METRICS AND TOOLS TO GUIDE DESIGN OF GRAPHICAL USER INTERFACES

A Dissertation
Submitted to the Graduate Faculty
of the
North Dakota State University
of Agriculture and Applied Science

By

Khalid Ali Alemerien

In Partial Fulfillment of the Requirements
for the Degree of
DOCTOR OF PHILOSOPHY

Major Department:
Computer Science

October 2014

Fargo, North Dakota

North Dakota State University
Graduate School

Title

METRICS AND TOOLS TO GUIDE DESIGN OF GRAPHICAL USER INTERFACES

By

KHALID ALI ALEMERIEN

The Supervisory Committee certifies that this ***disquisition*** complies with North Dakota State University's regulations and meets the accepted standards for the degree of

DOCTOR OF PHILOSOPHY

SUPERVISORY COMMITTEE:

KENNETH MAGEL
Chair

BRIAN SLATOR

WILLIAM PERRIZO

CAROL PEARSON

Approved:

10/17/2014
Date

BRIAN SLATOR
Department Chair

ABSTRACT

User interface design metrics assist developers evaluate interface designs in early phase before delivering the software to end users. This dissertation presents a metric-based tool called GUIEvaluator for evaluating the complexity of the user interface based on its structure. The metrics-model consists of five modified structural measures of interface complexity: Alignment, grouping, size, density, and balance. The results of GUIEvaluator are discussed in comparison with the subjective evaluations of interface layouts and the existing complexity metrics-models.

To extend this metrics-model, the Screen-Layout Cohesion (SLC) metric has been proposed. This metric is used to evaluate the usability of user interfaces. The SLC metric has been developed based on Aesthetic, structural, and semantic aspects of GUIs. To provide the SLC calculation, a complementary tool has been developed, which is called GUIExaminer.

This dissertation demonstrates the potential of incorporating automated complexity and cohesion metrics into the user interface design process. The findings show that a strong positive correlation between the subjective evaluation and both the GUIEvaluator and GUIExaminer, at a significance level 0.05. Moreover, the findings provide evidence of the effectiveness of the GUIEvaluator and GUIExaminer to predict the best user interface design among a set of alternative user interfaces. In addition, the findings show that the GUIEvaluator and GUIExaminer can measure some usability aspects of a given user interface. However, the metrics validation proves the usefulness of GUIEvaluator and GUIExaminer for evaluating user interface designs.

ACKNOWLEDGMENTS

I would like to thank all the people who contributed in some way to the work described in this dissertation. Foremost, I would like to express my sincere gratitude to my advisor Prof. Kenneth Magel for the continuous support of my Ph.D study and research, for his patience, motivation, enthusiasm, and immense knowledge. His guidance helped me in all the time of research and writing of this dissertation. I could not have imagined having a better advisor and mentor for my Ph.D study.

Besides my advisor, I would like to thank the rest of my dissertation committee: Prof. William Perrizo, Prof. Brian Slator, and Prof. Carole Pearon, for their encouragement, insightful comments, and hard questions.

I am deeply and forever indebted to my parents “Abu Khalid” and “Um Khalid” for their love, support and encouragement throughout my entire life. Also, I must express my father in law “Abu Malek” and my mother in law “Um Malek” for their continued support and encouragement. I would like to thank my brothers and sisters for supporting me spiritually throughout my life. In particular, I am also very grateful to my wife and best friend, Saba, without whose love and encouragement, I would not have finished this dissertation. I take this opportunity to express the profound gratitude and love to my beloved daughter “Lojain”.

An honorable mention goes to my friends for their understanding and supports on me in completing this research. Also, I would like to thank the NDSU community, not only for providing the funding, which allowed me to undertake this research, but also for giving me the opportunity to attend conferences and meet so many interesting people.

Last but not the least, the one above of us, “Allah”, for answering my prayers for giving me the strength to accomplish this research.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1. INTRODUCTION

The user interface is an important component of software applications. On the one hand, the user interface can facilitate the interaction between the user and the other software components. On the other hand, software applications have usability problems, which can result in confusing the users, and ultimately, loss of revenue [18]. Therefore, user interface design is becoming an increasingly important activity in software engineering [46]. Currently, if user expectations are not reasonably satisfied by the user interface, any software functionality may be unacceptable [13].

In Human Computer Interaction (HCI), a plethora of principles, standards, and guidelines are identified in research and industry communities that address the quality of user interface in general and its usability in particular [16] [17] [21]. But the designers still struggle to meet user expectations. Therefore, it is essential to have effective methods to evaluate user interfaces during the design-time [18] [19] [20]. One of the important methods is a metric-based evaluation, which provides quantitative details of user interface. Silva et al. [26] provided evidence about the capabilities of design metrics to evaluate the graphical user interface (GUI) during early stages of software development. As De Marco [54] stated that "*you cannot control what you cannot measure*". But measurement is meaningless if the quality model is not well-defined.

With the rapid advance of software, software metrics have become one of the foundations of software management in general and essential to the success of software development. Nowadays, software metrics are so important that recently they have been integrated into programming languages to provide developers statistical data about their code. One type of metrics suite is structural metrics. Of the structural metrics, particularly important are complexity metrics, which are used widely with code. Thus, complexity metrics have played a significant

role in software development. Complexity metrics, specifically, concentrate on the quality of software by examining its internal structure [45].

Recognizing the importance of software metrics, the software engineering community has put tremendous effort into developing software metrics in order to help developers understand software aspects, evaluate and improve software artifacts, and estimate and predict costs, effort, and time of different software development activities [48]. In addition, Ivory and Hearst [34] validated empirically web page design metrics. They found that metrics can predict the usability of web pages up to 80% that matches the results of expert evaluations of the same web pages. Moreover, there is a crucial demand to support the metric models by effective tools, which provide a precise metric calculation.

Therefore, the growth of interface complexity calls for the growth of the complexity metrics in the interface design process. In the HCI, user interface designers may use complexity metrics to understand the GUI usability, identify some interface issues, and quantify the interface aspects. Moreover, complexity metrics help to reduce the subjectivity of interface evaluation [47]. Although people realize the importance of interface metrics, the complexity metrics field still needs to grow quickly to meet the requirements of GUI design. This encourages us to investigate the importance of applying complexity metrics during the GUI design.

Another possible direction that can apply metrics to judge the quality of user interfaces is applying cohesion metrics. In the GUI design, cohesion metrics can be used to measure the relatedness among the widgets on a given user interface. These metrics measure the goodness of a given alternative design compared to the complexity metrics that measure the badness of a given alternative design. But, both of them can be used effectively to evaluate the user interface and predict the subjective rating of a given user interface.

A pervasive approach that supports user interface design is the model-based approach for designing the user interface [36]. But this approach has some limitations. First, effective automated tools are not available. Second, the evaluation of user interfaces is considered as a time-consuming task [37] and requires a high workload on interface designers [38] [40]. Third, fragment interface design is sometimes completed with precocious or suboptimal solutions [39]. Therefore, the metric-evaluation approach can be considered as an effective alternative to evaluate user interfaces. Furthermore, this approach considers the metrics as relevant estimators of interface aspects. Also, this approach may mitigate the potential model-based limitations. To conduct a metric-evaluation of user interfaces, we need a model that captures the interface aesthetics, a structured method to calculate the metrics that are defined in the conceptual model, and a tool that automatically provides designers some guidance as to which alternative user interface design best facilitates software use [17] [22]. Therefore, this methodology served as the basis to conduct this research.

To sum up, metrics provide quantitative and predictive views of potential problems of the user interface design. Thus, metrics are a powerful technique for evaluating interface design, regardless of which metrics suite has been used. In interface design time, designers need effective tools that quantify aesthetic factors and identify which factors can influence the user interface evaluation [15]. These tools are potentially beneficial in the user interface design process [17, 20].

1.1. Motivation

The primary motivation of this research is to investigate the role of complexity and cohesion metrics for evaluating user interfaces. First, a set of complexity metrics has been proposed based on the structure of interface layouts in order to help interface designers evaluate

interface designs. Second, Screen Layout Cohesion (SLC) metric, which is a hybrid metric. This metric measures aesthetic, structural, and semantic aspects of user interface. Therefore, there is a need to study and merge the complexity and cohesion metrics into the interface design environment for two reasons. First, there are several applications of the complexity measures in the software development process such as project size estimation, cost and time estimation, effort estimation, and software maintenance. In order to improve the user interface quality and the project controllability, it is necessary to control the interface complexity by measuring the related aspects in the user interface design. Second, the user interface is a key component of any software application. Moreover, good interfaces make the interaction between the human and the software seamless and as effortless as possible. Previous empirical studies have proven that the degree of layout complexity is important to aesthetic perception of user interfaces [11][12]. Therefore, if designers use the measures in GUI design to reduce the actual complexity and then mitigate perceived complexity, the user will be satisfied with the GUI design.

An additional motivating force is to address the lack of automatic tools to evaluate GUI designs. The calculation of structural measures has been performed manually or by using semi-automated tools. In this research, two quantitative tools were developed to, automatically, calculate the proposed complexity and cohesion metrics. However, quantitative tools can also help developers identify better solutions to user interface design problems and make better decisions when we have alternative designs [14].

The proposed metric-models and their complementary tools have been used to evaluate the interface layouts, even if their source code is not available. This may reduce the cost of software development.

1.2. Problem Statement and Research Questions

The findings of user interface evaluations vary when different end-users evaluate the same user interface, even if they use the same design criteria. Despite the abundance of usability and interface design principles and guidelines, user interface evaluation continues to be a pressing HCI issue [3][4]. The desired deliverable is that the GUI permits a user to perform tasks as effortless as possible. In contrast, the inconvenient user interface makes the interaction between the user and system functionality difficult. The existing GUI evaluation methods are ineffective in realizing this deliverable in efficient and inexpensive manner.

With the tremendous advances in technology, the user interfaces become complex. However, there is a need to balance between the interface complexity and usability [8]. Furthermore, the interface design is a difficult process [55][59] that needs effective tools and techniques to evaluate interface layouts in the early GUI development stages. The main challenge is the lack of effective numerical tools to measure the interface complexity and determine the best interface among alternative GUI designs [67].

Therefore, to handle this challenge, the complexity and cohesion metrics can be used not only help in reducing the cost of developing user interfaces but also in developing highly usable software applications. These metrics have been supported by complementary tools to automate the metrics calculations. In the perspective of problems regarding the design of high quality interfaces as well as less complex, this research addresses the following research questions:

RQ1: Is the GUIEvaluator effective to measure the complexity of a given user interface?

RQ2: How do the structural measures of user interface affect the interface complexity rating?

RQ3: To what extent can the metrics-model and its tool determine the best GUI design among a set of alternative GUI designs?

RQ4: For a given user interface, is there a significant correlation between the GUI usability factors given by users and the GUI rating given by the metrics-model and its tool?

RQ5: Can GUIEvaluator and its metrics measure, accurately, the complexity of a given user interface more than the existing metric-models?

RQ6: Can GUIEvaluator and its metrics predict, precisely, the usability of a given user interface more than the existing metric-models?

RQ7: Is the SLC metric effective to evaluate the usability of a given user interface?

RQ8: For a given pair of user interfaces for the same purpose, is the SLC metric effective to determine the better user interface design?

1.3. Proposed Solution

In order to enhance the user satisfaction through well-designed interfaces in early GUI development stages and support decision-making for user interface designs, the following solutions have been proposed. First, five design factors with their metrics based on the structural aspects of the interface layout have been proposed. Those design factors are: Alignment, grouping, size, density, and balance, which are considered as significant influences on interface usability [8]. This metrics suite is supported by a complementary tool called GUIEvaluator, which provides automatic metrics calculation. This tool is used to measure the complexity of an interface layout in order to judge the quality of the interface design.

Second, a Screen-Layout Cohesion (SLC) metric has been proposed, which measures the interface layout cohesion of widgets type on a screen, group and ungroup of widgets, and semantic relatedness. This metric is supported by a complementary tool called GUIExaminer,

which is used to determine whether an interface design is good or not. A good interface design means this interface design meets the user preferences. This tool uses 17 design principles to calculate the SLC metric for a given user interface. Given design alternatives, we can use the above solutions to determine which design is the optimal among the alternatives.

1.4. Objectives

One of the most significant issues in the user interface design is the need for an effective method for evaluating design in early development phase. In order to improve system efficiency and to diminish the user's mental load, the effective measures of the quality of user interface design should be defined [5]. Therefore, the main objective of this research is to develop evaluation-metric tools to automatically compute the complexity and cohesion metrics of user interfaces, which are based on the structural, aesthetic, and semantic aspects of user interface. These metrics allow designers to evaluate alternative interface designs of a given application. This objective raises a number of issues that need to be addressed. First, providing a metrics model, which extracts the structural, aesthetic, and semantic measures of a given user interface. Second, computing, automatically, the metrics using effective tools. Third, verifying the metrics in order to use them for evaluating user interfaces. Fourth, investigating the relations between the metrics and usability of user interfaces. Finally, investigating to what extent can the metrics meet the user expectations?

1.5. Methodology

In this section, I present the methodology, which has been used to answer the research questions. The answers can be obtained by conducting experiments. The methodology has the following steps:

1. Formulate hypotheses to be empirically tested. The hypotheses were stated to examine the effectiveness of the metric-models and their tools and validate them empirically.
2. Select a number of user interfaces, which are varying in terms of goals, number of controls on a user interface, and aesthetic orientation. Also, these interfaces are part of real applications or common for the research community. Forty user interfaces have been used to conduct the experiments in Chapters 4 and 6, and 48 user interfaces have been used to conduct the experiments in Chapter 7.
3. Determine the design factors that have been needed for investigation in this research. In the first metrics-model, the structural aspects of user interfaces have been investigated. In the SLC metric, the structural, aesthetic, and semantic aspects have been investigated.
4. Generate the interface evaluation-metric model. A mathematical formalization has been performed to transform the design factors into quantitative metrics. In addition, the values of these metrics were summed in one main metric that calculates the complexity of user interface. Each design factor was weighted based on the user rating. The same step was performed for SLC metric.
5. Calculate the values of each metric for each user interface.
6. Conduct a pilot experiment to prove the effectiveness of the metrics-model.
7. Develop GUIEvaluator and GUIExaminer tools that support the metrics computation.
8. Conduct actual experiments among end-users to collect reviews for each user interface.
9. Perform statistical tests to compare the subjective evaluations with results obtained by using the metric-models.
10. Analyze the results in discussion to show the relations between user reviews and metrics results.

CHAPTER 2. BACKGROUND

2.1. Graphical User Interface

A Graphical User Interface (GUI) is a software component that allows users to interact with an application by performing actions on it. From the user's perspective, the GUI can accept the user events as inputs and generate graphical output. Each GUI consists of graphical widgets. Each widget has a set of properties and their values which formalize the GUI layout. Users can perform actions using graphical widgets. These actions cause changes to the GUI state, which is shown to users immediately.

Recently, a variety of graphical widgets have been introduced by programming languages and GUI generators. In this dissertation, the tools have been developed using Visual Basic. Therefore, I have considered all widgets that are provided by Visual Basic. Also, I have considered the containers in Visual Basic such as GroupBox, Panel, and SplitContainer to represent groups of widgets in order to facilitate the metrics computation.

Currently, many types of user interfaces exist and three common types are: form-based, web-based, and virtual reality [57]. Form-based interfaces accept user's input using common devices, such as a mouse or keyboard, and provide graphical output on the screen. Web-based interfaces accept the user actions as inputs and generate web pages as output using the web browsers, these activities are performed via the internet or intranet. Lastly, virtual reality interfaces allow users to interact with three-dimension environment which does not exist in the real world. In this dissertation, I focus on form-based interfaces.

2.2. Graphical User Interface Layout

User interface layout shows the widgets structure and distribution on the screen. Users who are using a given application get their first impression of the user interface layout and visual

elements. Each user interface has a static structure that is shown to users when the interface opens and a dynamic structure that is generated during the program execution. There is a number of principles, guidelines, rules, and methods which help designers develop the interface layout. The widgets should be placed on the screen in certain proportions that may innately feel better for the end-user. Taking the aforementioned description into account, GUI design can be considered as service not self-expression [59].

Therefore, developing a sophisticated layout for GUIs is a critical activity in software development. In this dissertation, I concentrate on the structural aspects of GUIs. Especially, five common structural aspects that are involved in the metrics-model which is used to evaluate GUI layouts: Alignment, balance, size, grouping, and density.

2.3. Graphical User Interface Complexity

Software systems persistently are becoming more complex, especially, user interfaces. This claim always exists in software engineering, so there is nothing new. Nowadays, GUI designers confront the challenge of designing GUI that is simple and easy to use. In the GUI design, we can define that the complexity is a measurement of structural sophistication of GUI layout. Recently, user interfaces include more and more widgets to provide more features that are greater than the users' needs. So, this large number of features may lead to making GUIs complex. Regardless of the designer expertise, if GUI is poorly developed, the user will be frustrated with the software features [58]. Therefore, designers have distributed the widgets on the screen in an effective manner that facilitates performing the tasks easily and conveniently. Despite this, there is no optimal user interface look and feel, so some research work have attempted to bridge the gap between software engineers and HCI techniques by proposing

models that may help [35]. One of these techniques is the GUI complexity metrics which provide a powerful tool for evaluating user interface designs.

2.4. Software Measurement Theory

Software Measurement is important in software engineering, especially in empirical studies. Measurement theory validates the measures with respect to the attributes that they are assumed to measure. In addition, Measurement theory provides conditions, foundations of estimation and prediction models, clear definitions of attributes and their measures, measurement scales, and empirical properties of measures [63] [64]. To prove the practical utility of measures of intended attribute, these measures should be empirically validated [60] [65]. So, lack of empirical and theoretical validation of measures can lead to lack of confidence in those measures.

Measurement is defined as a set of operations to collect numbers or symbols which are assigned to attributes of entities in the real world somehow to characterize them according to clearly defined rules using a measurement approach [60]. Both entities and attributes are the key concepts of this definition. The entity is an object such as a widget on a given screen, process such as a GUI design, and an event such as usability testing milestone. An attribute is a defined characteristic of an entity such as the time to finish the GUI testing process, the size of source code of GUI, and the number of widgets on a given screen. Attributes are categorized into internal and external. Measuring the internal attributes depends on the entity itself, while measuring the external attributes depends on the entity and the environment in which the entity exists. Therefore, complexity, cohesion, and coupling are considered as internal attributes of the software product. In a user interface domain, complexity is considered as an internal quality attribute of user interfaces.

Finally, these attributes should be quantitatively expressed using metrics. These metrics should provide a measured approach and measure scale. The measurement approaches are the ways, such as measurement methods, measurement functions, and analysis models, which can be used by different metrics to obtain their respective measures [68].

This dissertation assumes that the product is the GUI and the internal attribute is the GUI complexity. Therefore, I provide measures that help us quantify GUI complexity, such as grouping complexity, density complexity, size complexity, alignment complexity, and balance complexity. In addition, number experiments have been performed to validate the proposed metrics. To sum up, we can apply the measurements to quantify the software development processes, resources, and products in general [61] [62] and especially the GUI design.

2.5. Graphical User Interface Metrics

In this section, we can highlight two types of GUI metrics: the GUI usability metrics and the GUI complexity metrics. While the GUI usability metrics are widely used in GUI evaluation, they still depend on subjective preferences. The research and industry communities need to pay attention to support utilizing the complexity metrics in GUI evaluation.

2.5.1. GUI Usability Metrics

GUI usability metrics can measure to what extent that a given user interface can be useful, easy to use, attractive, and satisfactory for specified users to achieve effectively particular objectives [33]. These metrics can be classified into two main categories: Testing metrics and predictive metrics.

Testing metrics measure the actual use of the software to identify encountered problems. So, these metrics require a fully functional software. A bundle of GUI usability testing metrics has been introduced in the usability literature: Preference metrics which quantify subjective

evaluation and user's preferences and performance metrics which measure the performance of users when performing a task in a certain context such as error rate and task time.

Predictive metrics provide a value that depicts some parameters and aspects of software usability, learnability, and efficiency of alternative designs to estimate or predict overall software usability [66]. In addition, predictive metrics can be used to predict learning, execution time, usability problems, and error cost of a given user interface design [67].

2.5.2. GUI Complexity Metrics

GUI complexity is computed based on the measures of user interface structure and the measures of human factors, which relate to the users such as: response time, search time, speed, and their preferences. Some software complexity metrics are incorporated into programming languages and development tools. These metrics exist as complementary functions in the software development, so developers can eliminate problems easily when the software development is done [23][24]. But there is still a need to propose, validate, and integrate complexity metrics to the GUIs development.

Software metrics, especially complexity metrics, can be utilized in a variety of ways with respect to GUIs [31]. First, complexity metrics can monitor some aspects of the GUI development, especially those that are easy to overlook. As a result, metrics can reduce the number of tasks that are assigned to the designer. Thus, the designer can focus on other critical GUI design aspects. Moreover, these metrics can be applied early without requiring user interface experts. Second, complexity metrics can play a significant role in GUI evaluation and simplify the systematic generation of GUI layouts. Consequently, these layouts can be optimized to correspond to the weighted average of selected metrics. Third, complexity metrics can be used to estimate the time, effort, and cost of implementing and testing GUIs. Finally, complexity

metrics can define which portion of a GUI has been tested by a given test suite. This is not an exhaustive list of ways to use complexity metrics. To summarize the aforementioned points, complexity metrics are a powerful tool for assisting GUI designers in generating high quality GUIs that meet user expectations.

Furthermore, combining the generation and evaluation tools in GUI development environment can help designers generate initial GUI layouts, modify these layouts, and evaluate the alternative designs for a given user interface using evaluation metrics within a coordinated development environment [31]. Therefore, this dissertation presents a metric-based tool that can facilitate the evaluation of GUI layouts in a software development environment without the need for fully functional software. Furthermore, adding the automation to the GUI evaluation can lead to potential benefits such as reducing the cost of GUI design and improving the consistency in the evaluation of alternative GUI designs [67].

2.5.3. Visual Cohesion Metrics

Cohesion is a measure of the degree of interrelatedness of software component parts [77]. The concept of cohesion is widely used in software engineering in a variety of ways. Cohesion can simplify the software structure and reduce dependencies between component parts. In the user interface design, cohesion can be applied to show how the widgets are related to each other for a given user interface. Therefore, visual cohesion is introduced based on the established cohesion concept in software engineering field. In GUI design field, cohesion metrics can be considered as hybrid metrics if they combine different design aspects such as semantic, structural, aesthetic aspects of user interfaces. Therefore, this dissertation presents GUIExaminer, a metric-based tool that can facilitate the evaluation of GUI layouts in a software development environment without the need for fully functional software.

