceedings of the Annual Symposium on Computer-Human Interaction in Play, ACM, 2017, pp. 329–340.

[11] A. Sonderegger, J. Sauer, The influence of design aesthetics in usability testing: Effects on user performance and perceived usability, Appl. Ergono. 41 (3) (2010) 403–410.

[12] N. Bessghaier, M. Souii, Towards usability evaluation of hybrid mobile user interfaces. Computer Systems and Applications (AICCSA), 2017 IEEE/ACS 14th International Conference on, IEEE, 2017, pp. 895–900.

[13] M. Minge, M. Thüring, Hedonic and pragmatic halo effects at early stages of user experience, Int. J. Hum.-Comput. Stud. 109 (2018) 13–25.

[14] A. Türkyilmaz, S. Kantar, M.E. Bulak, O. Uysal, et al., User experience design: aesthetics or functionality?. Managing Intellectual Capital and Innovation for Sustainable and Inclusive Society: Managing Intellectual Capital and Innovation; Proceedings of the MakeLearn and TIIM Joint International Conference 2015 ToKnowPress, 2015, pp. 559–565.

[15] A. Coyette, J. Vanderdonckt, Q. Limbourg, SketchiXML: a design tool for informal user interface rapid prototyping. International Workshop on Rapid Integration of Software Engineering Techniques, Springer, 2006, pp. 160–176.

[16] S. Kieffer, A. Coyette, J. Vanderdonckt, User interface design by sketching: a complexity analysis of widget representations. Proceedings of the 2nd ACM SIGCHI Symposium on Engineering Interactive Computing Systems, ACM, 2010, pp. 57–66.

[17] B. Deka, Z. Huang, C. Franzen, J. Hibschman, D. Afergan, Y. Li, J. Nichols, R. Kumar, Rico: a mobile app dataset for building data-driven design applications. Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology, ACM, 2017, pp. 845–854.

[18] G. Ines, S. Makram, C. Mabrouka, A. Mourad, Evaluation of mobile interfaces as an optimization problem, Procedia Comput. Sci. 112 (2017) 235–248.

[19] N. Bessghaier, M. Soui, C. Kolski, M. Chouchane, On the detection of structural aesthetic defects of android mobile user interfaces with a metrics-based tool, ACM Trans. Interact. Intell. Syst.(TiiS) 11 (1) (2021) 1–27.

[20] N. Mathur, S.A. Karre, Y.R. Reddy, Usability evaluation framework for mobile apps using code analysis. Proceedings of the 22nd International Conference on Evaluation and Assessment in Software Engineering 2018, ACM, 2018, pp. 187–192.

[21] J. Kjeldskov, M.B. Skov, Creating realistic laboratory settings: comparative studies of three think-aloud usability evaluations of a mobile system. Proceedings of the 9th IFIP TC13 International Conference on Human-Computer Interaction, 2003, pp. 663–670.

[22] E.B. Ayed, C. Kolski, R. Magdich, H. Ezzedine, Towards a context based evaluation support system for quality in use assessment of mobile systems. Systems, Man, and Cybernetics (SMC), 2016 IEEE International Conference on, IEEE, 2016, pp. 004350–004355.

[23] Y.S. Ryu. Development of usability questionnaires for electronic mobile products and decision making methods, Virginia Tech, 2005. Ph.D. thesis.

[24] R. Yáñez Gómez, D. Cascado Caballero, J.-L. Sevillano, Heuristic evaluation on mobile interfaces: a new checklist, Sci. World J. 2014 (2014).

[25] L. Kuparinen, J. Silvennoinen, H. Isomäki, Introducing usability heuristics for mobile map applications. Proceedings of the 26th International Cartographic Conference, August 25 30, 2013, Dresden, Germany, ISBN 978-1-907075-06-3, International Cartographic Association, 2013.

[26] J. Nielsen, R. Molich, Heuristic evaluation of user interfaces. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, ACM, 1990, pp. 249–256.

[27] D. Quiñones, C. Rusu, V. Rusu, A methodology to develop usability/user experience heuristics, Comput. Stand. Interfaces 59 (2018) 109–129.

[28] A. Riegler, C. Holzmann, Measuring visual user interface complexity of mobile applications with metrics, Interact. Comput. 30 (3) (2018) 207–223.

[29] K. Alemerien, K. Magel, GUIEvaluator: a metric-tool for evaluating the complexity of graphical user interfaces. SEKE, 2014, pp. 13–18.