2.6. Semantic Relatedness

Is a metric that measures the distance between the terms based on the semantics or the likeness of their meaning. Natural Language ToolKit (NLTK) [90] is a free suite of libraries, software, and corpora for Python language to support the research in natural processing language. One of the important corpuses that used in this toolkit is WordNet [94], which is a collection of English words, examples, and relation among the terms. It provides sets of synonyms of terms. Moreover, one of measures that use to compute the semantic relatedness is the Wup, which is a metric that measures the path length to the root node from the least common subsumer (LCS) of the pair of nouns or verbs.

In this research, the NLTK has been used to provide values of semantic relatedness among a set of widget names. These values have been stored in the database that is used for the GUIExaminer to calculate SLC metric. To verify the values provided by NLTK, the WordNet::Similarity [93] has been used, which is free software that measures the semantic relatedness between a pair of terms. It provides nine measures of Semantic similarity and relatedness, which are based on the lexical database WordNet [91]. These measures return a numeric value that represents the degree to which they are similar or related. The returned value is in the range 0 to 1. The existing studies have focused on the effects of semantic relatedness or cohesion of layout widgets such as Niemelä and Pertti [95] investigated the influences of semantic cohesion and spatial grouping of widgets on learning the interface and Tim and Anthony [96] [97] investigated the effects of semantic grouping on visual search. In this research, the semantic relatedness or cohesion can be used as a main parameter of the SLC metric. The semantic relatedness metric provides a numeric value that describes, semantically, the relatedness among the widgets in one group.

CHAPTER 3. RELATED WORK

In this chapter, I will review some of the existing work in the area of my research. I focus on the conception and use of metrics in the GUI design process, the metric-evaluation tools that provide metrics calculations, the empirical studies that focused on user interface complexity, and cohesion metrics in graphical user interface design.

3.1. Complexity Metrics for Graphical User Interface Design

Stickel et al. [2] introduced a method to calculate the visual complexity using the number of possible interactions, functional elements, higher level of structures, organizational elements, and summed entropy of RGB values. The entropy of screens was calculated by using a MatLab script. Sears [3] developed a layout metric called Layout Appropriateness (LA). Each task needs a sequence of actions to be performed and for each task, LA metric attempts to calculate the costs of each sequence of actions. Therefore, these costs are based on distance of mouse moving, number of eye fixation to extract relevant information, or number of changes in direction. These metrics need a detailed description of the sequence of user tasks and functional elements. In addition, collecting the task description is a time consuming mission.

Kang and Seong [5] introduced Task-To-Action Model (TTA) which describes the procedure to perform a task by operators. They suggested three new measures to determine the complexity of user interface design: Operation, transition, and screen complexity. This model has three shortcomings: First, the measures cannot consider the difference between the states. Second, the measures may be partially dependent. Third, the evaluation is incomplete to provide absolute evaluation results.

Ngo et al. [11, 12] introduced a new aesthetic model which consists of 14 aesthetic measures for screen layouts. Each measure is supported by an example. Fongling et al. [9]

developed a metrics model to evaluate the complexity of web pages consisting of four design factors: Density, alignment, grouping, and size. In addition, Parush et al. [10] developed a numerical model to evaluate the GUI. This model consists of four screen factors: element size, grouping, alignment, and local density. The above metrics were calculated manually. In addition, Silva et al. [26] proposed UMLi to help GUI designers elicit user interface elements in use cases and their scenarios during the software design. In addition, they used design metrics for evaluating UMLi compared with UML. It was the first study to investigate the benefits of extensions of UML. They used the Library system as a case study to illustrate UMLi. Then they selected design metrics to assess the structural complexity, behavioral complexity, and visual complexity.

Kokol et al. [29] introduced a set of metrics in order to assess some characteristics of user interface. They focused on efficiency and effectiveness of user interfaces through the following measures: Amount of information per screen, quantity of input information between two enters and returns, and difference between screens similar to the metrics that measure coupling and cohesion of code. Furthermore, they measured the screen structure based on graph theory. The shortcoming of this research is the weights that were used in equations were set by the researchers without explanation. In addition, they focused on the data belonging to the entities on one or two different user interfaces.

Tullis [30] performed experiments to evaluate the effectiveness of a set of task-independent metrics for evaluating user interfaces. These metrics measure the following: The percentage of space on the user interface to present information and group of items, the number of items and groups on the user interface, average size of groups, and alignment. This metrics model was supported by a tool. Designers have to input the following information: ASCII file of

the interface, physical characteristics, and distance between the end-user and screen. This information is required for the interface analysis process. Kim and Foley [31] proposed a set of metrics to help designers generate interfaces iteratively. The metrics measure: Ratio of used space on the left and right sides and the top and bottom sides of the interface, symmetry, and unused space, the ratio of height and width of user interface compared to the "golden ratio", and size of the user interface compared to a desired range. But the researchers did not weight the metrics. In addition, they focused on generating user interfaces rather than evaluating the effectiveness of their metrics.

Shazia et al. [32] introduced a set of metrics to estimate the design quality of web applications. These metrics were used for Object Oriented Design Model (OODM). OODM includes four design models. Each model was supported by a metric as the following: Component model that uses reusability metric, navigational model that uses navigational accessing time metric, an operational partitioning model that uses operation performance metric, and user interface model that uses interface coherence metric. In addition, these metrics were evaluated through detailed example that illustrates how to calculate these measures. But the researchers do not provide evidence of the effectiveness of these metrics to evaluate webpage layouts.

Rauterberg [41] proposed metrics to evaluate user interfaces through the dimensions: Interactive directness, functional feedback, and flexibility. These metrics can be used to classify the user interfaces into command, desktop, and menu. But the researcher did not provide any details about metrics evaluation and computation. In addition, Xing [42] developed metrics of information complexity for automation display. The author used three factors associated with complexity: numeric size, variety of elements, and relation between elements. These factors are

associated with the stages of information processing in the human brain: perception, cognition, and action in order to measure the information complexity and its implications. Each combination between the factors and stages of human brain is supported by a metric. These metrics are based on theoretical studies. So, there is a need for evidence to validate these metrics before using them in the real world. The proposed metrics-model consists of five complexity metrics that measure the structural aspects of user interfaces. These metrics were validated through experiments that proved the effectiveness of these metrics to evaluate user interfaces. The findings show a high correlation between the user preferences and the values that are produced by the metrics-model.

3.2. Graphical User Interface Evaluation Tools

Several research studies were run in the interface design field that focused on developing and calculating metrics models manually or semi-automatically, which help them calculate some aspects of user interface designs. Comber and Maltby [1][25] developed an application called Launcher, which is used to calculate the layout complexity based on dimension and position of objects. In addition, they defined the layout complexity as a quantitative method for determining the relative order and disorder of the objects. Zheng et al. [4] developed a MatLab tool to calculate image vectors which are the values of five image features: color saliency, intensity saliency, spatial frequency, color entropy, and texton entropy.

Sears [6] introduced a metric-based tool called AIDE, which partially automates the designing and evaluating user interface layouts based on various metrics for a given set of interface controls. AIDE includes the following metrics: Efficiency, balance (both horizontal and vertical), alignment, and constraints. Miyoshi and Murata [8] developed a numerical tool for evaluating the usability of a screen design based on its complexity measures. This tool was

developed based on four factors (size, density, grouping, and alignment) to investigate the relationship between those factors and search time. González et al. [13] presented BaLOReS, which is a suite of aesthetic principles and their metrics together. Those principles are: Balance, linearity, orthogonality, regularity, and sequentiality. In addition, they developed a tool called BGLayout which helps them automate the metrics calculation using image processing techniques.

Seffah et al. [33] developed a tool called Quality in Use Integrated Management (QUIM), which consists of a repository of 127 usability metrics. These metrics were collected from existing models and standards. The tool was developed based on a unified hierarchal model that defines 10 factors, each of which represents a specific aspect of usability. Also, these factors were decomposed into 26 criteria which define the 127 usability metrics.

Yoon et al. [35] proposed three metrics to assist designer's decision making in the design process, including inefficiency, complexity, and incongruity. A case study was used in this research work, 3 solutions for an MP3 player. They used OCD (Operation-Control Diagram) and S-O matrix (State-Operation matrix) for modeling system aspects. The metric values can reflect the status of interface design.

Silva et al. [52][53] developed a tool called GUISurfer, a source code analysis tool which is used to minimize the time to understand and evaluate software systems. Moreover, the tool is composed of three components: File Parser, AstAnalyser, and Graph. This tool reverse engineers the GUI layer of software systems through generating state machine model and showing how graph theory can be useful in this model. They used three metrics in the analysis of the user interface's quality: shortest distance between vertices, pagerank, and betweeness.

GUIEvaluator extends the ideas presented in previous work in two directions. First,

GUIEvaluator is a fully automated tool that calculates and analyzes the five structural interface measures. This helps designers determine which design factor has a high complexity rating and allows designers to evaluate, redesign, and reevaluate their interface designs from within a single interface design environment. The previous tools are semi-automated tools. Furthermore, these tools calculated other design metrics. Second, GUIEvaluator incorporates the weight for each design factor based on user preferences. This leads to consider the human factors as a part of the proposed metrics while the previous studies did not take into account the importance of design factors.

3.3. Empirical Studies on User Interface Complexity and Usability Metrics

Coskun and Grabowski [27] studied the complexity of GUI and its impacts on the user's acceptance for Safety-Critical Systems such as the effects of GUI complexity of user's behaviors and decisions. However, they used qualitative metrics to measure the complexity of GUI with Navigation and Piloting Expert System (NPES). So, they introduced a set of metrics that include general metrics and NPES-specific metrics. The shortcoming of this study is that it is based on subjective rating without quantitative methods and limited number of participants (three participants). In addition, the study time was very long, it took more than 90 minutes, which may have an affect the participant evaluation.

Sobiesiak and Diao [28] used complexity metrics to understand and quantify the software usability to identify some software usability issues. Complexity analysis is a technique that can be used to evaluate software usability and help to reduce the subjectivity of usability evaluation. Furthermore, they proved that higher complexity measure correlates lower usability for the same task. Therefore, there is a trade-off between usability and complexity. Their study was conducted at task, release, and product levels. However, two case studies are used in complexity analysis:

First, installation of instant messaging software, and second, development and design of DB2 for Linux, UNIX, and WINDOWS. Third, they compared complexity analysis with other usability evaluation techniques. In this research, they did not use the structural measures to evaluate the complexity of user interfaces.

Victor et al. [42] used Web Quality Model (WQM) to classify GUI usability metrics, which are measured in three dimensions: First, quality includes content, presentation, and navigation. Second, life cycle processes include processes with standard ISO12207. Third, web features based on Quint2 model and ISO 9126 standard (functionality, usability, efficiency, portability, maintainability, and reliability).

Lo et al. [44] developed a GUI estimation effort model to investigate the relationship between the effort to code and unit test of GUI systems and number types of widgets (e.g., text boxes, check boxes, list boxes). This model assists developers predict the GUI development effort. They classified widgets into three categories based on development effort: action widgets (involving database, not involving database), data widgets (involving list, not involving list), and static widgets. Moreover, this model was built based on the number of widgets that appear in each category and the effort in terms of hours. This study consists of 8 hypotheses that were tested and these hypotheses represent the relationship between development effort and widgets.

Three empirical studies [49][50][51] were conducted to investigate the effectiveness of the metrics that were proposed in [41]. The author performed three comparative usability studies. In the first study, the author compared menu-driven interface (CUI) with graphical user interface. The author found that the mean task completion time with GUI is significantly shorter than with CUI. In the second study, the author compared two different versions of the same application (one with a GUI based on net-shaped dialog and the other one with a GUI based on the

hierarchical dialog structure) and the author found that there is no significant difference between them in task completion time and number of mask successions. In the last study, the author compared two different versions of the same application (one with CUI based on net-shaped dialog and the other one with CUI based on the hierarchical dialog structure) and the author found that there is no significant difference between them in target discrepancy.

Mamillapally et al. [56] performed an experiment to compare between web site design versions based on usability and complexity metrics. So, this study illustrates how interface complexity metrics can be used early to engineer a more usable website.

Since the aforementioned empirical studies provide an evidence to designers about the importance of employing the design metrics in user interface evaluation, the design metrics in general and the complexity metrics specifically are effective tools in the user interface design process.

3.4. Cohesion Metrics for Graphical User Interface Design

Several studies have proposed metrics focused on cohesion in the user interface design. Constantine [77] proposed a visual coherence metric to evaluate the user interface quality. The results show that users prefer the more visual designs. This metric computes the visual cohesion based on the semantic aspects of widgets that exist in any visual group. The shortcoming of this metric is only focused on the visual group. So, this metric is insufficient if there is no visual group on a given user interface. Moreover, three non-functional user interface designs were utilized in this study to validate the visual coherence metric. So, the number of interfaces is limited to generalize their findings.

Kokol et al. [29] defined a set of metrics to evaluate some aspects of user interface. One part of the hybrid metric is the differentiation metric, which is mainly developed based on the

screen coherence. They measured the cohesion in terms of relationships between data of entities on a given user interface. Cohesion can be low, medium, or high. The cohesion metric of the study has the following weaknesses: It consists of weights assigned to data of entities; these weights were set arbitrarily by the authors. In addition, there is no way to determine the influence of each type of data that was used in this metric on the overall cohesion value for a given user interface. Finally, the metrics suite has not been validated theoretically or empirically. So, there is no evidence about its validity to use in the GUI design.

Shazia et al. [32] proposed a user interface model which is supported by interface coherence metric. This metric is used to estimate the design quality of web applications. This metric calculates three cohesion modes: Low, medium, and high. The overall cohesion is determined by summing the values of cohesion modes. The researchers have not taken into account the weight of these cohesion modes because they assumed that they have the same influence on overall cohesion. This is considered as a weakness of this metric. Moreover, this metric computes the cohesion based on the relationships between the widgets on the same screen or other screens, but they did not specify which relationships were considered in this metric.

Ngo et al. [11] [12] introduced a cohesion metric as part of an aesthetic model, which consists of 14 aesthetic measures, to evaluate user interface layouts. They provided formulas to compute these measures. They provided an example to show how to calculate manually these measures. The researchers used only height and width of widgets to calculate the cohesion measure. So, they focused on one aspect to judge the GUI coherence.

To overcome the shortcomings of the above cohesion metrics in the user interface design, SLC metric, which is a cohesion metric that can be used to evaluate the quality of user interfaces taking into account the weighted aesthetic, structural, and semantic measures of user interfaces.

CHAPTER 4. THE GUIEVALUATOR TOOL AND THE METRICS-MODEL

4.1. The GUIEvaluator Tool

4.1.1. The Architecture

The architecture of GUIEvaluator is data-centric. Figure 4.1 shows GUIEvaluator's architecture which consists of five components: Data Extractor, MetricsCalculator, Complexity Analyzer, GUI, and Database. The Data Extractor plays an intrinsic role in GUIEvaluator's functionality. It is utilized to read a user interface (.vb) developed in Visual Basic and extract the structural properties of each widget on the user interface using reflection techniques which are provided by Visual Basic. The MetricCalculator is used to provide metrics calculation, which makes the metrics' values available for Complexity Analyzer. The Complexity Analyzer is implemented to analyze the results of metrics calculation in order to provide a final decision about a given user interface design. All the GUIEvaluator's functionalities are presented through a simple attractive GUI. Finally, the Database is utilized to store the collected data, factors' weights, and calculated values of metrics. With these components, GUIEvaluator can be utilized to evaluate user interface layouts whether the user interface is under development or a part of running applications.

4.1.2. Graphical User Interface of GUIEvaluator

The user interface of GUIEvaluator was developed to be as simple as possible to facilitate the data extraction and metrics calculation for a given user interface. Figure 4.2 shows the GUI of the data extraction process. The user can utilize the GUI to perform the following: Upload new project and new user interface, extract data about user interface layout, update the weight of

design factors, show the values of structural metrics and the overall user interface complexity, and save the results.

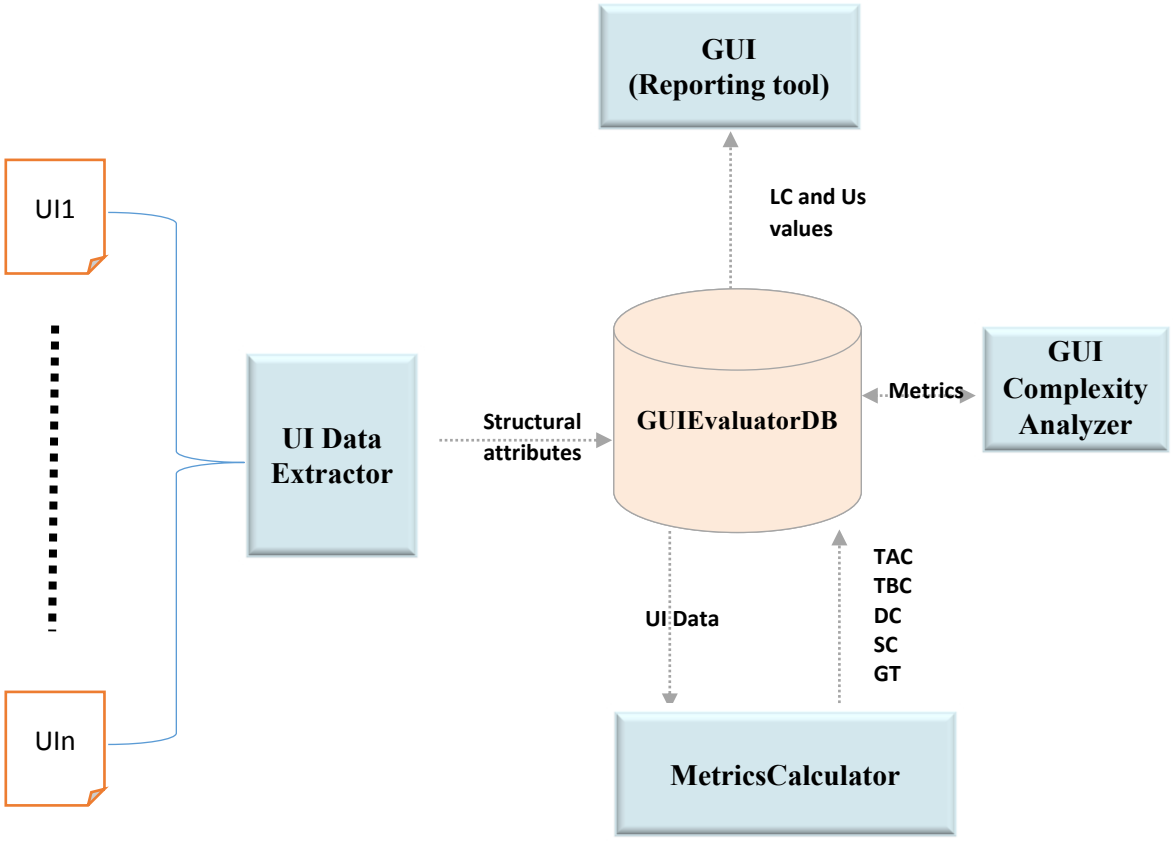

Figure 4.1. The architecture of GUIEvaluator

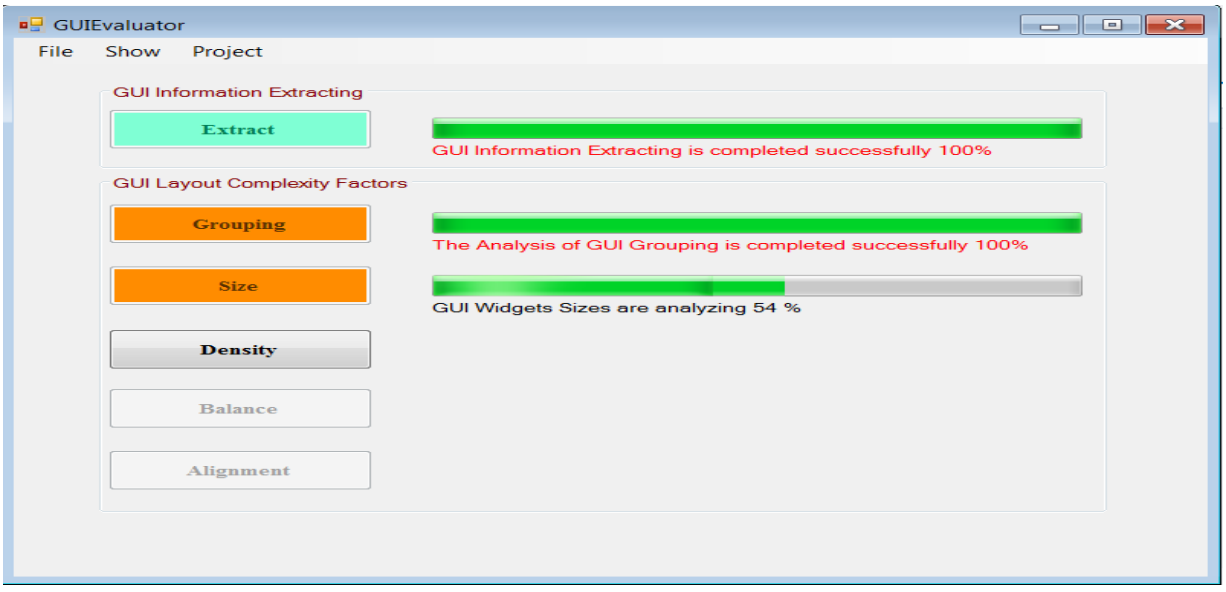

Figure 4.2. The extraction and analysis window of GUIEvaluator

4.2. The Metrics-Model

This model consists of five metrics for measuring five design factors: Alignment, grouping, density, balance, and size. The values of these metrics will be used to calculate the interface design complexity. These metrics are the following:

1. *Alignment*

Alignment metric measures the vertical and horizontal alignment of objects in two levels: A group level (Local Alignment) and screen level (Global Alignment). These two alignment levels are combined to calculate the Total Screen Alignment Complexity.

A. Local Alignment

Equation 4.1 calculates the alignment for each group. Where Vp is the number of vertical alignment points and Hp is the number of horizontal alignment points. K is the number of grouped objects on the screen. The range of values of GA is [0, 1]. Equation 4.2 calculates the Alignment Complexity (AC) for all groups on the screen. Where the Weight is the number of objects in a group (i) divided by the total number of grouped objects, and m is the number of groups on the screen.

$$Group\ Alignment\ (GAi) = \frac{\sum_{i=1}^{K}(Vp+Hp)}{2K} \quad \text{(Eq. 4.1)}$$

$$AC = \sum_{i=1}^{m} GAi * Weight(i) \quad \text{(Eq. 4.2)}$$

B. Global Alignment

The global alignment is calculated as shown in Eq. 4.3, where Vp is the number of the vertical alignment points and Hp is the number of the horizontal alignment points**.** N is the number of ungrouped objects on the screen. The range of values of the SA is [0, 1]**.**

$$Screen\ Alignment\ (SA) = \frac{\sum_{i=1}^{N}(Vp+Hp)}{2N} \quad \text{(Eq. 4.3)}$$

C. Total Screen Alignment Complexity

Equation 4.4 shows the Total Alignment Complexity (TAC).Where weight1 is the ratio of the number of grouped objects to the total number of objects on the screen, while the weight2 is the ratio of the number of ungrouped objects to the total number of objects on the screen.

$$TAC = AC * weight1 + SA * weight2 \quad \text{(Eq. 4.4)}$$

2. *Balance*

Balance metric uses the number and size of objects from Eq. 4.5 and Eq. 4.6, respectively, for each quarter of the screen to calculate the balance complexity. The BQni and BQnj represent the number of objects in *ith* and *jth* quarters. The range of ratio of BQni and BQnj is [0, 1] and the range of values of BQn Overall is [0, 1], where 0 means unbalanced and 1 means fully balanced in terms of number of objects. The BQsi and BQsj represent the sum of sizes of objects in ith and jth quarters. The range of ratio of BQsi and BQsj is [0, 1] and the range of values of BQs Overall is [0, 1], where 0 means unbalanced, 1 means fully balanced in terms of object size. Equation 4.7 is the Total Balance Complexity (TBC).

$$BQn = \frac{\sum_{k=1}^{6} \frac{BQni}{BQnj}}{6} \quad \text{(Eq. 4.5)}$$

$$BQs = \frac{\sum_{k=1}^{6} \frac{BQsi}{BQsj}}{6} \quad \text{(Eq. 4.6)}$$

$$TBC = 1 - (0.5 * BQn + 0.5 * BQs) \quad \text{(Eq. 4.7)}$$

3. *Density*

Density metric measures the screen occupied by objects, where Eq. 4.8 calculates the Local Density (LD) for the group *jth* and Eq. 4.9 calculates the Global Density (GD). In Eq. 4.10, the Density-Complexity (DC) is calculated taking into account the W1, which is the ratio of

the area of groups to the screen area, and W2, which is the ratio of ungrouped area to the screen area.

$$LDj = \frac{\sum_{i=1}^{groupedobjects} Size\ of\ object\ i\ in\ a\ group\ J}{Area\ of\ group\ J} \quad \text{(Eq. 4.8)}$$

$$GD = \frac{\sum_{k=1}^{ungroupedobjects} Size\ of\ object\ k}{Area\ of\ ungrouped} \quad \text{(Eq. 4.9)}$$

$$DC = (\frac{\sum_{j=1}^{n} LDj}{n}) * W1 + GD * W2 \quad \text{(Eq. 4.10)}$$

4. Size

Size metric measures the object size complexity of two levels: The object size complexity (SC_k) as in Eq. 4.11 and the overall size complexity (SC) as in Eq. 4.12, where N is the number of objects in type *kth* and Sj is the number of different sizes, where Sj is 1 if the object size is not counted before and 0 if the object size is counted. Wi is the number of object types and Weight (k) is the number of objects in type *kth* divided by the total number of objects on the screen.

$$Object\ Size\ Complexity\ (SC_k) = \frac{\sum_{j=1}^{N} Sj}{N} \quad \text{(Eq. 4.11)}$$

$$SC = \frac{\sum_{k=1}^{Wi} SCk * Weight(k)}{Wi} \quad \text{(Eq. 4.12)}$$

5. Grouping

Grouping metric measures the number of objects that have a clear boundary by line, background, color, space, or size. Equation 4.13 calculates the percentage of ungrouped objects (UG), where the GW is the objects that are grouped together. The value of GW equals 1, if the object exists in a group, otherwise GW equals 0. N is the total number of objects on the screen. Equation 4.14 calculates the ratio of the number of different object types (G) to the total number of objects (M) in all groups, where the Weight is the ratio of total number of grouped objects to

total number of objects on the screen. To calculate the grouping complexity (GT), we can use Eq. 4.15.

$$Ungrouped\ Objects\ Complexity\ (UG) = 1 - \frac{\sum_{i=1}^{N} GW}{N} \quad \text{(Eq. 4.13)}$$

$$Grouped\ Objects\ Complexity\ (GC) = \frac{G}{M} * Weight \quad \text{(Eq. 4.14)}$$

$$\textit{The Grouping Complexity (GT)} = UG + GC \quad \text{(Eq. 4.15)}$$

6. Overall Screen Layout_Complexity (LC)

Equation 4.16 calculates the Overall Screen Layout_Complexity (LC), which includes the values of TAC, TBC, DC, SC, and GT metrics. Each metric has a weight (w1, w2, w3, w4, and w5, respectively), which are calculated based on the participant rating. The values of those weights are 0.84, 0.76, 0.80, 0.72, and 0.88, respectively. And also, the sum of the weights equals 1.

$$LC = (TAC * w1 + TBC * w2 + DC * w3 + SC * w4 + GT * w5) * 100\% \quad \text{(Eq. 4.16)}$$

We can measure the usability of user interfaces using the complexity metrics that have been defined in the metrics-model. Let usability is denoted by U_s then we can write

$$U_s = 1 - LC \quad \text{(Eq. 4.17)}$$

4.3. An Illustrated Example of Layout Complexity Calculation by GUIEvaluator

In this example, given two real user interface layouts for the same purpose are used: The interface of remote desktop connection provided by Microsoft as shown in Fig 4.3 and the interface of remote desktop enabler provided by Intelliadmin as shown in Fig 4.5. The proposal complexity metrics have been calculated for those two user interfaces using the GUIEvaluator. The main goals of this example are to illustrate how to calculate the proposal complexity metrics, and compare the values of layout complexity for real user interfaces in order to confirm the

practicality of proposal metrics. Fig 4.4 and Fig 4.6 show the corresponding models of the remote desktop connection interfaces by Microsoft and Intelliadmin, respectively. These models show the layout structure without content to facilitate extracting the structural measures of a given user interface.

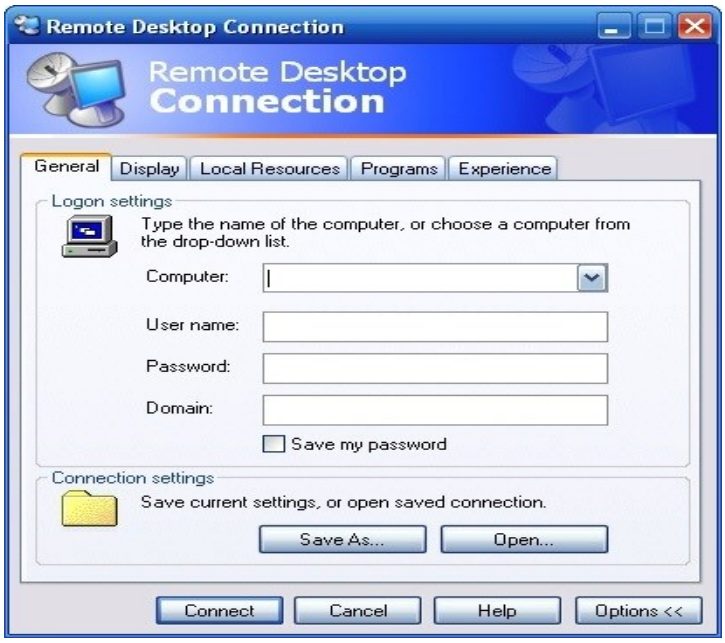

Figure 4.3. The user interface of remote desktop connection of Microsoft

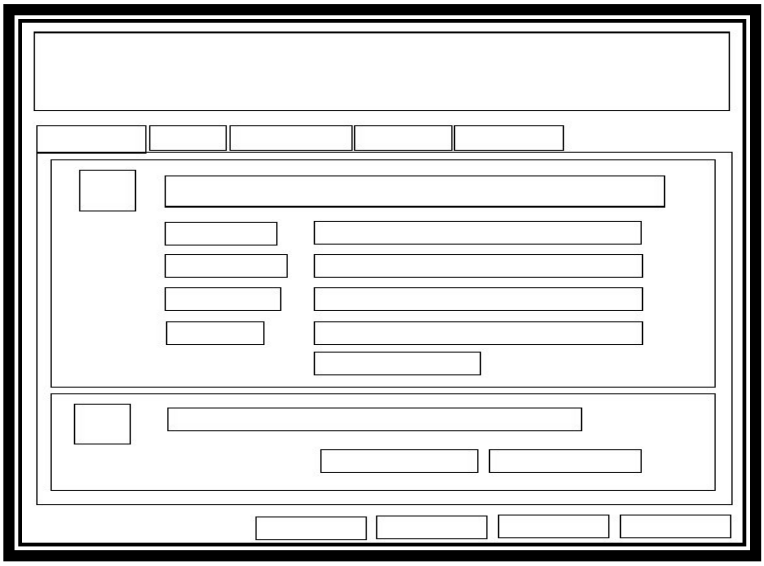

Figure 4.4. The model of the user interface for remote desktop connection of Microsoft

Figure 4.5. The user interface of remote desktop connection enabler of intelliadmin

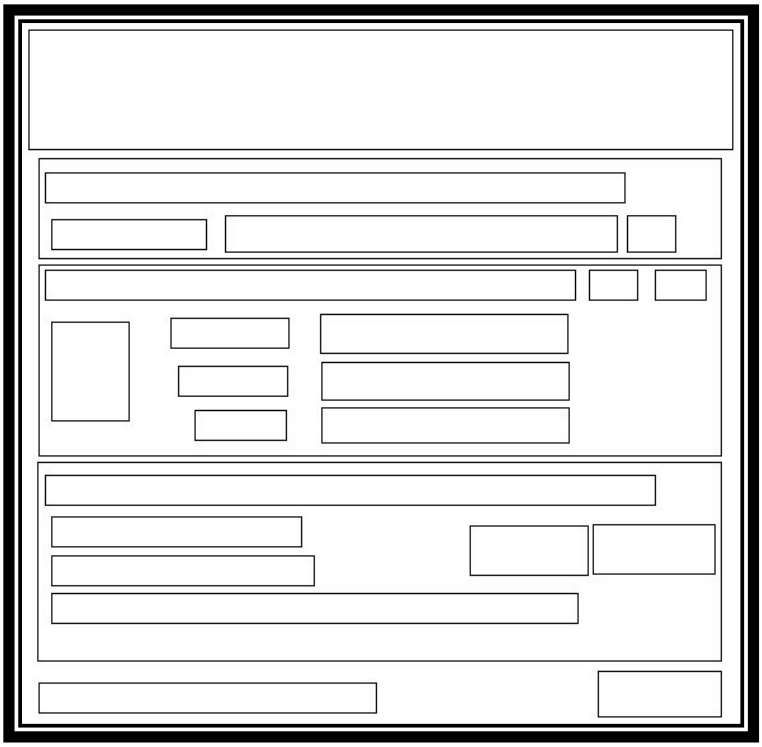

Figure 4.6. The model of the user interface of remote desktop connection of intelliadmin

Table 4.1 presents the values of properties of the two user interfaces of remote desktop connection. These values are in pixels. These properties are the number and percentage of grouped and ungrouped widgets, total number of widgets, width and height to calculate the area of interface, center in both dimensions X and Y, and number of groups. The values of these properties are used as a part of the complexity metrics calculations. For example, the number and percentage of grouped widgets are used in the grouping complexity metric calculation and Center_X and Center_Y are used in balance complexity metric calculation.

To calculate the complexity of local density and grouping, the widget groups and their properties of a given interface layout have to be determined. Table 4.2 shows three widget groups and their properties of the remote desktop connection interface by Intelliadmin and four widget groups and their properties of the remote desktop connection interface by Microsoft. Each group has the following properties: The location, which is identified by the values of X1, X2, Y1, and Y2, the number of widgets, the number and percentage of widget types, and the area, which is computed based on the width and height of interface layout.

Table 4.1. Two user interfaces of remote desktop connection and their properties

Property	**User Interface**	
	By intelliadmin	**By Microsoft**
Number of Widgets	23	25
Number of Grouped Widgets	20	24
Percentage of Grouped Widgets	0.870	0.960
Number of Ungrouped Widgets	3	1
Percentage of Ungrouped Widgets	0.130	0.04
Height	505	577
Width	451	504
Center_X	225.5	252
Center_Y	252.5	288.5
Area	214225	275688
Number of Groups	3	4

Table 4.2. The widget groups on the remote desktop connection interfaces and their properties

Property	Widget groups on the remote desktop connection interface						
	By intelliadmin			By Microsoft			
	Step1	Step2	Step3	Menu	Logon	Connection	Buttons
X1	13	13	13	11	22	22	100
X2	440	440	440	483	475	475	487
Y1	139	210	330	137	172	418	528
Y2	206	326	465	177	416	505	461
Number of Widgets	4	10	6	5	11	4	4
Width	427	427	427	452	452	452	387
Height	67	116	135	45	242	80	67
Area	28609	49532	57645	20340	109384	36160	25929
Number of Widget Types	3	4	4	1	5	3	1
Percentage of Widget Types	0.75	0.4	0.67	0.2	0.45	0.75	0.25

The values of properties of all widgets for both user interfaces by Intelliadmin and Microsoft are presented in Table 4.3 and Table 4.4, respectively. The two tables, for each widget, present the group id of widgets, the widget width and height to calculate its area, widget type, and the locations of widgets through the values of X1, X2, Y1, and Y2. The remote desktop connection interface by Intelliadmin includes 23 widgets, where 20 of these widgets are in groups and three widgets are not grouped. In contrast, the remote desktop connection interface by Microsoft comprises 25 widgets, where 24 of these widgets are exist in groups and one widget is not grouped.

Table 4.5 shows the values of five complexity measures and their subjective weights for both user interfaces of remote desktop connection by Intelliadmin and Microsoft. The values of the five complexity measures of the user interface by Intelliadmin are: Size complexity (0.870), alignment complexity (0.739), density complexity (0.519), grouping complexity (0.368), and balance complexity (0.371). In contrast, the values of the five complexity measures of the user interface by Microsoft are: Size complexity (0.720), alignment complexity (0.640), density complexity (0.423), grouping complexity (0.171), and balance complexity (0.236). The values of

Table 4.3. Widget properties of user interface of remote desktop connection of intelliadmin

Group ID	Widget Type	Width	Height	Size	X1	X2	Y1	Y2
1	Label	376	17	6392	18	394	148	165
1	Label	114	17	1938	18	132	176	193
1	Button	28	28	784	390	418	170	198
1	TextBox	67	28	1876	134	201	170	198
2	PictureBox	56	60	3360	23	79	246	306
2	Label	320	15	4800	17	337	214	229
2	Label	74	15	1110	86	160	240	255
2	Label	68	15	1020	88	269	262	277
2	Label	57	15	855	102	159	298	313
2	RadioButton	45	15	675	341	386	214	229
2	RadioButton	40	15	600	391	431	214	229
2	TextBox	164	32	5248	166	330	235	267
2	TextBox	164	32	5248	166	330	264	296
2	TextBox	164	32	5248	166	330	293	325
3	RadioButton	162	15	2430	23	185	353	368
3	RadioButton	162	15	2430	23	185	372	387
3	CheckBox	332	15	4980	23	355	392	407
3	Label	386	15	5790	18	404	333	348
3	Button	95	22	2090	267	362	353	375
3	Button	85	22	1870	353	438	353	375
N	PictureBox	452	105	47460	0	452	0	105
N	Button	90	22	1980	348	438	468	490
N	LinkLabel	193	15	2895	12	205	469	484

five complexity measures have been used in the computation of overall layout complexity for both interfaces. The overall layout complexity values of remote desktop connection interfaces by Intelliadmin and Microsoft are 0.567 and 0.431, respectively. According to these calculations, the interface of Microsoft is less complex than the interface of Intelliadmin with complexity scores 4 and 6, respectively.

Table 4.4. Widget properties of user interface of remote desktop connection of Microsoft

Group ID	Widget Type	Width	Height	Size	X1	X2	Y1	Y2
1	Button	63	23	1449	13	76	132	155
1	Button	57	23	1311	74	131	132	155
1	Button	114	23	2622	129	243	132	155
1	Button	67	23	1541	243	310	132	155
1	Button	76	23	1748	307	383	132	155
2	Label	230	30	6900	94	324	190	220
2	Label	62	15	930	94	156	234	249
2	Label	70	15	1050	94	164	262	277
2	Label	376	15	5640	94	470	278	293
2	Label	24	15	360	94	118	316	331
2	TextBox	239	25	5975	178	417	275	300
2	TextBox	239	25	5975	178	417	312	337
2	TextBox	239	25	5975	178	417	350	375
2	ComboBox	239	25	5975	178	417	231	256
2	CheckBox	132	20	2640	178	310	386	406
2	PictureBox	40	40	1600	41	81	187	227
3	Button	120	25	3000	301	421	465	490
3	Button	120	25	3000	175	295	465	490
3	Label	285	15	4275	94	379	438	453
3	PictureBox	40	40	1600	41	81	436	476
4	Button	87	25	2175	104	191	530	555
4	Button	87	25	2175	297	384	530	555
4	Button	87	25	2175	394	481	530	555
4	Button	87	25	2175	200	287	530	555
N	PictureBox	504	88	44352	0	504	0	88

Table 4.5. The values of overall layout complexity and its measures

Complexity Measure	Weight	User Interface of remote desktop connection	
		By Intelliadmin	By Microsoft
Size Complexity (SC)	0.72	0.870	0.720
Alignment Complexity (TAC)	0.84	0.739	0.640
Density Complexity (DC)	0.80	0.519	0.423
Grouping Complexity (GT)	0.88	0.368	0.171
Balance Complexity (TBC)	0.76	0.371	0.236
Overall Screen Layout Complexity (LC)		**0.567**	**0.431**
Complexity Score		**6**	**4**

CHAPTER 5. EXPERIMENTS

Most of metric-models have lack empirical validation of their effectiveness to evaluate and predict the user interface quality. Therefore, to validate the metrics-model and the GUIEvaluator tool, three experiments have been performed. First, validation of the metrics-model and GUIEvaluator. In this experiment, I investigated the effectiveness of GUIEvaluator and the metrics-model to measure the complexity of user interfaces and predict users' ratings. Second, effectiveness of GUIEvalutor for selecting the better alternative GUI design. Third, effectiveness of GUIEvalutor for measuring the GUI usability. These experiments provide empirical validation of the effectiveness of the metrics-model to evaluate the complexity and usability of user interfaces.

The main assumption in this dissertation is: Complex user interface leads to low usability rating by users and more time required for users to extract information from user interfaces. Several research studies have supported this assumption [3] [30] [81] [82] [83]. Therefore, the user interface complexity value has been considered as an inverse measure of user interface usability [2] [28].

5.1. Methodology

5.1.1. Participants

The study was conducted at North Dakota State University. As shown in Table 5.1, the participants were 50 student volunteers (17 females, 33 males), where 80% of participants were graduate students and 20% were undergraduate students. The participants who are majoring in computer science or related fields were 50% and 50% were from other majors. The data that were collected about the participants show that 46% of participants reported having no

Table 5.1. Participant characteristics

Characteristic	Measure	Number of Participants	Percentage of Participants
Gender	Male	33	0.66
	Female	17	0.34
Age	18-25	19	0.38
	26-35	30	0.60
	36-45	1	0.02
Major	Computer Science and Related Fields	25	0.50
	Other	25	0.50
Education Qualifications	Graduate	40	0.80
	Undergraduate	10	0.20
User Experience in software development	No Experience	23	0.46
	1-2 Years	12	0.24
	3-5 Years	8	0.16
	6 Years and Above	7	0.14

experience in software development. But 24%, 16%, and 14% of participants reported having one or two years, three to six years, and six years and above, respectively.

5.1.2. User Interfaces of Analysis

In the three experiments, 40 user interfaces were used as objects for this study from various sources. All the user interfaces have been developed in Visual Basic 2012. The number of widgets, the number of groups, and the number of grouped and ungrouped widgets have been used as criteria to classify the user interfaces into Class A, Class B, and Class C.

Briefly, the user interfaces were categorized into three classes in terms of the number of widgets as the following: Class A (number of widgets <= 25), Class B (26 =< number of widgets <=50), and Class C (number of widgets >=51). In terms of number of groups, the user interfaces

were categorized as the following: Class A (number of groups <= 1), Class B (2 < number of groups <=4), and Class C (number of groups >=5). Another way to categorize the user interfaces is using the number of grouped and ungrouped widgets. Moreover, using the number of grouped widgets helps us classify the user interfaces into Class A (number of grouped widgets <= 19), Class B (20 < number of grouped widgets <=38), and Class C (number of grouped widgets >=39). Furthermore, using the number of grouped widgets helps to sort the user interfaces into Class A (number of ungrouped widgets <= 10), Class B (11 < number of ungrouped widgets <=20), and Class C (number of ungrouped widgets >=21). All of these criteria must be satisfied for a user interface to be used in the experiments. Number of widgets is used as the primary criterion to categorize the user interfaces. Table 5.2 shows, for each user interface category, the number and percentage of user interfaces.

5.1.3. Data Collection

This section presents a brief description of data collected during the study. A survey application has been developed to collect the data. The study took approximately 55 minutes. To exploit the power of empirical evaluation, three experiments have been performed: Experiment 1 validates the GUIEvaluator and the metrics-model; Experiment 2 examines the effectiveness of

Table 5.2. The user interfaces for analysis and associated data

Number of Widgets			**Number of Groups**			**Number of Grouped Widgets**			**Number of Ungrouped Widgets**		
Category	**# of Widgets**	**%**	**Category**	**# of Groups**	**%**	**Category**	**# of GWidgets**	**%**	**Category**	**# of UGWidgets**	**%**
Class A <=25	14	35	Class A <=1	12	30	Class A <=19	13	32.5	Class A <=10	19	47.5
Class B 26-50	13	32.5	Class B 2-4	14	35	Class B 20-38	13	32.5	Class B 11-20	8	20
Class C >=51	13	32.5	Class C >=5	14	35	Class C >=39	14	35	Class C >=21	13	32.5

GUIEvalutor for selecting the better alternative GUI design; and Experiment 3 investigates the effectiveness of GUIEvalutor for measuring the GUI usability.