[30] M. Soui, M. Chouchane, I. Gasmi, M.W. Mkaouer, PLAIN: PLugin for predicting the usability of mobile user interface. VISIGRAPP (1: GRAPP), 2017, pp. 127–136.
[31] M. Zen, Metric-based evaluation of graphical user interfaces: model, method, and software support. Proceedings of the 5th ACM SIGCHI Symposium on Engineering Interactive Computing Systems, ACM, 2013, pp. 183–186.
[32] M. Zen, J. Vanderdonckt, Towards an evaluation of graphical user interfaces aesthetics based on metrics. Research Challenges in Information Science (RCIS), 2014 IEEE Eighth International Conference on, IEEE, 2014, pp. 1–12.
[33] C.J. Boyatzis, R. Varghese, Children's emotional associations with colors, J. Genetic Psychol. 155 (1) (1994) 77–85.
[34] A.J. Elliot, M.A. Maier, A.C. Moller, R. Friedman, J. Meinhardt, Color and psychological functioning: the effect of red on performance attainment, J. Exp. Psychol. Gener. 136 (1) (2007) 154.
[35] M.M. Terwogt, J.B. Hoeksma, Colors and emotions: preferences and combinations, J. Gener. Psychol. 122 (1) (1995) 5–17.
[36] J. Noiwan, A.F. Norcio, Cultural differences on attention and perceived usability: investigating color combinations of animated graphics, Int. J. Hum.-Comput. Stud. 64 (2) (2006) 103–122.
[37] A. Miniukovich, A. De Angeli, Visual impressions of mobile app interfaces. Proceedings of the 8th Nordic Conference on Human-Computer Interaction: Fun, Fast, Foundational, ACM, 2014, pp. 31–40.
[38] P. Valdez, A. Mehrabian, Effects of color on emotions, J. Exp. Psychol. Gener. 123 (4) (1994) 394.
[39] K. Reinecke, T. Yeh, L. Miratrix, R. Mardiko, Y. Zhao, J. Liu, K.Z. Gajos, Predicting users' first impressions of website aesthetics with a quantification of perceived visual complexity and colorfulness. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, ACM, 2013, pp. 2049–2058.
[40] D. Cyr, M. Head, H. Larios, Colour appeal in website design within and across cultures: a multi-method evaluation, Int. J. Hum.-Comput. Stud. 68 (1–2) (2010) 1–21.
[41] A. Garrido, G. Rossi, N.M. Medina, J. Grigera, S. Firmenich, Improving accessibility of web interfaces: refactoring to the rescue, Univ. Access Inf. Soc. 13 (4) (2014) 387–399.
[42] A. Darejeh, D. Singh, An investigation on ribbon interface design guidelines for people with less computer literacy, Comput. Stand. Interfaces 36 (5) (2014) 808–820.
[43] B. Deka, Z. Huang, R. Kumar, Erica: interaction mining mobile apps. Proceedings of the 29th Annual Symposium on User Interface Software and Technology, ACM, 2016, pp. 767–776.
[44] A. Sahami Shirazi, N. Henze, A. Schmidt, R. Goldberg, B. Schmidt, H. Schmauder, Insights into layout patterns of mobile user interfaces by an automatic analysis of android apps. Proceedings of the 5th ACM SIGCHI symposium on Engineering interactive computing systems, ACM, 2013, pp. 275–284.
[45] K. Alharbi, T. Yeh, Collect, decompile, extract, stats, and diff: mining design pattern changes in android apps. Proceedings of the 17th International Conference on Human-Computer Interaction with Mobile Devices and Services, ACM, 2015, pp. 515–524.
[46] S. Ulusoy, A. Batıoğlu, T. Ovatman, Omni-script: device independent user interface development for omni-channel fintech applications, Comput. Stand. Interfaces 64 (2019) 106–116.
[47] D.C.L. Ngo, L.S. Teo, J.G. Byrne, Modelling interface aesthetics, Inf. Sci. 152 (2003) 25–46.
[48] Google, (accessed online 21, june 2021), 2019, (https://material.io/design).
[49] A. Seffah, M. Donyaee, R.B. Kline, H.K. Padda, Usability measurement and metrics: a consolidated model, Softw. Qual. J. 14 (2) (2006) 159–178.
[50] P.S. Gott, Cognitive abilities following right and left hemispherectomy, Cortex 9 (3) (1973) 266–274.
[51] UI-Restructuring, 2021, (https://github.com/NarjessBessghaier/UI-Restructuring).