At the beginning of the user study, participants have been asked to provide background information. I was interested to know whether there were participants who had previously developed software applications or not. The participant had the decision to select, randomly, which experiment to perform first. The three experiments should be performed by each participant.

Experiment 1 has been run as follows. First, the participants have been provided by examples that explain the design factors and how to rate them. Second, the participants selected randomly the interface from a list provided in the survey application. And then, the participants were asked to rate the five design factors (alignment, grouping, density, balance, and size) and the user interface design overall using a 7-point Likert scale. The participants had to evaluate all the interfaces. Third, to obtain the weight for each of the five complexity measures in Eq.4.16, the participants rated the importance of each measure.

In Experiment 2, there were nine pairs of user interfaces. Each pair consists of two alternative GUI designs for the same purpose. The participant selected the preferred user interface design from each pair of user interfaces. Participants used a 7-point Likert scale, which is provided by the survey application, to provide their subjective evaluation.

In Experiment 3, the participants rated 18 user interfaces based on the following GUI usability measures: Ease to use, attractiveness, usefulness, and overall satisfaction of user interface. These measures were rated by participants using a 7-point Likert scale, which is provided by the survey application. The data have been, statistically, collected and analyzed using t-test, Pearson correlation test, and 2 sample two-tailed proportion test, especially the z-

test. The statistical results provide statistical evidence of the effectiveness of the metrics-model and its tool. The results are shown in the following subsections in detail.

5.2 . Experiment 1: Validation of GUIEvaluator Tool and its Metrics

5.2.1. Results and Analysis

The major objective of this experiment is to investigate the usefulness and effectiveness of the GUIEvaluator and its metrics-model in evaluating the user interface complexity. Therefore, the investigation process addresses the following two research questions:

RQ1: Is the GUIEvaluator effective to measure the complexity of a given user interface?

RQ2: How do the structural measures of user interface affect the interface complexity rating? The hypotheses associated with RQ1 are:

H1: given a specific user interface, the means of interface complexity for the user rating and the GUIEvaluator are not equal.

H2: given a specific user interface, there is a strong positive correlation between the user rating and the GUIEvaluator in terms of interface complexity value.

To test these two hypotheses, the t-test was performed on (*H1*) and the Pearson correlation test on (*H2*) for both the user rating and the GUIEvaluator, at a significance level of 0.01, on 18 interface layouts. The data have met the assumptions for a t-test: First, the data are continuous which were collected based on the 7-point Likert scale. Second, the participants are randomly sampled and the user interfaces are randomly selected in this experiment. Third, the sample sizes are equal for the two populations (n=18). Finally, the collected data are normally distributed. If the t-test does not show a significant difference between the subjective rating and the GUIEvaluator rating, we can accept the null hypothesis and conclude that the means of variances of the two populations are equal.

Table 5.3 shows that the means of the user rating and the GUIEvaluator are 0.561 and 0.575, respectively. Furthermore, to examine the hypothesis (*H1*), the t-test was performed for (*H1*) as shown in Table 5.3, df=32.39, for a significance level of 0.01. It shows there is no difference between the means of interface complexity of the user rating and the GUIEvaluator. In addition, to test the hypothesis (*H2*), the Pearson correlation testwas performed. In Table 5.3, the R value (0.804) shows a strong positive correlation between the user rating and the GUIEvaluator at a significance level of 0.01.

Figure 5.1 presents the complexity values of 18 user interface layouts, which were rated by both the participants and the GUIEvaluator. Strong similarities can be observed between the complexity values for both the user rating and the GUIEvaluator. Therefore, GUIEvaluator can be utilized to accurately evaluate the complexity of user interfaces.

Table 5.3. Pearson correlation test and t-test results between the user rating and the GUIEvaluator

	Mean	**R**	**p-val.**	**t-val.**	**df**	**p-val.**
User Rating	0.561	0.804	<0.00001	0.366	32.39	0.7166
GUIEvaluator Rating	0.575					

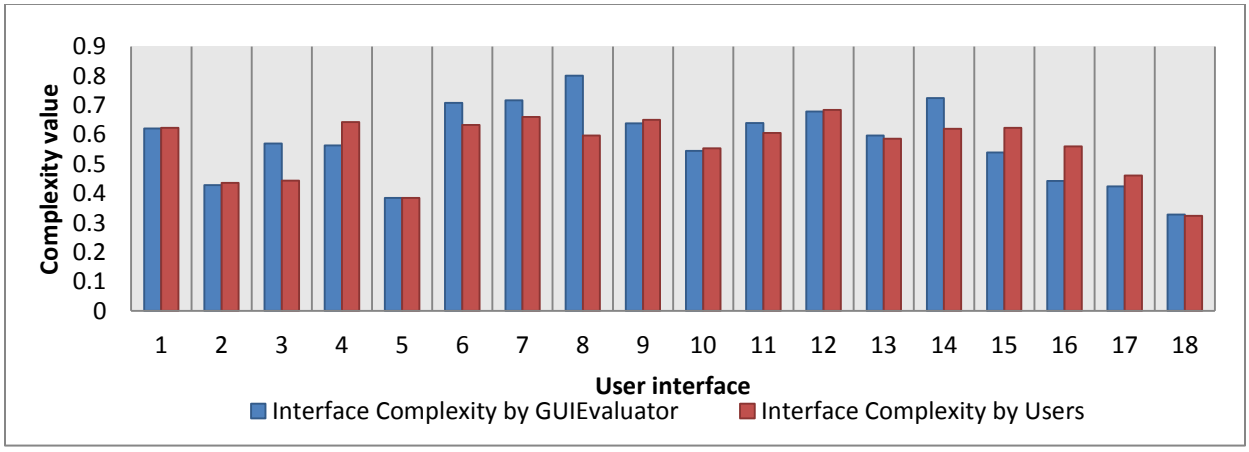

Figure 5.1. Comparison of the interface complexity values of the user rating and GUIEvaluator

The hypotheses associated with RQ2 are:

H3: given a specific user interface, the size value is strongly correlated with interface complexity values given by both the users and the GUIEvaluator.

H4: given a specific user interface, the alignment value is strongly correlated with interface complexity values given by both the users and the GUIEvaluator.

H5: given a specific user interface, the density value is strongly correlated with interface complexity values given by both the users and the GUIEvaluator.

H6: given a specific user interface, the grouping value is strongly correlated with interface complexity values given by both the users and the GUIEvaluator.

H7: given a specific user interface, the balance value is strongly correlated with interface complexity values given by both the users and the GUIEvaluator.

To test these hypotheses, the Pearson correlation test and the t-test were performed for both the user rating and the GUIEvaluator with the five complexity measures, at a significance level of 0.01, on 18 screen layouts. Table 5.4 shows the results of the Pearson correlation test and t-test for the values of the five complexity measures and the values of interface complexity given by both the user rating and the GUIEvaluator. On the one hand, Table 5.4 shows that there is a strong positive correlation between the user rating and the GUIEvaluator and the following design factors at a significance level of 0.01: Size, alignment, density, and balance. On the other hand, the grouping factor has a weak positive correlation with the user rating and the GUIEvaluator with R values 0.462 and 0.095, respectively. Therefore, the hypotheses (*H3, H4, H5,* and *H7*) can be accepted, but we fail to accept the hypothesis (*H6*).

Figure 5.2 compares the values of five complexity measures (size, alignment, density, grouping, and balance) with the interface complexity values of the user rating. From Fig. 5.2 we

have observed the following: First, the interface complexity is strongly affected by the size measure. The findings show that the size measure has a strong correlation with the interface complexity that was rated by the participants, but the participants rated the size measure less important compared with the rest of the measures. Second, another surprising result was that the grouping measure had the highest importance according to the user rating, but the findings show that the grouping measure has the lowest correlation with the values of interface complexity that are rated by the participants. Third, the value of interface complexity increases as the values of balance, density, and alignment measures increase. To sum up, the values of structural measures are consistent with overall interface complexity. Figure 5.3 compares the values of five complexity measures (size, alignment, density, grouping, and balance) given by the participants with the interface complexity values of the GUIEvaluator. By analyzing Fig. 5.3, a strong resemblance has been observed with the results from Fig.5.2. Both results are consistently correlated. Thus, we can claim that the metrics-model is effective to evaluate the complexity of GUIs.

Table 5.4. Pearson correlation test and t-test results for the five interface design factors with the user rating and the GUIEvaluator tool

Interface Design Factor	**User Rating**				**GUIEvaluator Rating**			
	Pearson Correlation Test		**t-test**		**Pearson Correlation Test**		**t-test**	
	R	p-val.	t-val.	p-val.	R	p-val.	t-val.	p-val.
Size	0.943	<0.00001*	-0.731	0.470	0.767	<0.001*	-1.013	0.319
Alignment	0.836	<0.00001*	-0.780	0.442	0.662	0.0028*	-1.005	0.323
Density	0.683	0.00180*	-0.772	0.445	0.585	0.0098*	-1.050	0.302
Grouping	0.462	0.05370	-2.608	0.014	0.095	0.7080	-2.657	0.012
Balance	0.825	<0.00001*	-0.560	0.579	0.705	0.0011*	-0.857	0.398

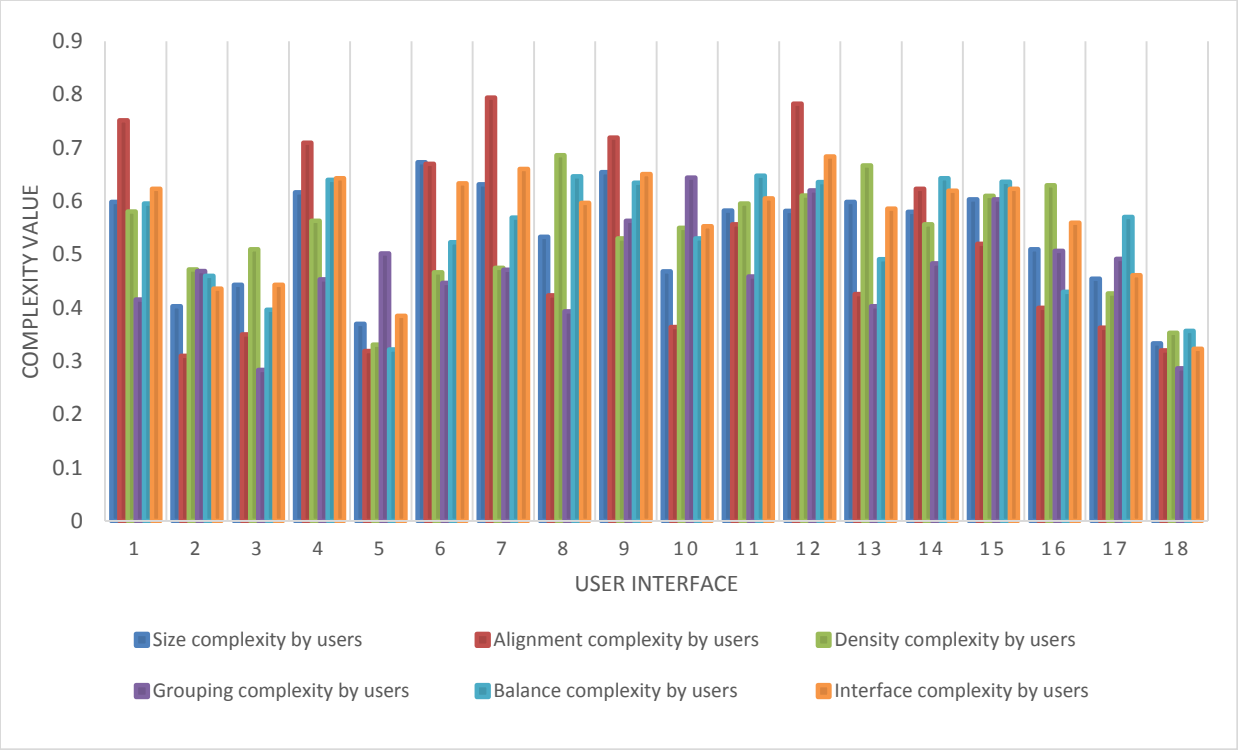

Figure 5.2. Comparison of the values of the five complexity measures with the interface complexity values of the user rating

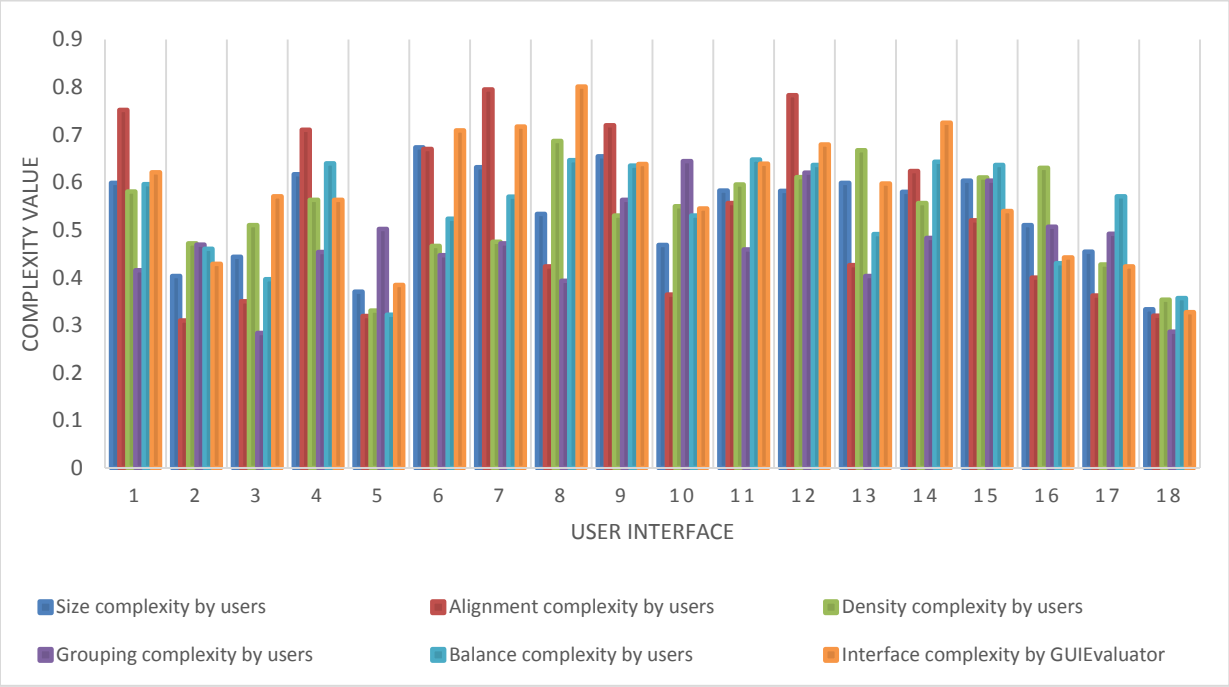

Figure 5.3. Comparison of the values of the five complexity measures with the interface complexity values of the GUIEvaluator

5.2.2. Discussion

As previously outlined, the interface complexity values, which are rated by both the participants and the GUIEvaluator, are consistent. Furthermore, the findings show that the proposed metrics-model is useful to evaluate user interfaces, but these metrics do not have equal magnitude of importance. I investigate that the grouping measure has a weaker correlation with complexity values for both the user rating and the GUIEvaluator. Perhaps the reasons behind this are: (1) Misunderstanding the object-grouping on the screens, (2) 46% of participants have no experience in software development, or (3) The grouping metric may be insufficient to measure the widgets grouping.

In Table 5.3, the findings show that the R value is 0.804 between the user rating and the GUIEvaluator for 18 interface layouts. However, with the interface layouts 3, 8 and 14, there is a non-trivial difference between the interface complexity values for both the user rating and the GUIEvaluator. The complexity measure that causes this difference is the objects-grouping given by user rating. The values of grouping are less than the average of the values of other complexity measures for the interface layouts 3, 8, and 14. This encourages us to focus on grouping measure to investigate the causes behind the variance of the values between the user rating and the GUIEvaluator. In summary, whether we use GUIEvaluator or user rating to evaluate the interface complexity, we reach the same conclusion. Therefore, the findings confirm the effectiveness of GUIEvaluator and its metrics model to be used during the early stages in software development.

5.3. Experiment 2: Effectiveness of GUIEvaluator for Selecting the Best Alternative GUI Design

5.3.1. Results and Analysis

The objective of Experiment 2 is to investigate the effectiveness of the metrics-model and its tool to determine the best GUI layout among alternative GUI designs. Therefore, the investigation was performed to address this research question:

RQ3: To what extent can the metrics-model and its tool determine which is the best GUI design among a set of alternative GUI designs?

The hypothesis associated with RQ3 is:

H8: for a given pair of user interfaces, the means of the subjective rating and the GUIEvaluator rating are not equal.

To examine, statistically, hypothesis (*H8*), the z-test was performed for both the user rating and the GUIEvaluator rating, at a significance level of 0.05, on nine pairs of user interfaces. The data that were collected in this experiment met the assumptions of the z-test. Those assumptions are: The user rating and GUIEvaluator rating populations are independent of one another. Moreover, the user rating and GUIEvaluator rating populations are normally distributed and the number of values is 50. The z-test has been utilized in this experiment to determine whether the difference between the user rating proportion and GUIEvaluator rating proportion is significant or not. In addition, if there is no significant difference between the two populations, the z-test fails to reject the null hypothesis, which assumes that the means of two sample proportions are equal.

Table 5.5 shows, statistically, that there is no significant difference between the user rating and GUIEvaluator proportions, at significance level 0.05, for eight out of nine pairs of user

Table 5.5. The results of two sample z-test of user rating proportion and GUIEvaluator proportion at a significance level 5%

Pair of User Interfaces	User Rating Proportion	GUIEvaluator Proportion	Difference	z-value	p-value
UI1 and UI2	0.16	1	0.84	3	*0.0032**
UI3 and UI4	0.96	1	0.04	0.3	0.7730
UI5 and UI6	0.66	1	0.34	1	0.3148
UI7 and UI8	0.64	1	0.36	1	0.2940
UI9 and UI10	0.94	1	0.06	0.4	0.7212
UI11 and UI12	0.76	1	0.24	0.8	0.4296
UI13 and UI14	0.94	1	0.06	0.4	0.7212
UI15 and UI16	0.54	1	0.44	1.3	0.1990
UI17 and UI18	0.70	1	0.30	0.9	0.3585

interfaces. Therefore, we can accept the null hypothesis that the user rating and GUIEvaluator proportions are equal. In addition, Table 5.5 shows that the difference range between the two proportions of eight pairs of user interfaces is 0.06 to 0.44. This difference range provides sufficient evidence that more than 50% of participants have supported the GUIEvaluator rating of given user interfaces. Therefore, the results provide evidence that supports the acceptance of (*H8)*. But the data show that the only one pair of user interfaces has a significance difference between the user rating and GUIEvaluator proportions (*p=0.0032*). Table 5.6 shows which user interface has been preferred by both the participants and the GUIEvaluator among nine pairs of alternative user interfaces.

5.3.2. Discussion

The results strongly support the conclusion that the metrics-model and GUIEvaluator can be used to determine which user interface is the best among a set of user interface designs. I found that the accuracy of the GUIEvaluator in the evaluation of user interfaces is 88.8%.

Table 5.6. The results of users' preferences and GUIEvaluator among nine pairs of alternative user interfaces

Pair of User Interfaces	By Users' Rating	By GUIEvaluator	Match/Not Match
UI1 and UI2	UI2	UI1	Not Match
UI3 and UI4	UI3	UI3	Match
UI5 and UI6	UI6	UI6	Match
UI7 and UI8	UI7	UI7	Match
UI9 and UI10	UI9	UI9	Match
UI11 and UI12	UI11	UI11	Match
UI13 and UI14	UI14	UI14	Match
UI15 and UI16	UI15	UI15	Match
UI17 and UI18	UI17	UI17	Match

These findings provide sufficient evidence regarding the usefulness of GUIEvaluator in the user interface evaluation. But the data were collected about the first pair of user interfaces show a significant difference between the subjective evaluation and the GUIEvaluator rating. The causes of this difference are. First, all widgets on the UI1 are not grouped while all widgets on the UI2 are grouped. On the one hand, the GUIEvaluator set 0 to the grouping complexity if widgets are not grouped. On the other hand, the participants may not take into account the grouping factor when they evaluated this pair of interfaces. Further discussion of the grouping factor is available in the Experiment 1.

Second, the values of alignment factor are inconsistent as well. The widgets on UI1 are organized into two columns while the widgets on the UI2 are organized into four columns. Therefore, the number of vertical and horizontal points on the UI1 is less than the number of vertical and horizontal points on the UI2. These reasons may lead to that difference.

To sum up, whether the GUIEvaluator or the user rating is used to determine the best GUI design among alternative GUI designs during the design phase, we reach the same conclusion. Therefore, the findings confirm the effectiveness of GUIEvaluator and its metrics-model to be used during the early stages in software development.

5.4. Experiment 3: Effectiveness of GUIEvalutor for Measuring the GUI Usability

The findings of the above two experiments show the effectiveness of the metrics-model and its tool for the user interface evaluation. Another interesting GUI aspect is the GUI usability, which needs to be investigated. In this experiment, I focus on three factors influencing GUI usage [71][72][73]. These factors are: User interface attractiveness, easy to use, and user interface usefulness. These factors have been utilized widely in many usability studies for various purposes [74] such as mobile payment, website usage, technology acceptance, e-learning, and so forth.

The above usability factors have been validated conceptually and empirically through a number of research studies [75][76]. Therefore, this experiment was designed and performed in order to investigate the effectiveness of the metrics-model and its tool to predict the GUI usability utilizing user interface attractiveness, easy to use, and user interface usefulness. In addition, these design factors are used to predict the overall user satisfaction for a given user interface.

5.4.1. Results and Analysis

A controlled experiment has been designed and performed considering the following research question:

RQ4: For a given user interface, is there a significant correlation between the GUI usability factors given by users and the GUI rating given by the metrics-model and its tool?

The hypothesis associated with RQ4 is: *(H9)* given a specific user interface, there is a significant correlation between the GUI usability factors provided by users and the GUI rating provided by the metrics-model and its tool. To test this hypothesis, the Pearson correlation test has been performed for 12 user interfaces, at significance level 0.05. Table 5.7 shows the findings of this analysis of users' preferences, in terms of GUI usefulness, easy to use, and attractiveness, with user satisfaction and GUIEvaluator rating.

The results show a very strong positive correlation between the above usability factors and the user satisfaction. The usefulness and attractiveness of user interfaces and the users' satisfaction have the strongest positive correlation with the value of R is 0.9751 (*P<0.00001*). But the value of R is 0.9171, at a confidence level of (*p=0.00003*), between Easy to use factor and the users' satisfaction. These findings provide a demonstration of the factors validity to measure the user satisfaction. Therefore, these factors can be used in order to validate the metrics-model and its tool to predict the usability of user interfaces.

The results of data analysis show that there is a strong positive correlation between the three usability factors and the GUIEvaluator rating. According to Table 5.7, the strongest positive correlation is shown between the attractiveness factor and the GUIEvaluator rating, where the value of R is 0.8691 at a significance level of (*p=0.0002*). Furthermore, the usefulness

Table 5.7 The results of pearson correlation test of the GUI usability factors with the users' satisfaction and GUIEvaluator rating

GUI Usability Factor	**Users' Satisfaction**		**GUIEvaluator Rating**	
	Pearson Correlation Test		**Pearson Correlation Test**	
	R	p-val.	R	p-val.
Usefulness	0.9751	<0.00001*	0.7678	0.0030*
Easy to Use	0.9171	0.00003*	0.7180	0.0080*
Attractiveness	0.9751	<0.00001*	0.8691	0.0002*

and easy to use factors have a strong positive correlation with the GUIEvaluator rating though not as strong as the attractiveness factor. The R values, for both usefulness and easy to use, are 0.7678 and 0.7180, respectively, at significance levels (0.003) (0.008), respectively. Considering the R and p-values for each of the above mentioned usability factors and user satisfaction and GUIEvaluator rating, the hypothesis (*H9*) is supported.

Scatter plots have been used to provide an overview of the collected data and examine the possible relationships between the usability factors and both the users' satisfaction and the GUIEvaluator ratings. Figure 5.4 shows the user interface attractiveness and user satisfaction of 12 user interfaces, which were evaluated by 50 participants. The scatter plot illustrates the positive trend between the attractiveness factor and users' satisfaction, as the value of attractiveness factor increases, the users' satisfaction values increase. Then, Figure 5.5 shows the user interface usefulness and users' satisfaction and Figure 5.6 shows that the easy to use factor and user satisfaction of 12 user interfaces, which were evaluated by 50 participants. According to these scatter plots, both the values of usefulness and easy to use factors have similar trends to increase with the users' satisfaction values. The scatter plot illustrates the positive trend between the easy to use and usefulness factors and users' satisfaction.

To sum up, the graphs illustrate the shared trends among the three design factors and users ratings. From this, we can inform that the three design factors are useful and effective to evaluate the usability of user interfaces. This provides a statistical evidence of their effectiveness to predict the usability of user interfaces. Therefore, the correlation among these design factors and GUIEvaluator rating will be investigated in order to prove the effectiveness of the metrics-model to predict the usability of user interfaces.

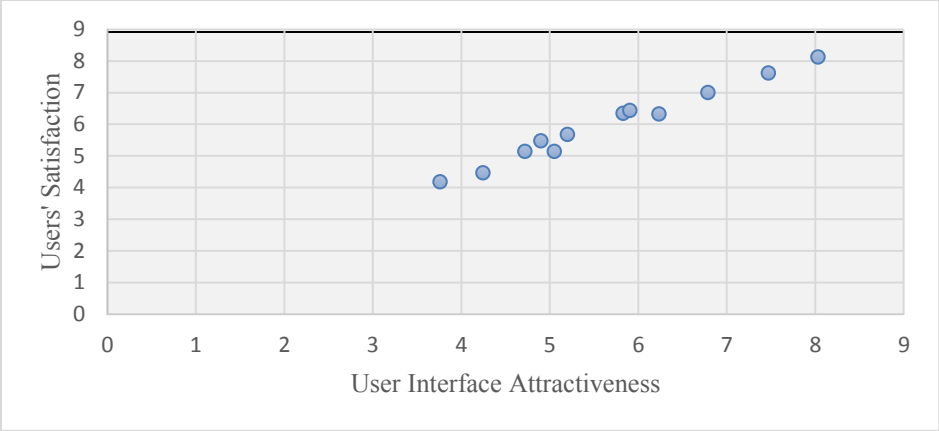

Figure 5.4. The correlation between the values of user interface attractiveness and users' satisfaction of 12 user interfaces

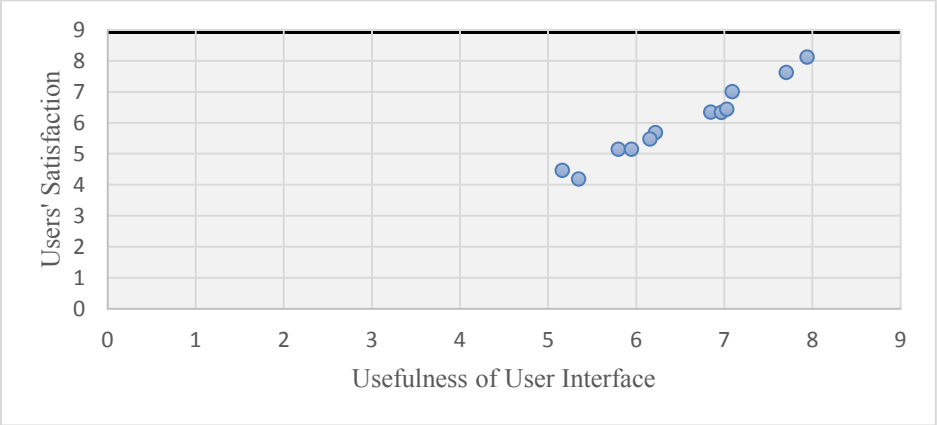

Figure 5.5. The correlation between the values of usefulness of user interface and users' satisfaction of 12 user interfaces

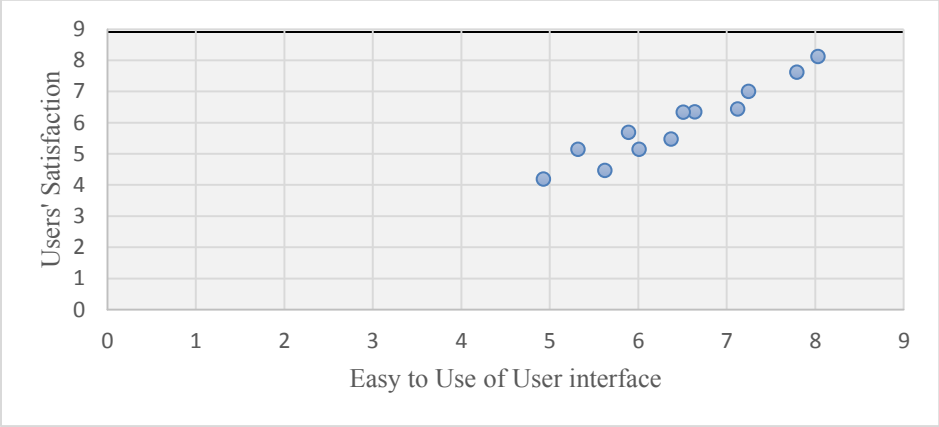

Figure 5.6. The correlation between the values of easy to use factor and users' satisfaction of 12 user interfaces

Figure 5.7 shows the user interface attractiveness and GUIEvaluator rating of 12 user interfaces, which were evaluated by 50 participants. The scatter plot illustrates the positive trend between the attractiveness factor and GUIEvaluator rating, as the values of attractiveness factor increase, the GUIEvaluator ratings increase. Thus, a linear relationship and a positive trend are shown, in the scatter plot, between the user interface attractiveness and GUIEvaluator rating.

Figure 5.8 shows the user interface usefulness and GUIEvaluator rating and Figure 5.9 shows the easy to use factor and GUIEvaluator rating of 12 user interfaces, which were evaluated by 50 participants. The trends observed between the values of usefulness and easy to use factors and the values of GUIEvaluator rating indicate strong positive correlations. Thus, the scatter plots describe the relationship between the GUIEvaluator rating and both usefulness and easy to use factors is linear and has a positive trend.

The graphs show a strong positive correlation between the three usability factors and GUIEvaluator ratings. This proves the effectiveness of the metrics-model and its tool to predict the usability of user interfaces based on the structural aspects of these user interfaces.

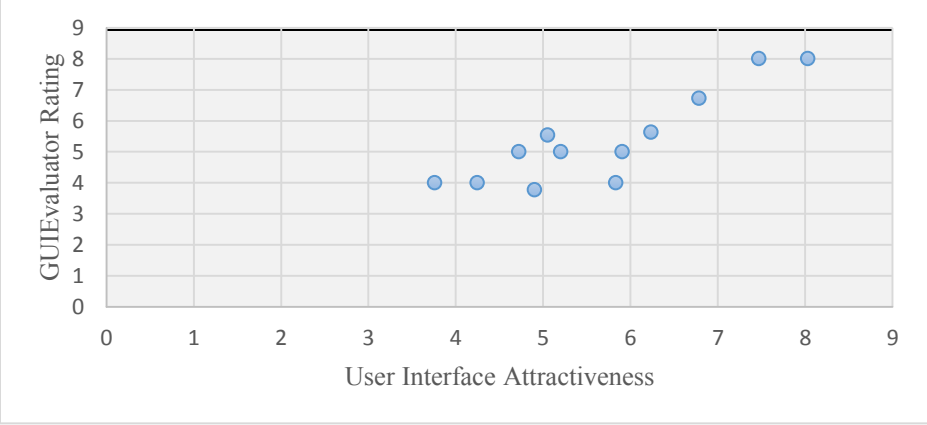

Figure 5.7. The correlation between the values of user interface attractiveness and GUIEvaluator rating of 12 user interfaces

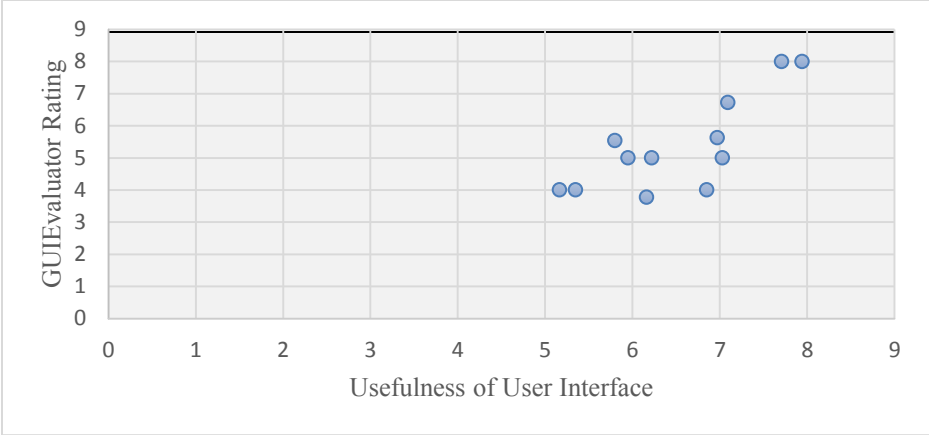

Figure 5.8. The correlation between the values of usefulness of user interface and GUIEvaluator tool of 12 user interfaces

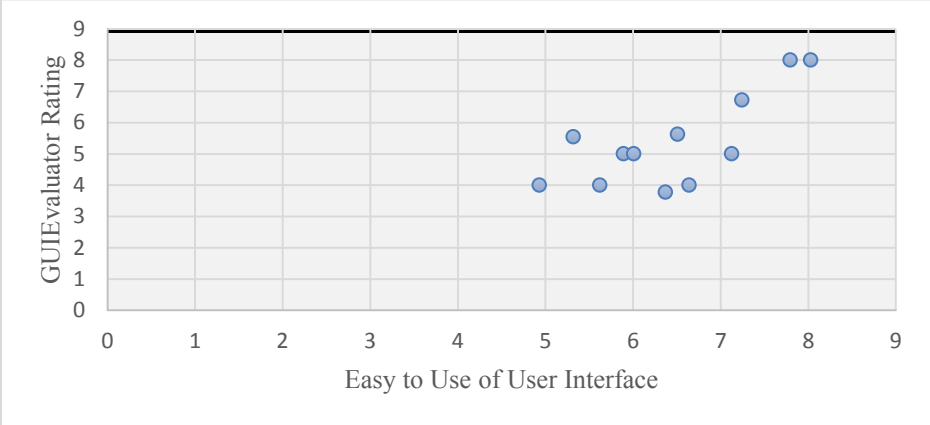

Figure 5.9. The correlation between the values of easy to use factor and GUIEvaluator tool of 12 user interfaces

5.4.2. Discussion

As the scatter plots illustrate, the GUIEvaluator has a positive and strong trend with the usability factors. These findings provide further evidence of the effectiveness of the metrics-model and GUIEvaluator tool to evaluate user interfaces from the GUI usability perspective. In addition, these findings underscore an importance of the further investigation of using software metrics in the GUI evaluation process. For instance, there is need to combine the GUI complexity metrics with semantic aspects to provide a more accurate evaluation of GUIs.

However, ultimately, more comprehensive metric-models will be effective tools to evaluate graphical user interfaces.

5.5. Threats to Validity

In general, any user study is subject to threats to validity. Therefore, these threats must be taken into account in order to estimate their impact on the findings of these user studies. In this section, the internal, external, and construct threats to the validity of the findings of the experiments have been described.

5.5.1. Internal Validity

One of the popular internal threats is the potential faults of the software tools that were used for statistical analysis. This threat may affect the inferences that are made regarding the effectiveness of the metrics-model and GUIEvaluator. To avoid this threat, I validated these tools that were used in the Experiments 1, 2, and 3 compared to similar purpose tools. For instance, the statistical tests were performed using R language, and then the results were compared with the results of other online statistical packages for the same data such as [69][70]. Moreover, this could increase the confidence in the obtained results.

Another possible internal threat is that of the potential faults of the GUIEvaluator, which is used to support the metrics calculations. This threat may have an effect on the conclusions that are made by the tool regarding the evaluation of user interfaces. To avoid this threat, a pilot experiment was conducted to validate the metrics-model compared with subjective evaluation of a number of user interfaces. Furthermore, I compared the metrics values that are calculated manually for a set of user interfaces with the metrics values that are computed by the GUIEvaluator for the same user interfaces. This comparison provides evidence about the accuracy of the metrics calculations by GUIEvaluator.

5.5.2. External Validity

I focus on the user interface representativeness issue that affects the generalization of the findings. Fourty user interfaces have been used, which may limit the external validity of the findings of the experiments. To avoid this threat, I used user interfaces with various numbers of widgets, from 15 to 85. This helps us categorize the user interfaces into three classes (Class A, Class B, and Class C). To control this threat, there is a need for additional experiments with the wider population of user interfaces. Moreover, these user interfaces represent different domains. Another issue is that of the user representativeness. Fifty participants have been recruited to conduct the experiments. The number of participants could be relatively small. In order to control this threat, additional experiments can be conducted with the wider population of users.

5.5.3. Construct Validity

The complexity measures which were utilized in this research are not the only possible GUI layout complexity measures. I focused on five common structural measures (alignment, grouping, balance, density, and size) to evaluate the GUI layout complexity. To avoid this threat, future studies will be performed using other complexity measures and semantic measures as well.

CHAPTER 6. COMPARISON WITH THE EXISTING GUI COMPLEXITY METRICS

A comparative analysis has been performed between the metrics-model and the existing GUI complexity metrics models. The models that were considered in this comparison meet those two criteria: First, they have been used to evaluate properness of user interfaces taking into account the GUI complexity. Second, they were proposed based on the structural aspects of user interfaces. The models that meet these two criteria were proposed in [9] [10] [13] [12] [17] [84-86]. The results of these models have been compared with users' preferences, which were collected through two controlled experiments.

6.1. The Existing GUI Complexity Metric-Models

This section explains the collected complexity metrics for the Experiment 1 and 2 in this chapter.

1 - Layout Uniformity (LU) Metric

This metric was originated by Costantine [86] then it was redefined and studied by Montero et al. [85] to develop GUILayout++ tool, which is used to produce user interface prototypes through iterative designs based on user evaluation. Layout Uniformity is a structural metric that measures the spatial arrangement of user interface widgets without taking into account type of widgets and how they are used. LU metric is formulated based on the rationale that highly disordered user interface can hinder software usability. LU metric focuses on the size and alignment of widgets on the user interface. Equation 6.1 and 6.2 show how calculate LU. Where Nco is total number of widgets on the user interface. Nh and Nw are number of different heights and widths, respectively. Nt, Nl, Nb, and Nr represents number of different top-edge,

left-edge, bottom-edge, and right edge, respectively. Also, M is the minimum value of sizes and alignments needed to adjust the value of LU range from 0 to 100.

$$LU = 100 * (1 - \frac{(Nh+Nw+Nt+Nl+Nb+Nr)-M}{6 * Nco-M} \quad \text{(Eq.6.1)}$$

$$M = 2 + 2\,[2\,\sqrt{N\ components}\,] \quad \text{(Eq.6.2)}$$

LU is a useful metric for the designers who have lack to know when the interface layout might be improved. In general, a value of Layout Uniformity anywhere between 50% and 85% is considered reasonable.

2 – Complexity Metrics of Screen Layout by Fu

This metrics model computes four complexity metrics: Alignment, size, grouping, and local density [9]. Size Complexity (SC) is calculated based on the variety of sizes for each widget type on the user interface as shown in Eq.6.3. Where n_{size} is the number of sizes and n is the total number of widgets. Local Density (LD) measures the extent to which the user interface is occupied by visual widgets. The local Density metric was calculated taking into account the rationale that the widgets density of given user interface is restricted to the optimal density value 50% of that user interface. Equation 6.4 illustrates how the LD metric can be calculated, where a_i is the area of widget i, a_{frame} is the area of the frame, and n is the number of widgets on the frame.

$$Size\ Complexity\ (CS) = 1 - \frac{\sum_{I}^{type}(n\ size-1)}{n} \quad \in [0,1] \quad \text{(Eq.6.3)}$$

$$Local\ Density\ (LD) = 1\text{-}2\left|0.5 - \frac{\sum_{i}^{n} ai}{a\ frame}\right| \quad \in [0,1] \quad \text{(Eq.6.4)}$$

Grouping Complexity (CG) measures the extent to which widgets on a user interface are shown visually as one piece. Those widgets have the same function are surrounded by a boundary, such as the line or background color. According to Eq.6.5, Grouping Complexity metric is calculated as follows:

$$Grouping\ Complexity\ (CG) = \frac{gi}{g} \in [0,1] \quad \text{(Eq.6.5)}$$

Where g_i is the number of groups with clear boundary by line, background, color, or space and g is the total number of groups. Alignment for Simplicity (AS) metric computes the number of horizontal and vertical alignment points for widgets on a user interface. Equation 6.6 explains how to calculate AS metric, where n_{vap} and n_{hap} are the number of vertical and horizontal alignment points, respectively. Where n is the number of widgets on the frame.

$$Alignment\ for\ Simplicity\ (AS) = \frac{3}{nvap+nhap+n} \in [0,1] \quad \text{(Eq.6.6)}$$

Finally, Screen Complexity (SC) is the average of the four aforementioned metrics as shown in Eq.6.7.

$$SC = (AS + CS + LD + CG) / 4 \quad \text{(Eq.6.7)}$$

3- Complexity Model by Parush

This model was introduced by Parush et al. [10] for evaluation of graphical user interfaces (GUIs). This model was developed based on four design factors: Size, local density, Alignment, and grouping, which are used to calculate GUI complexity. Each design factor was weighted based on subjective judgment. These weights are 0.23, 0.16, 0.32, and 0.28 for size, local density, alignment, and grouping, respectively. The design factors are illustrated as follows: Size factor, which categorizes widgets into groups according to their actual sizes. Therefore, the Size Complexity was calculated as shown in Eq.6.8. Where W_i is the weight of each widget type and C_{si} is the average size of each widget type.

$$Size\ Complexity = \sum_i^{type} Wi\ Csi \quad \text{(Eq.6.8)}$$

Complexity of Density was computed using standard deviation of local and overall density values. Where local density is defined as the percentage of used space in a group and overall density is defined as the percentage of used space in entire screen. Alignment Complexity and

Grouping Complexity were calculated taking into account number of horizontal and vertical dimensions for four cases: Frames and contour lines, widgets in the same group, widgets in all groups, and widgets outside the groups. Then, the resulted value will be multiplied by the weight, which is the number of widgets divided by the number of indention lines for both dimensions. Finally, Total Complexity, which is a sum of all of the complexity measures of each factor multiplied by the subjective weights found in the experiment.

4 – BaLOReS

Is a metric-model for evaluating the quality of GUI layouts. This metrics-model was proposed by Gonzalez et al. [13] [84]. Five design factors are identified in this model: Balance, Linearity, Orthogonality, Regularity, and Sequentiality. Equation 6.9 illustrates the balance factor (Ba). Where n_{hac} is the number of widgets horizontally aligned with other widgets, n_c is the number of widgets on the given user interface, h_l is the accumulated height of left side widgets, and h_r is the accumulated height of right side widgets.

$$Ba = \frac{nhac}{nc}\left(1 - \frac{hl-hr}{hl+hr}\right) \quad \text{(Eq.6.9)}$$

Both the horizontal and vertical alignments are widely used to calculate Linearity, Orthogonality, Regularity, and Sequentiality. In Eq.6.11, linearity can be calculated based on horizontal alignment of widgets. Where n_{hac} is the number of widgets horizontally aligned with other widgets, n_{va} is the number of different vertical alignments, and n_c is the number of widgets in the given user interface.

$$Li = \frac{nhac\ X\ nva}{nc^2} \quad \text{(Eq.6.10)}$$

Sequentiality is calculated based on vertical alignment of widgets using Eq.6.11. Where n_{vac} is the number of widgets vertically aligned with other widgets, n_{ha} is the number of different horizontal alignments, and n_c is the number of widgets in the given user interface. Orthogonality

is calculated based on horizontal and vertical alignment of widgets using Eq.6.12. Where $n_{hac,}$ n_{vac}, and n_c are defined in Eq.6.10 and Eq.6.11. Moreover, Regularity factor is calculated based on horizontal and vertical alignment of widgets using Eq.6.13, where n_s is the number of different shapes.

$$Se = \frac{nvac\ X\ nha}{nc^2} \quad \text{(Eq.6.11)}$$

$$Or = \frac{nvac\ X\ nhac}{nc^2} \quad \text{(Eq.6.12)}$$

$$Re = \frac{nvac\ X\ nhac\ X\ (nc-ns+1)}{nc^3} \quad \text{(Eq.6.13)}$$

In order to calculate the overall user interface score, the Eq.6.14 can be used for calculating the Composition metric, which involves the values of Balance, Linearity, Orthogonality, Regularity, and Sequentiality**.** Where P_i is the nonempty panel or area with metric defined, a_{pj} is panel area expressed in pixel^2, m_{pi} is the calculated metric in pi panel.

$$Composition = \frac{ap1\ X\ mp1+\cdots+api\ \ X\ mpi}{ap1+\cdots+api} \quad \text{(Eq.6.14)}$$

5 – Metric Model by Ngo

Ngo et al. [12] proposed 13 metrics for evaluation GUIs. These metrics were redefined and studied by Zen [17]. Five metrics have been used in the comparative study in this dissertation. These metrics are: Balance, Unity, Simplicity, Regularity, and Density. Unity metric is used to calculate the grouping while Simplicity and Regularity metrics are used to calculate the horizontal and vertical alignments. The above five metrics will be used to calculate the order of user interfaces.

- *Balance*

To calculate the balance metric, Eq.6.15 can be used, where BMvertical and BMhorizontal are the vertical and horizontal balances, respectively. BMvertical and

BMhorizontal are calculated using Eq.6.16 and Eq.6.17, respectively.

$$BM = 1 - \frac{|BMvertical|+|BMhorizontal|}{2} \in [0,1] \quad \text{(Eq.6.15)}$$

$$BMvertical = \frac{Wl-Wr}{max(|Wl|,|Wr|)} \quad \text{(Eq.6.16)}$$

$$BMhorizontal = \frac{Wt-Wb}{max(|Wt|,|Wb|)} \quad \text{(Eq.6.17)}$$

To calculate W for four sides, the Eq.18 has been used as follows:

$$Wj = \sum_i^{nj} aij * dij \text{ , } j = l, r, t, b \quad \text{(Eq.6.18)}$$

Where l, r, t, b are left, right, top, and bottom, respectively; aij is the area of widget i on side j; dij is the distance between the central lines of the widget and the frame; and nj is the total number of widgets on the side.

- *Unity*

Grouping factor is measured through Unity metric, which is calculated using Eq.6.19 as follows:

$$UM = \frac{\left|1-\frac{Nsize-1}{n}\right|+\left|1-\frac{Alayout-\sum_i^n Ai}{Aframe-\sum_i^n Ai}\right|}{2} \in [0,1] \quad \text{(Eq.6.19)}$$

Where A_i, A_{layout}, and A_{frame} are the areas of widget i, the layout, and the frame, respectively; N_{size} is the number of sizes used; and n is the number of widgets on the frame.

- *Simplicity*

To measure the Simplicity (SMM), they counted the number of different starting positions of widgets. Therefore, the horizontal and vertical alignments are used to calculate the Simplicity as shown in Eq.6.20. Where n_{vap} and n_{hap} are the numbers of vertical and horizontal points and n is the number of widgets on the screen.

$$SMM = \frac{3}{nvap + nhap + n} \in [0,1] \quad \text{(Eq.6.20)}$$

- *Regularity*

Regularity (RM) is calculated based on spaced horizontal and vertical alignment points. From the Eq.6.21, the regularity of user interface can be calculated as follows:

$$RM = \frac{|RMalignment|+|RMspacing|}{2} \in [0,1] \quad \text{(Eq.6.21)}$$

- *Density*

It is a percentage of widget positions on the entire user interface layout. Density (DM) is calculated in Eq.6.22, where A_i and A_{frame} are area of widget i and the user interface; and n is the number of widgets on the user interface.

$$DM = 1 - 2\left|0.5 - \frac{\sum_i^n Ai}{Aframe}\right| \in [0,1] \quad \text{(Eq.6.22)}$$

- *Order and Complexity metrics*

Order Metric (OM) is the aggregated sum of 13 measures. To perform OM calculation, Eq.6.23 is used to find the value of OM as follows:

$$OM = g\{\} = \frac{1}{m}\sum_i^{13} \propto iMi \in [0,1] \quad \text{(Eq.6.23)}$$

M_i values were not determined. Therefore, the 13 factors have the same weight and the weights are set to 1.

Table 6.1 shows the metrics-models that were used for evaluating the complexity and usability of user interfaces based on structural measures. These measures are: Alignment, grouping, size, density, and balance. The metrics-models by Fu and Parush can support only the calculations of alignment, grouping, size, and local density to measure the overall complexity of user interfaces. The metrics-models [9] [10] [13] [12] [17] [84-86] measure the usability of user interfaces. The metric-model by Ngo measures the complexity and usability of user interfaces based on alignment, grouping, density, and balance factors. This model has been used to develop

Table 6.1. Summary of the metric-models and their structural metrics

	Alignment	Grouping	Size	Density	Balance	Overall Complexity	Usability
GUIEvaluator	Y	Y	Y	Y	Y	Y	Y
Fu [9]	Y	Y	Y	Local Density	X	Y	X
Parush [10]	Y	Y	Y	Local Density	X	Y	X
Montero [85] Costantine [86]	Y	X	Y	X	X	X	Y
Zen [12] Ngo [17]	Y	Y	X	Y	Y	Y	X
Gonzalez [13] Gonzales [84]	Y	X	X	X	Y	X	Y

QUESTIM tool. BaLOReS measures the usability of user interface through alignment and balance factors. Furthermore, the usability of user interfaces was measured by the metrics-model of Costantine. This model was used to develop GUILayout++ tool taking into account the alignment and size calculations.

6.2. Experiments

To perform a comparative study among the complexity metrics-models for evaluating user interfaces, three experiments have been performed: First, validation of the complexity metrics-models for calculating, precisely, the complexity of a given user interface. In this experiment, the investigation shows that the effectiveness of each metric-model to measure the GUI complexity and predict users' ratings. Second, the effectiveness of the metric-models for selecting the best GUI design among alternative GUI designs meets user expectations. Third, the effectiveness of the metrics-models for measuring the usability of GUIs. These experiments provide empirical validation of effective metrics-model to evaluate the complexity and usability of user interfaces.

The methodology that has been used for conducting the comparative study was explained in Chapters 1 and 5. This methodology explains the participants' characteristics, user interfaces

that have been used for analysis, and how to collect the data. The data were, statistically, collected and analyzed using t-test, Pearson correlation test, and two sample two-tailed proportion tests, especially the z-test. The metrics calculation step has been added, for other existing metrics-models, to the methodology, where the metrics calculations, for each metrics-model, were performed precisely based on the formulas and description appeared in the published papers of these metric-models. The results of these experiments are analyzed and discussed in the following subsections in detail.

6.2.1. Experiment 1: Effectiveness of the Existing Metric-Models for Measuring the Complexity of User Interfaces

6.2.1.1. Results and Analysis

The major objective of this experiment is to investigate the useful and effective metric-model for evaluating the user interface complexity, accurately. Therefore, the investigation process addresses the research question as follows:

RQ5: Can GUIEvaluator and its metrics measure, accurately, the complexity of a given user interface more than the existing metric-models?

The hypothesis associated with RQ5 is:

H10: given a specific user interface, GUIEvaluator and its metrics can, accurately, measure its complexity more than the existing metric-models.

In order to test (*H10*), t-test and the Pearson correlation test were performed, at a significance level of 0.05, on 18 user interfaces. The data have met the assumptions for a t-test: First, the data are continuous which were collected based on the 7-point Likert scale. Second, the participants are randomly and user interfaces are randomly sampled in this experiment. Third, the sample sizes are equals for both two populations (n=18). Fourth, the collected data are

normally distributed. If the t-test does not show a significant difference between the subjective rating and the rating of metrics-model, the null hypothesis can be accepted. As a result, we can conclude that the means of variances of the two populations are equal.

Table 6.2 shows the means of complexity scores by the user rating and the following metrics-models: GUIEvaluator, Fu, Parush, and Ngo are 0.575, 0.3297, 0.594, and 0.5258, respectively. The t-test was performed as shown in Table 6.2 to test the hypothesis (*H10*). On the one hand, the findings show that there is no significant difference among the means of the participants' rating and the metrics-models: GUIEvaluator, Parush, and Ngo. On the other hand, the findings show that there is a significant difference between the means of the participants' rating and metrics-model by Fu. Furthermore, a strong positive correlation, with R value (0.804) at significance level 0.01, is found between the GUI complexity rating by GUIEvaluator and the participants' rating. The correlation among the metrics-models of Fu and Parush, with R values (0.2856 and 0.02852, respectively), and the participants' rating is a weak and positive while the complexity rating by Ngo has a weak negative correlation with the participants' rating with R value (-0.3706) at significance level of 0.05. According to the Table 6.2, the GUIEvaluator has the highest correlation value with the participants' rating for evaluating the complexity of user interfaces.

Figure 6.1 presents the complexity scores of 18 user interfaces, which are rated by the participants, GUIEvaluator, Fu, Parush, and Ngo. On the one hand, Strong similarities have been observed among the participants' evaluation and the GUIEvaluator and Ngo's metric-models. On the other hand, all the complexity scores by Fu model for all user interfaces are less than the complexity scores by the participants' evaluation. Overall, the means of Parush's model are, slightly, equal, but the complexity scores, individually, have dissimilarities with the complexity

Table 6.2. The results of pearson correlation test and t-test between the participant rating with mean = 0.561 and the metric-models in terms of the complexity of user interfaces

Metrics-Model	GUI Complexity Rating by the Participants				
	Pearson correlation test			t-test	
	Mean	R	p-val.	t-val.	p-val.
GUIEvaluator	0.575	***0.804****	<0.00001	0.366	**0.7166***
Fu	0.3297	0.2856	0.251	6.682	<0.00001
Parush	0.594	0.0252	0.921	0.3437	**0.7332***
Ngo	0.5258	-0.3706	0.1296	1.245	**0.2212***

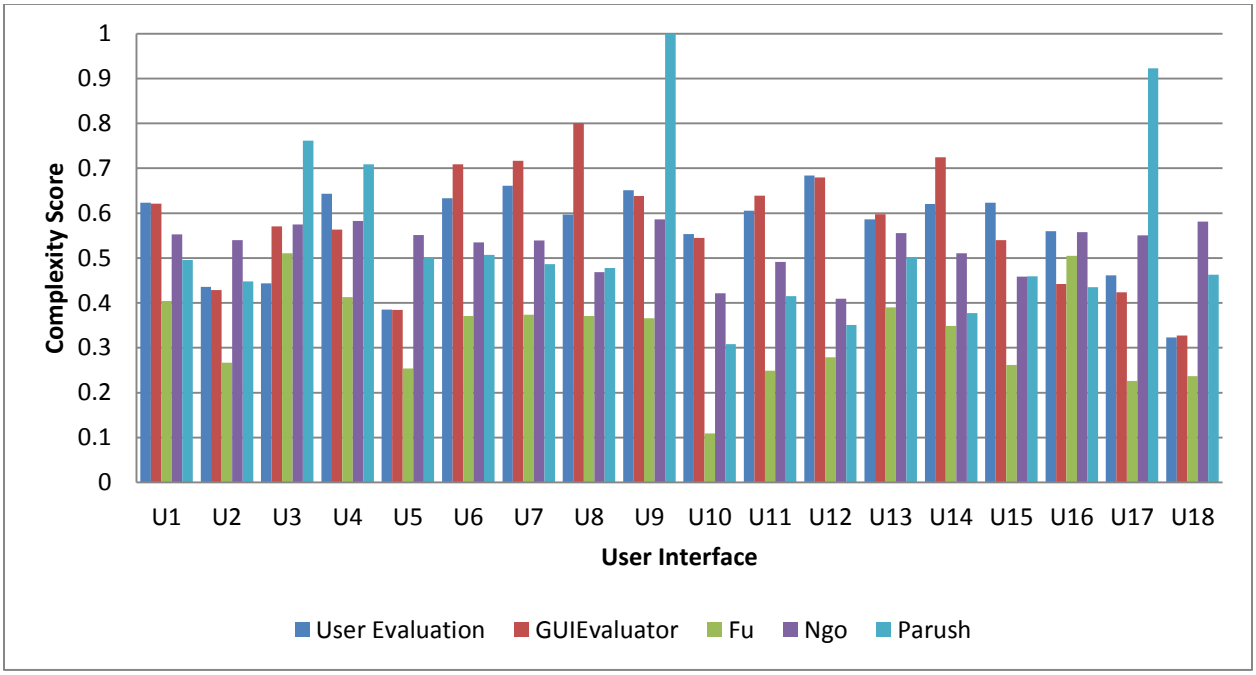

Figure 6.1. Comparison of the complexity scores of user evaluation and metric-models

scores of the participants' evaluations. Therefore, the GUIEvaluator is the most accurate metrics-model to evaluate the complexity of user interfaces.

For further analysis, the Pearson correlation test t-test were performed for the users' evaluation and metrics-models with five complexity measures, at a significance level of 0.05, on 18 GUI layouts. These measures are: Alignment, grouping, size, density, and balance. Table 6.3

shows the Pearson correlation test and t-test results for the alignment that was measured by the metrics-model with the overall user interface complexity scores that were provided by the participants' evaluation.

In terms of alignment complexity, the GUIEvaluator has the highest correlation with overall complexity scores of GUIs with R value (0.672) at significance level ($p=0.0028$). Moreover, the means of the GUIEvaluator, for alignment factor, and overall complexity score have no significant difference at significance level 0.05. Both the Fu and Ngo metrics-models have a moderate negative correlation of the alignment factor with the overall complexity score with R values: -0.6235 and -0.4167, respectively. On the one hand, Fu's model has t-value (11.764) with ($p<0.00001$), which means there is a significant difference between the means of alignment that is measured by Fu's model and overall user interface complexity score. On the other hand, Ngo's model has t-value (1.716) with ($p=0.0952$), which means there is no significant difference between the means of alignment values that were measured by Fu's model and overall user interface complexity score.

The Parush's model has a weak positive correlation, in terms of alignment values, with an overall GUI complexity score with R value (0.114) and significant value ($p=0.6524$). In addition, there is a significant difference between the means of alignment that was measured by Parush's model and overall complexity score of given user interfaces at ($p<0.00001$).

Figure 6.2 compares the overall complexity scores by the participants' evaluation and alignment complexity scores by the metrics-models for 18 user interfaces. There are resemblances among the overall user interface complexity score and the alignment values that were measured by the GUIEvaluator and Ngo models as shown in Fig. 6.2. Furthermore, the alignment values that were measured by Fu and Parush models have a noticeable divergence

Table 6.3. The results of pearson correlation test and t-test of the alignment factor with the overall user interface complexity by the metric-models

Metric-Model	Pearson correlation test		t-test	
	R	p-val.	t-val.	p-val.
GUIEvaluator	0.672	0.0028*	-1.005	0.323
Fu	-0.6235	0.0057*	11.764	<0.00001
Parush	0.114	0.6524	5.623	<0.00001
Ngo	-0.4167	0.086	1.716	0.0952

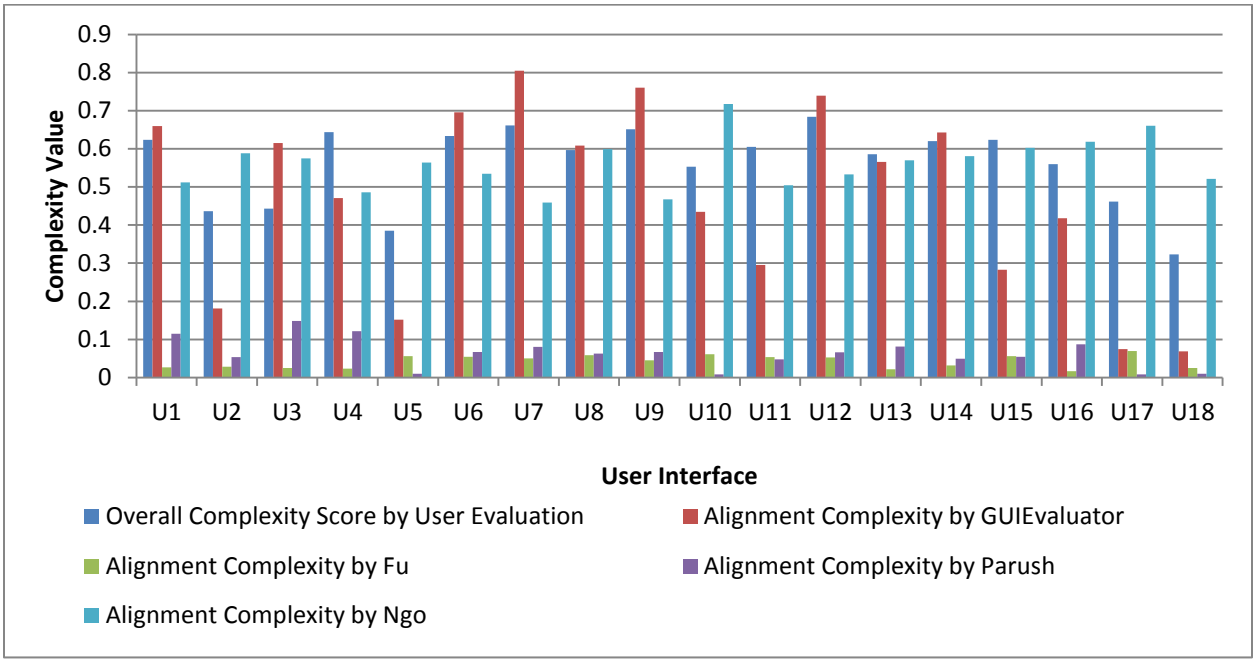

Figure 6.2. Comparison of overall complexity scores by the user evaluation and alignment complexity scores by the metric-models

with the overall complexity scores, where those two metrics-models have small values compared with the complexity scores for the 18 user interfaces. Therefore, these results provide statistical evidence of the effectiveness of alignment factor to measure the complexity of GUIs in GUIEvaluator and Ngo models. To sum up, GUIEvaluator and Ngo models are the effective models for evaluating the complexity of GUIs in terms of alignment.

Table 6.4 shows the Pearson correlation test and t-test results for the grouping factor with the overall user interface complexity by the metrics-models. As shown in Table 6.4, there is a strong positive correlation between the grouping values that were measured by Fu model and overall complexity scores with R value (0.7684) at a significance level of ($p<0.05$). In contrast, the grouping values of GUIEvaluator and Parush's model have a weak positive correlation with overall scores with R values, 0.095 at ($p=0.7080$) and 0.114 at ($p=0.6524$), respectively. Moreover, there is a strong negative correlation between overall complexity scores and the values of grouping factor that were measured by the Ngo model with R value (-0.9566) at significance level at ($p<0.05$).

Table 6.4 shows unexpected results of the t-test, where the means of grouping factor that were measured by metrics-models have significant difference with means of overall complexity score with significant level at ($p<0.05$). These results support the interpretations that have been made to explain unexpected results of the grouping factor in Chapter 5. To sum up, grouping factor may effect negatively on the effectiveness of any of the metrics-models for evaluating the complexity of user interfaces. Therefore, grouping factor still needs further investigation.

Table 6.4. The results of pearson correlation test and t-test of the grouping factor with the overall user interface complexity given by the metric-models

Metrics-Model	**Pearson correlation test**		**t-test**	
	R	**p-val.**	**t-val.**	**p-val.**
GUIEvaluator	0.095	0.7080	-2.657	0.012
Fu	0.7684	0.0002*	2.4514	0.01952
Parush	0.114	0.6524	5.623	<0.00001
Ngo	-0.9566	<0.00001*	4.28	0.0001

Figure 6.3 shows the comparison of overall complexity scores by the participants' evaluation and grouping complexity scores by the metrics-models. Somewhat, there is a convergence among the values of grouping of GUIEvaluator and Fu models and overall complexity scores of the participants' evaluation for 14 user interfaces. In contrast, this convergence is shown for the user interfaces 5, 10, 17, and 18. Figure 6.3 illustrates that there is a significant difference between the grouping values of Parush and Ngo models and overall complexity scores for the 18 user interfaces. Therefore, in terms of grouping, Fu's metrics-model is the most effective model for evaluating the complexity of GUIs.

The widget size factor is widely used in metrics-models, such as the GUIEvaluator, Fu, and Parush, to evaluate the complexity of GUIs. Therefore, this factor has been investigated to illustrate its effectiveness on the overall complexity score of GUIs. Therefore, Table 6.5 shows the Pearson correlation test and t-test results for the size factor with the overall user interface

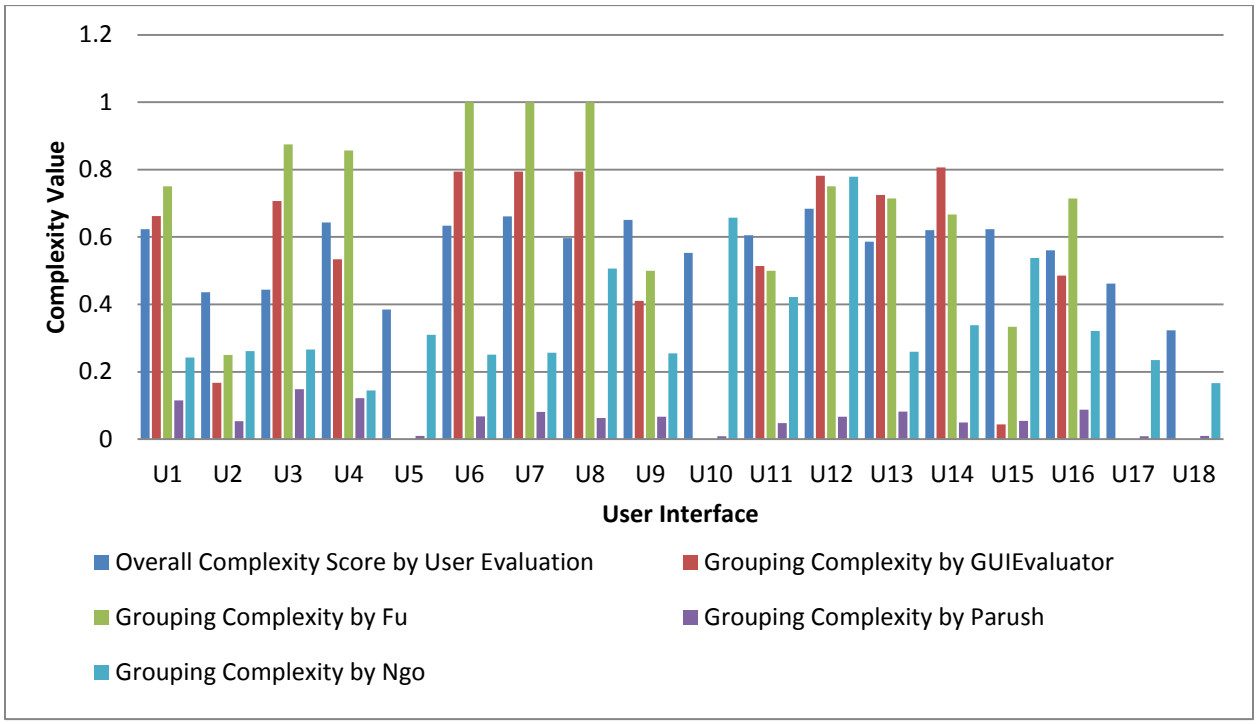

Figure 6.3. Comparison of overall complexity scores by the user evaluation and grouping complexity scores by the metric-models

complexity by the metrics-models. There is a strong positive correlation between the values of the size factor of both the GUIEvaluator and Parush models with R values 0.767 and 0.9949, respectively, and overall user interface complexity at a significance level of ($p<0.001$). In contrast, Fu's model has a weak positive correlation with R value (0.2077) and Ngo's model has not measured the size factor as an individual metric but it is included as a part of other GUI complexity metrics.

In addition, Table 6.5 shows the results of t-test, where the GUIEvaluator and Parush models have no significant difference of the means of size factor and the mean overall complexity scores with ($p=0.319$) and ($p=0.0654$), respectively, at a significance level of 0.05. In comparison with the GUIEvaluator and Parush, the t-test results of Fu's model show that there is a significant difference between the values of size factor and overall complexity of GUIs at significance level of 0.05. Therefore, size factor plays a significant role to evaluate the complexity of GUIs in both the GUIEvaluator and Parush models.

Figure 6.4 presents a comparison of the overall complexity score by user evaluation and the size complexity scores by the metrics-models, where the values of size factor of GUIEvaluator and Parush and overall complexity scores can be shown convergent. In contrast,

Table 6.5. The results of pearson correlation test and t-test of the size factor with the overall user interface complexity by the metric-models

Metrics-Model	**Pearson Correlation Test**		**t-test**	
	R	**p-val.**	**t-val.**	**p-val.**
GUIEvaluator	0.767	<0.001*	-1.013	0.319
Fu	0.2077	0.4082	3.154	0.0034
Parush	0.9949	<0.0001*	1.904	0.0654
Ngo	N/A	N/A	N/A	N/A

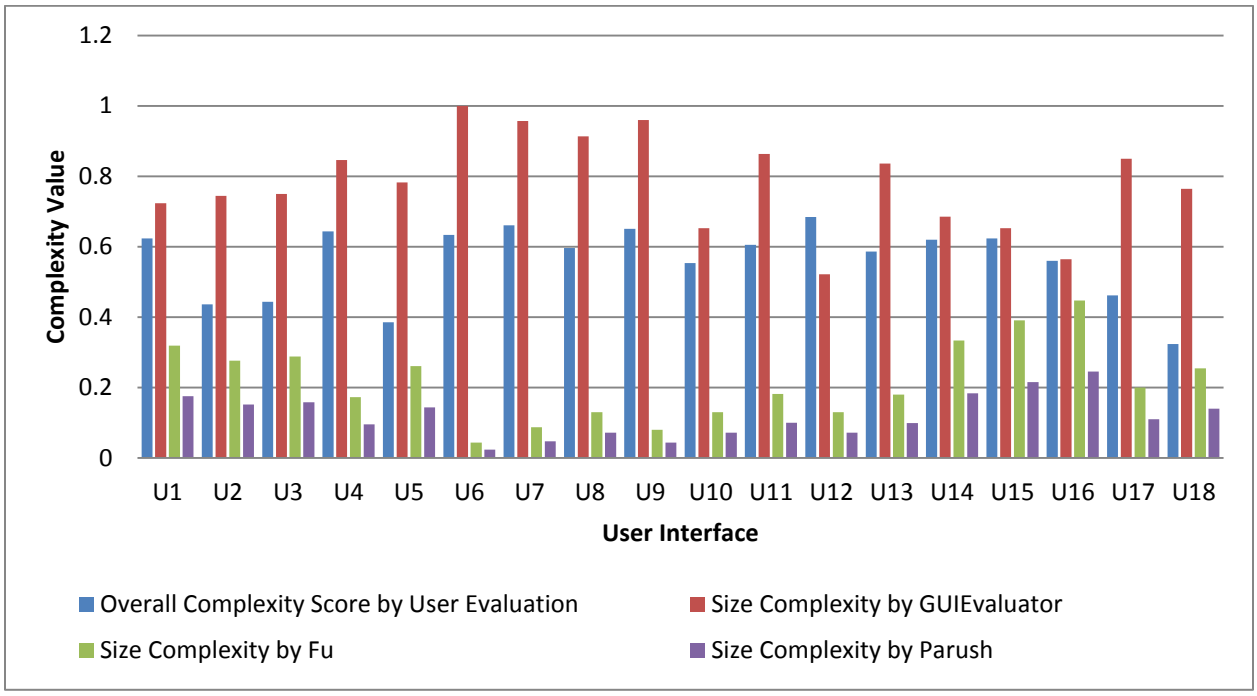

Figure 6.4. Comparison of overall complexity scores by the user evaluation and the size complexity scores by the metric-models

the difference between the size values that were measured by the metrics Fu's model and the values of the overall complexity of GUIs is clear. Therefore, the GUIEvaluator and Parush model are the most effective models to evaluate the complexity of GUIs in terms of size complexity.

Table 6.6 shows the Pearson correlation test and t-test results of the density factor with the overall user interface complexity by the metric-models. As shown in Table 6.6, the density values the measured by the metrics-models (GUIEvaluator, Fu, and Parush) have a moderate positive correlation with the overall complexity scores of GUIs with R values 0.585, 0.5097, and 0.6369, respectively at significance level ($p<0.05$). In contrast, the Ngo's model has a strong negative correlation with the overall complexity scores of GUIs with R value (-0.8522) at significance level ($p<0.05$).

Furthermore, Table 6.5 shows the results of t-test, where the GUIEvaluator model has no significant difference of the means of density factor and the mean overall complexity scores with

Table 6.6. The results of pearson correlation test and t-test of the density factor with the overall user interface complexity by the metric-models

Metrics-Model	Pearson correlation test		t-test	
	R	p-val.	t-val.	p-val.
GUIEvaluator	0.585	0.0098*	-1.050	0.302
Fu	0.5097	0.0307*	3.104	0.0038
Parush	0.6369	0.0045*	2.835	0.0077
Ngo	-0.8522	<0.00001*	7.527	<0.00001

(*p=0.302*) at significance level of 0.05. In comparison with the GUIEvaluator, the t-test results of other metrics-models show that there is a significant difference between the values of density factor and overall complexity of GUIs at significance level of 0.05. Therefore, density factor plays a significant role to evaluate the complexity of GUIs in the GUIEvaluator.

Figure 6.5 compares the overall complexity scores of the participants' evaluation and the density complexity scores by the metric-models. Somewhat, the density values of metric-models (GUIEvaluator, Fu, and Ngo) and overall complexity scores are convergent. In contrast, the density values of Parush's model and overall complexity scores are divergent.

To sum up, the results belonging to the density factor provides statistical evidence of the importance of the density factor to calculate the complexity of GUIs, especially in GUIEvaluator.

The balance factor calculations are provided by GUIEvaluator and Ngo models. The Pearson correlation test and t-test results for the balance factor with the overall user interface complexity by the metrics-models are given in Table 6.7. The results of the Pearson correlation test show a strong positive correlation between the values of the balance factor of GUIEvaluator and the overall complexity scores with R value (0.705) at significance level (*p<0.05*).

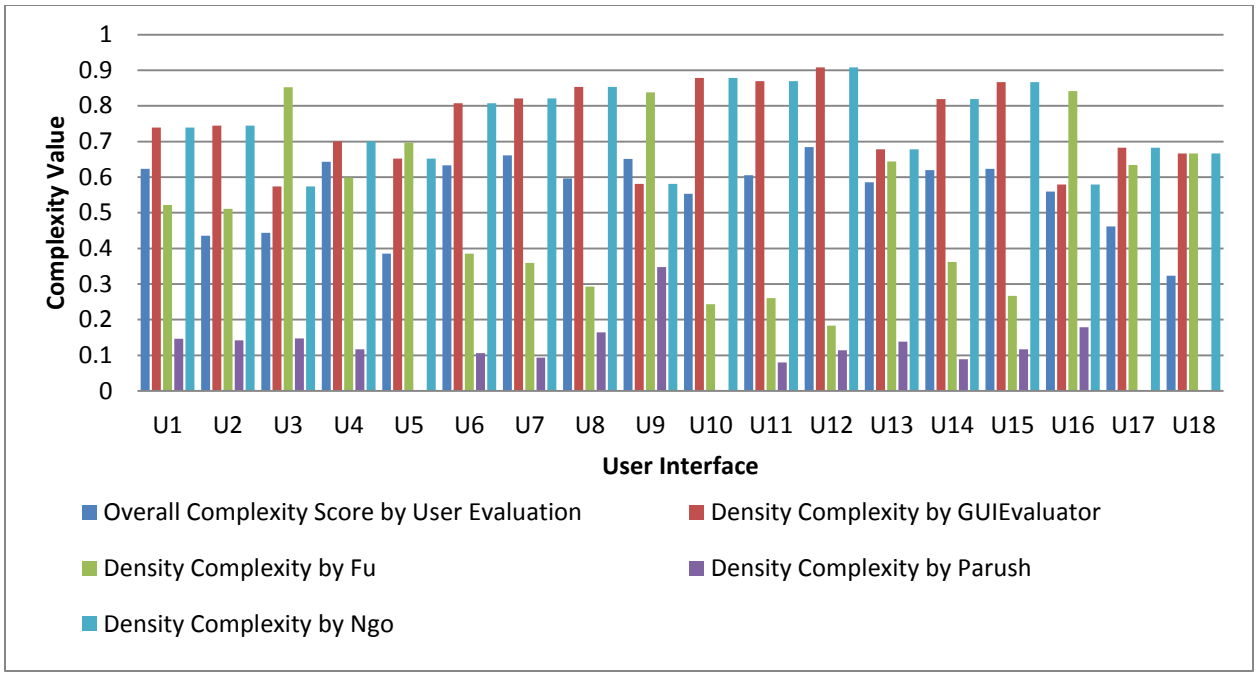

Figure 6.5. Comparison of overall complexity scores by the user evaluation and the density complexity scores by the metric-models

Furthermore, the values of the balance factor of Ngo's model have a weak positive correlation with R value (0.4167) at a significance confidence value (*p=0.0854*). In contrast, the balance factor is not calculated by the Fu and Parush models. At significance level 0.05, the t-test results show that there is no significance difference between the means of the balance factor by GUIEvaluator and overall complexity scores with t-value (-0.857) at significance level (*p=0.398*). In comparison with the GUIEvaluator, the balance factor of Ngo's model has a significant difference with overall complexity scores of GUIs.

Figure 6.6 presents a comparison of overall complexity scores by the participants' evaluation and balance complexity scores by the metrics-models. As shown in Fig. 6.6, the values of balance factor of GUIEvaluator have similarities with overall complexity scores of 18 GUIs. In contrast, the values of balance factor of GUIEvaluator have dissimilarities with overall complexity scores of most of GUIs in this experiment.

Table 6.7. The results of pearson correlation test and t-test of the balance factor with the overall user interface complexity by the metric-models

Metrics-Model	Pearson Correlation Test		t-test	
	R	p-val.	t-val.	p-val.
GUIEvaluator	0.705	0.0011*	-0.857	0.398
Fu	N/A	N/A	N/A	N/A
Parush	N/A	N/A	N/A	N/A
Ngo	0.4167	0.0854	12.873	<0.00001

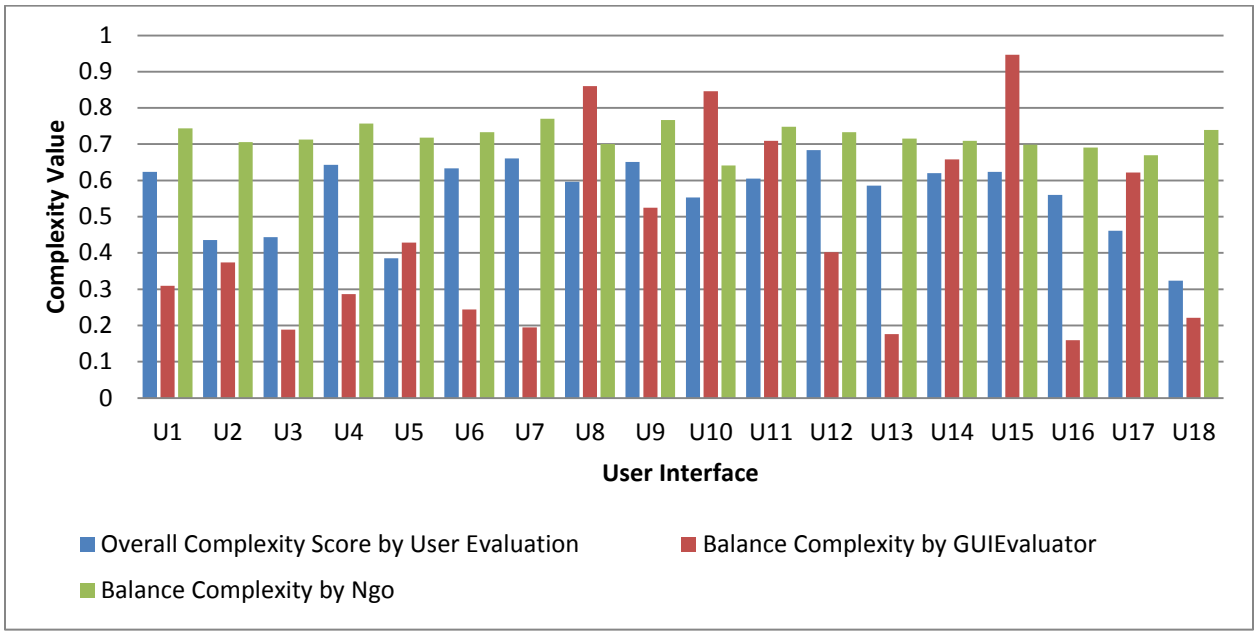

Figure 6.6. Comparison of the overall complexity scores given by user evaluation and the balance complexity scores given by the metric-models

In order to evaluate the complexity of GUIs, the balance factor of GUIEvaluator is more effective than the balance factor of Ngo's model to be used for accurate calculations of GUI complexity.

Table 6.8 presents summary of the relationships: Strong (S), Moderate (M), and Weak (W), which were provided by the Pearson correlation test, for the metrics-models: GUIEvaluator, Fu, Parush, and Ngo. The GUIEvaluator has strong positive correlations among the alignment,

Table 6.8. Pearson correlation level (Strong (S), Moderate (M), Weak (W)) among the five design factors and the user interface complexity scores by the metric-models

Design Factor	Metrics-Model			
	GUIEvaluator	**Fu**	**Parush**	**Ngo**
Alignment	S +	M -	W +	W -
Grouping	W +	S +	W +	S -
Size	S +	W +	S +	N/A
Density	M +	M +	M +	S -
Balance	S +	N/A	N/A	M +

size and balance factors and overall complexity scores of GUIs, a weak positive correlation between the grouping factor and overall complexity scores of GUIs, and a moderate positive correlation between the density factor and overall complexity scores of GUIs.The overall complexity scores of GUIs have a moderate negative correlation, a strong positive correlation, a weak positive correlation, and a moderate positive correlation with the alignment, grouping, size, and density factors in Fu's model, respectively.

Furthermore, the overall complexity scores of GUIs have weak positive correlations with alignment and grouping and strong and moderate positive correlations with size and density factors in Parush's model, respectively. In Ngo's model, the grouping and density factors have a strong negative correlation, the alignment factor has a weak negative correlation, and the balance factor has a strong negative correlation with overall complexity scores of GUIs. The conclusion that can be drawn from the findings in Table 6.8 is the GUIEvaluator has the highest correlation scores between the five design factors and overall complexity scores of GUIs.

6.2.1.2. Discussion

To perform Experiment 1, four metrics-models: GUIEvaluator, Fu, Parush, and Ngo, were studied to determine the most effective model to measure the complexity of user interfaces in comparison with the participants' evaluation. The findings of this experiment outline that the GUI complexity scores of the participants' evaluation and the GUIEvaluator are consistent. In

contrast, the complexity scores of Fu and Parush models and the participants' evaluations are, slightly, inconsistent and the complexity scores of Ngo model and the participants' evaluation are inconsistent. The potential causes behind the inconsistency may be one or more of the following: First, the balance factor is not supported by Fu and Parush models and size factor is not supported by Ngo model. As a result, the other factors will gain more weight and their effect on overall complexity scores will increase. In other words, if one of the other factors has a weak correlation with overall complexity scores of a given user interface, the effect of this factor will cause the overall complexity score will be diminished. Second, in Ngo model, the weights of design factors that have been used are equal. In practice, this is not true because the design factors have different weights according to the user attitudes and preferences. Third, the investigation found that the grouping factor has a weak positive and correlation with complexity scores which were measured by GUEvaluator and Parush and a strong negative correlation with complexity scores which were measured by Ngo compared to the GUI complexity scores that were provided by the participants. Perhaps the reasons behind this are: (1) Misunderstanding the object-grouping on the screens, (2) 46% of participants have no experience in software development, or (3) The grouping metric may be insufficient to measure the widgets grouping.

To summarize the findings of this experiment, whether the GUIEvaluator or user rating is used to evaluate the user interface complexity, we reach the same conclusion. Therefore, the findings confirm the effectiveness of GUIEvaluator and its metrics to measure the complexity of GUIs during the early stages in software development.

6.2.2. Experiment 2: Effectiveness of the Metric-Models for Predicting the Subjective Rating of GUI Usability

6.2.2.1. Results and Analysis

The objective of Experiment 2 is to investigate the effectiveness of the following metric-models: GUIEvaluator, GUILayout++, and BaLOReS to predict user rating of usability of a given user interface. This objective was investigated through two directions: First, examining the effectiveness of these three metrics-models to determine the best GUI layout among alternative GUI designs meets user expectations. Second, examining the effectiveness of these three metrics-models to measure the usability of a given user interface. Therefore, the investigation was performed to address this research question:

RQ6: Can GUIEvaluator and its metrics predict, precisely, the usability of a given user interface more than the existing metric-models?

The hypotheses associated with RQ6 are:

H11: For a given pair of user interfaces, GUIEvaluator and its metrics is the best metrics-model that predicts the subjective rating rather than the other metrics-models.

H12: Given a specific user interface, there is a significant correlation between the GUI usability provided by users and the GUI rating provided by the metric-models.

A controlled experiment has been designed and performed considering the RQ6. To examine the hypothesis (*H11*), the z-test was performed for both the user rating and the rating of GUIEvaluator, GUILayout++, and BaLOReS, at a significance level of 0.05, on nine pairs of user interfaces. The z-test has been performed in this experiment to determine whether the difference between the user rating proportion and metric-models rating proportions is significant or not. Furthermore, if there is no significant difference between the two proportions, the z-test

fails to reject the null hypothesis, which assumes that the means of two sample proportions are equal.

Table 6.9 shows the findings of users' rating and the metrics-models rating among nine pairs of alternative user interfaces. The findings show that the GUIEvaluator rated eight pairs of user interfaces out of nine pairs as the same as users' rating with accuracy (88.8%). In contrast, GUILayout++ and BaLOReS rated six pairs of user interfaces out of nine pairs as the same as users' rating with accuracy (66.7%).

Table 6.10 shows the findings of two sample z-test of user rating proportion and metrics-models proportions at significance level of 0.05. As shown in the Table 6.10, there is no significant difference between the proportions of user rating and GUIEvaluator rating, at a significance level of 0.05, for eight out of nine pairs of user interfaces. Therefore, the null hypothesis can be accepted that the proportions of user rating and GUIEvaluator are equal. Moreover, the difference range between the two proportions of eight pairs of user interfaces is

Table 6.9. The results of users' preferences and the metric-models among nine pairs of alternative user interfaces

Pair of User Interfaces	**Users' Rating**	**GUIEvaluator**	**LU in GUILayout++**	**BaLOReS**
UI1 and UI2	UI2	***UI1****	UI2	UI2
UI3 and UI4	UI3	UI3	UI3	UI3
UI5 and UI6	UI6	UI6	UI6	UI6
UI7 and UI8	UI7	UI7	UI7	UI7
UI9 and UI10	UI9	UI9	***UI10****	***UI10****
UI11 and UI12	UI11	UI11	UI11	***UI12****
UI13 and UI14	UI14	UI14	***UI13****	UI14
UI15 and UI16	UI15	UI15	UI15	UI15
UI17 and UI18	UI17	UI17	***UI18****	***UI18****

0.06 to 0.46. But the findings show that the only one pair of user interfaces has a significance difference between the user rating and GUIEvaluator proportions (*p=0.0032*). The findings provide sufficient evidence that more than 50% of participants have supported the GUIEvaluator rating of given user interfaces. By using the GUIEvaluator, the findings support the acceptance of the hypothesis (*H11).*

Furthermore, Table 6.10 shows, statistically, that there is no significant difference between the proportions of user rating and GUILayout++ rating, at a significance level of 0.05, for six out of nine pairs of user interfaces. Therefore, we can accept the null hypothesis that the user rating and GUILayout++ proportions are equal. Moreover, the difference range between the two proportions of six pairs of user interfaces is 0.04 to 0.46. But the findings show that there are three pairs of user interfaces have significant differences between the user rating and GUILayout++ proportions (*p<0.0001 and p= 0.0385*). The findings provide weak evidence that more than 50% of the participants have supported the GUILayout++ rating of given user interfaces. By using the GUILayout++, the findings support, slightly, the acceptance of the hypothesis (*H11).*

According to the Table 6.10, the findings show, statistically, that there is no significant difference between the proportions of user rating and BaLOReS rating, at a significance level of 0.05, for six out of nine pairs of user interfaces. Therefore, we can accept the null hypothesis that the user rating and BaLOReS proportions are equal. Moreover, the difference range between the two proportions of six pairs of user interfaces is 0.04 to 0.46. But the findings show that there are three pairs of user interfaces have significant differences between the user rating and BaLOReS proportions (*p=0.0175, 0.0385, and p<0.0001*). The findings provide weak evidence that more than 50% of participants have supported the BaLOReS rating of given user interfaces. By using

Table 6.10. The results of two sample z-test of user rating and metric-models proportions at a significance level 5%

Pair of User Interfaces	GUIEvaluator			LU in GUILayout++			BaLOReS		
	Difference	z-value	p-value	Difference	z-value	p-value	Difference	z-value	p-value
UI1 and UI2	0.84	3	***0.0032****	0.16	0.6	0.5386	0.16	0.6	0.5386
UI3 and UI4	0.04	0.3	0.7730	0.04	0.3	0.7730	0.04	0.3	0.7730
UI5 and UI6	0.34	1	0.3148	0.34	1	0.3148	0.34	1	0.3148
UI7 and UI8	0.36	1	0.2940	0.36	1	0.2940	0.36	1	0.2940
UI9 and UI10	0.06	0.4	0.7212	0.94	4.4	***<0.0001****	0.94	4.4	***<0.0001****
UI11 and UI12	0.24	0.8	0.4296	0.24	0.8	0.4296	0.76	2.4	***0.0175****
UI13 and UI14	0.06	0.4	0.7212	0.94	4.4	***<0.0001****	0.06	0.4	0.7212
UI15 and UI16	0.46	1.3	0.1990	0.46	1.3	0.1990	0.46	1.3	0.1990
UI17 and UI18	0.30	0.9	0.3585	0.70	2.1	***0.0385****	0.70	2.1	***0.0385****

the BaLOReS, the findings support, slightly, the acceptance of the hypothesis (*H11).* The findings show that GUIEvaluator has the most effective metrics among the metrics-models for predicting the users' ratings of user interfaces. The Pearson correlation test was performed at significance level of 0.05 to test the hypothesis (*H12*): Given a specific user interface, there is a significant correlation between the GUI usability provided by users and the GUI rating provided by the metrics-models. Table 6.11 shows the findings of Pearson correlation test results of users' satisfaction and the ratings of the metric-models: GUIEvaluator, BaLOReS, and LU in GUILayout++ for 18 user interfaces. As shown in Table 6.11, there is a strong positive correlation between the GUIEvaluator rating and users' satisfaction for 18 user interfaces with R value (0.8069), at a significance level of 0.05. In contrast, there is a moderate positive correlation among the BaLOReS and LU in GUILayout++ ratings and users' satisfaction for 18 user interfaces with R value (0.7072 and 0.6268), respectively, at a significance level of 0.05.

Table 6.11. The results of pearson correlation test of the users' satisfaction and the ratings of the metric-models of 18 user interfaces

Metric-Model	Pearson correlation test		
	R	**p-val.**	**Relationship**
GUIEvaluator Rating	***0.8069***	0.00005*	Strong Positive Correlation
BaLOReS Rating	0.7072	0.00103*	Moderate Positive Correlation
Rating of LU in GUILayout++	0.6268	0.0054*	Moderate Positive Correlation

Therefore, the findings, strongly, support the acceptance of the hypothesis (*H12*) when the GUIEvaluator is used to predict the user satisfaction of a given user interface.

Scatter plots have been used to provide an overview of the collected data and investigate the possible relationships between the users' satisfaction and the metrics-models ratings. Figure 6.7 shows the GUIEvaluator rating and users' satisfaction of 18 user interfaces, which were evaluated by 50 participants. The scatter plot illustrates that there is a strong positive trend between the GUIEvaluator rating and users' satisfaction, as the value of GUIEvaluator rating increases, the users' satisfaction score increases.

Figure 6.8 shows the BaLOReS rating and users' satisfaction of 18 user interfaces, which were evaluated by 50 participants. The scatter plot illustrates that there is a moderate positive trend between the BaLOReS rating and users' satisfaction, as the value of BaLOReS rating increases, the user satisfaction score, slightly, increases.

Figure 6.9 shows the GUILayout++ rating and users' satisfaction of 18 user interfaces, which were evaluated by 50 participants. The scatter plot illustrates that there is a weak positive trend between the GUILayout++ rating and users' satisfaction, as the value of GUILayout++ rating increases, the user satisfaction score, slightly, increases.

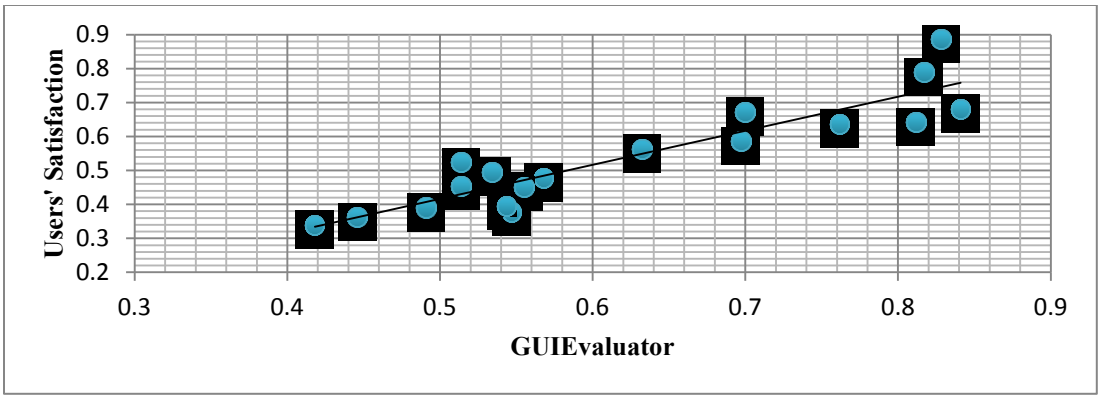

Figure 6.7. Correlation between the values of GUIEvaluator rating and users' satisfaction of 18 user interfaces

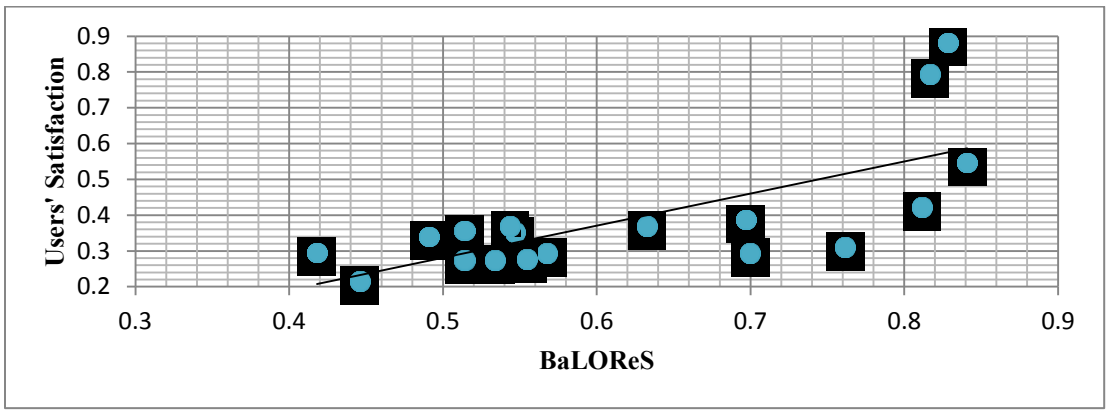

Figure 6.8. Correlation between the values of BaLOReS rating and users' satisfaction of 18 user interfaces

6.2.2.1. Discussion

Experiment 2 focuses on the use of the complexity metrics to evaluate the usability and uniformity of user interfaces. In this experiment, three metrics-models have been used: GUIEvaluator, BaLOReS, and LU in GUILayout++. To determine the optimal user interface layout that meets the user expectation from a given pair of user interfaces, nine pairs of user interfaces were evaluated by both the participants' evaluations and the metrics-models. The findings show that the GUIEvaluator has eight out of nine pairs, which meet the users' preferences with accuracy 88.8%, BaLOReS determined six out of nine pairs, which meet the users' preferences with accuracy 66.7%**,** and LU in GUILayout++ determined six out of nine

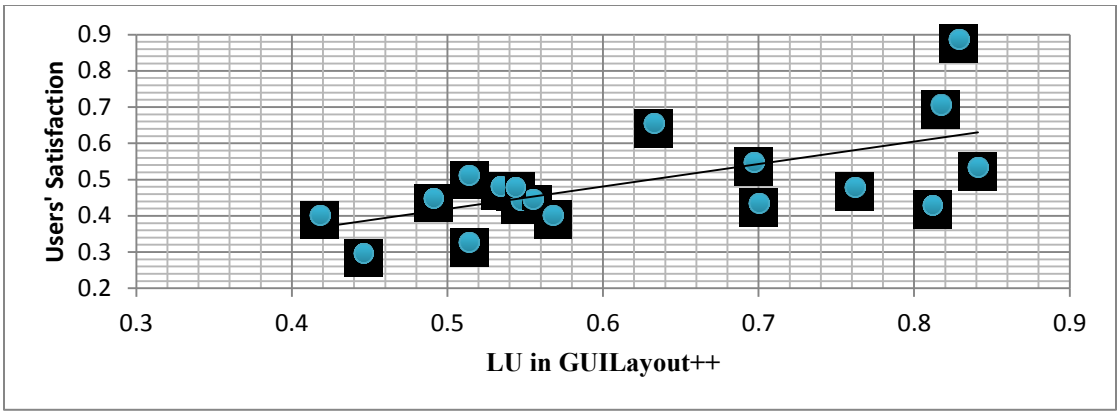

Figure 6.9. Correlation between the values of rating of LU in GUILayout++ and users' satisfaction of 18 user interfaces

pairs, which meet the users' preferences with accuracy 66.7%. The reasons that may cause the significant difference between the GUILayout++ and participants' evaluations for some user interfaces are: First, the user interfaces 11, 12, 17, and 18 were crowded by widgets, which means the GUILayout++ is not sufficient to evaluate the crowded user interfaces. Second, the number of groups in the user interfaces 9 and 10 are: Four and two, respectively. This means that the participants may not take into account the grouping factor when they evaluated this pair of user interfaces, where both the user interfaces have the same layout even the number of groups were not equal.

Moreover, there is a significant difference between the BaLOReS and the participants' evaluations for some user interfaces. The reasons that may cause this difference are: First, the user interfaces 17 and 18 were crowded by widgets, which means the BaLOReS is not sufficient to evaluate the crowded user interfaces. Second, the number of groups in the user interfaces 13 and 14 are: Zero and 2. This means that the participants may not take into account the grouping factor when they evaluated this pair of user interfaces, where both the user interfaces have the same layout even the number of groups were not equal. Furthermore, BaLOReS does not support grouping factor as individual metric. Third, the horizontal and vertical alignments play

significant role to calculate the metrics that are provided by BaLOReS may cause the significant differences among the three pairs of user interfaces that did not meet the users' preferences.

The findings of this experiment show that the GUIEvaluator has a strong positive correlation with the participants' evaluations of the usability of 18 user interfaces with R value (0.8069). . In contrast, there is a moderate positive correlation among BaLOReS and LU in GUILayout++ ratings and the participants' satisfaction for 18 user interfaces with R value (0.7072 and 0.6268), respectively, at a significance level of 0.05. This variance in the correlations may refer to the number of design factors that are supported by each metrics-model. The GUIEvaluator has five design factors: Alignment, grouping, size, density, and balance. The GUILayout++ provides metrics to measure alignment and size factors. The BaLOReS provides metrics that are, mainly, constructed based on alignment and balance factors. As a result, the number of structural design factors that are supported by the metric-models may be considered as a shortcoming of these models.

Therefore, taking all the aforementioned findings into account, GUIEvaluator is the most accurate metrics-model to evaluate the uniformity and usability of a given user interface. These findings provide strong evidence regarding the usefulness and effectiveness of GUIEvaluator in the user interface evaluation.

6.3. Threats to Validity

In general, any user study is subject to threats to validity. Therefore, these threats must be taken into account in order to estimate their impact on the findings of these user studies. In addition to the threats to validity that have been illustrated in Chapter 5, more threats will be discussed in this section.

6.3.1. Internal Validity

The possible internal threat is that of the potential faults in metrics calculation. This threat may have an affect on the conclusions that are made by the metrics models regarding the evaluation of user interfaces. To avoid this threat, the metrics calculation has been conducted several times in different times, then comparing the final calculation results. This comparison provides additional evidence about the accuracy of the metrics calculations by the metrics-models. Another possible internal threat is the number of metrics that are calculated by each metrics-model. Some of these models support only three or four structural metrics, while the GUIEvaluator provides a calculation for five structural metrics. This may effect on the generalization of the findings of the conducted experiments. Therefore, a comparison analysis was conducted for each individual structural factor and the overall GUI complexity and usability values.

6.3.2. External Validity

The possible external threat to validity is the metric-models representativeness issue that may affect the generalization of the findings. To avoid this threat, five metric-models have been investigated in these experiments compared to the GUIEvaluator. To my knowledge, these models represent the most existing models that measure the complexity and usability of GUIs since 1990s until now. Moreover, the excluded models are the models do not fit to the problem discussed in this dissertation and there is a lack information about how to calculate their metrics. Also, knowing that many attempts have been made to contact the authors of the research papers that describe these models but without response.

CHAPTER 7. SCREEN LAYOUT COHESION (SLC) METRIC

7.1. Introduction

Currently, the commonly accepted methods for evaluating user interfaces are: Prototyping, inspection techniques, and usability testing. Each practice has strengths and weaknesses, but the common concerns regarding of the effectiveness of these evaluation methods are going over budget and time [99] [100]. The most important reason that causes these issues are involving the subjective evaluation in these practices. As a matter of fact, subjective evaluation has been used widely to help GUI designers evaluate GUI layout alternatives during the early phase of GUI design. To assist GUI designers with evaluating, effectively, GUI layout designs during an evaluation with the least cost and time possible, a number of tools have been used to help GUI designers in designing productive, effective, and satisfying user interfaces. However, users continue to complain that user interfaces are complicated, not effective and satisfactory, and reduce their productivity [98].

To solve this problem, a different approach may be required to evaluate the quality of GUI layouts. This approach should evaluate different aspects of a given user interface. Therefore, this research focuses on applying the visual cohesion as metric to predict the usability of user interfaces. Cohesion is a measure of the degree of interrelatedness of software component parts [77]. The concept of cohesion is widely used in software engineering for a variety purposes. Cohesion can simplify the software structure and reduce dependencies between component parts. In the user interface design, cohesion can be applied to show how the widgets are related to each other for a given user interface. The visual cohesion metric can be considered as a hybrid metric if it combines some or all of the semantic, structural, and aesthetic aspects of a given user interface. Using hybrid metrics may be more effective to evaluate user interfaces.

Moreover, hybrid metrics are sufficient to provide good prediction rates [78]. Li and Cheung [79] conducted an empirical study, which provided evidence of effectiveness of hybrid metrics to assess software complexity. Hybrid metrics endeavor to remedy the shortcomings of using a single-factor metric [80]. Therefore, there is a need to have hybrid metrics, which measure a combination of structural, aesthetic, and semantic aspects of a given user interface. Furthermore, these metrics can be powerful tools to evaluate the quality of user interfaces.

To take advantage of hybrid metrics, the SLC metric has been proposed to evaluate the quality of GUIs, especially the GUI usability. The SLC is expected to be effective to evaluate user interfaces during the early software design phase. Therefore, this research attempts to resolve the following research questions:

RQ7: For a given pair of user interfaces for the same purpose, is the SLC effective to determine the better user interface design?

RQ8: Is the SLC effective to evaluate the usability of a given user interface?

7.2. Screen Layout Cohesion (SLC) Metric

This section presents the Screen Layout Cohesion (SLC) metric, which is a unified hybrid metric. This metric is computed taking into account aesthetic, structural, and semantic aspects of a given user interface. The main objective of SLC is to assist GUI designers evaluate their GUI layouts in early development phases. SLC extracts the attributes of widgets in order to define the relationships among widgets. The SLC is computed based on two main measures. First, aesthetic and structural relatedness, which includes 17 aesthetic and structural attributes (height, width, left margin, right margin, bottom margin, top margin, backcolor, font size, font style, font type, border style, forecolor, x1_location, x2_location, y1_location, y2_location, and text alignment). These attributes are defined in Visual Basic 2012 to characterize the widgets in the interface

layout. The SLC calculation is not restricted to these attributes, it is possible to integrate other attributes. Therefore, the number of relationships between two widgets on the screen is determined based on the number of attributes associated with both of them.

Second, semantic relatedness, which is computed based on semantic similarities among a set of widgets that exist in one group on the screen layout.

The SLC computes the cohesion based on the values of the following metrics:

1 - Group Layout Cohesion (GLC) metric

According to the Eq.7.2, the relationships (R_{ik}) between widgets (w_i and w_k) are computed based on the 17 attributes that are associated with all pairings of widgets. So, R_{ik} represents the similarity relatedness between the widgets in terms of structural and aesthetic attributes. Thus, when the values of R_{ik} are computed, the GLC metric can be calculated, where the value of GLC is the cumulative sum of the weighting cohesion values for all groups on a given user interface as shown in Eq.7.1. The ratio of the number of widgets in a specific group (G_t) to the total number of grouped widgets on the screen is considered as the weight of each group (Gw_t).

$$\textbf{\textit{Group Layout Cohesion (GLC)}} = \sum_{t=1}^{Groups} \left(\frac{\sum_{i=1}^{w} \sum_{k=1}^{w-i} \text{Rik}}{(w*(w-1))/2} \right) * Gw_t \qquad \text{(Eq. 7.1)}$$

w: the number of widgets, **Rik**: the relationship between widgets on a screen.

$$Rik = \frac{Rh+Rw+RI+Rr+Rb+Rt+Rft+Rc+Rfs+Rf+Rbr+Rfc+Rx1+Rx2+Ry1+Ry2+Ral}{Number\ of\ Relationships} \qquad \text{(Eq. 7.2)}$$

h: height, **w**: width, **l**: left margin, **r**: right margin, **b**: bottom margin, **t**: top margin, **c**: backcolor, **fs**: font size, **f**: font style, **ft**: font name, **br**: border style, **fc**: forecolor, **x1**: x1_location, **x2**: x2_location, **y1**: y1_location, **y2**: y2_location. **al**: text alignment.

2 - UnGroup Layout Cohesion (UGLC) metric

UGLC metric uses the value of R_{ik} that is computed in Eq.7.2, but R_{ik} is the similarity relatedness of ungrouped widgets. The value of UGLC is the cumulative sum of the R_{ik} of ungrouped widgets on a given user interface as shown in Eq.7.3.

$$\textbf{\textit{UGroup Layout Cohesion (UGLC)}} = \frac{\sum_{i=1}^{w}\sum_{k=1}^{w-i} \boldsymbol{Rik}}{\boldsymbol{w*(w-1)/2}} \qquad \text{(Eq. 7.3)}$$

3 - Widget Layout Cohesion (WLC) metric

In computing the WLC metric, the relationships (R_{ik}) between widgets (w_i and w_k) are calculated based on the 17 attributes that are associated with all pairings of widgets of the same type. Eq. 7.2 computes the similarity relatedness (R_{ik}) between the widgets of the same type in terms of structural and aesthetic attributes. Thus, when we get the values of R_{ik}, we can compute the WLC, where the value of WLC is the cumulative sum of the weighting cohesion values for all widget types on a given user interface as shown in Eq.7.4. The ratio of the number of widgets from the same type (T) to the total number of widgets on the screen is considered as the weight of each widget type (Tw_t).

$$\textbf{\textit{Widget Layout Cohesion (WLC)}} = \sum_{t=1}^{Types}\left(\frac{\sum_{i=1}^{w}\sum_{k=1}^{w-i}\mathrm{Rik}}{(w*(w-1))/2}\right)* Tw_t \qquad \text{(Eq. 7.4)}$$

4 - Semantic Relatedness (SR) metric

In Eq. 7.5, the S_{ik} can be calculated, which is the similarity relatedness between all pairings of widgets (w_i and w_k) that exist in a group. The value of SR metric is the cumulative sum of the values of weighting semantic relatedness for all groups on a given user interface as shown in Eq.7.5. The ratio of the number of widgets in a specific group (G_t) to the total number of grouped widgets on the screen is considered as the weight of each group ($\boldsymbol{Sw_t}$).

$$\textbf{\textit{Semantic Relatedness (SR)}} = \sum_{t=1}^{Groups}\left(\frac{\sum_{i=1}^{w}\sum_{k=1}^{w-i}\mathrm{Sik}}{(w*(w-1))/2}\right)* Sw_t \qquad \text{(Eq. 7.5)}$$

In Eq. 7.6, the values of GLC, UGLC, WLC, and SR are used to compute the Screen Layout Cohesion metric (SLC). The values of GLC and UGLC are weighted based on the number of grouped and ungrouped widgets to the total number of widgets on a given user interface (W1, W2), respectively. In addition, SR is weighted by W1 because SR represents the semantic relatedness of widget groups on a given user interface.

$$\textbf{\textit{Screen Layout Cohesion (SLC)}} = \frac{GLC*W1+UGLC*W2+WLC+SR*W1}{2+W1} \quad \text{(Eq. 7.6)}$$

A complementary tool, called GUIExaminer, has been developed to provide the SLC metric calculation for a given user interface.

7.3. The GUIExaminer Tool

GUIExaminer is a tool that provides, automatically, the GUI designers a visual cohesion score based on the aesthetic, structural, and semantic aspects of GUIs to measure the quality of GUIs. The architecture of GUIExaminer is data-centric. Figure 7.1 shows GUIExaminer's architecture which consists of the following components: UI Data Extractor, Semantic Relatedness Extractor, MetricsCalculator, Visual Cohesion Analyzer, GUI, and GUIExaminerDB. The Data Extractor plays an intrinsic role in GUIExaminer's functionality. It is used to read a user interface (.vb) developed in Visual Basic and extract the structural and aesthetic attributes of each widget on the user interface using reflection techniques which are provided by Visual Basic 2012. The MetricCalculator is used to provide metrics calculation, which makes the metrics' values available for Visual Cohesion Analyzer. The metrics provided by the MetricCalculator are: Group Layout Cohesion metric (GLC), UnGroup Layout Cohesion metric (UGLC), Widget Layout Cohesion metric (WLC), and Screen Layout Cohesion metric (SLC). The Visual Cohesion Analyzer is implemented to analyze the results of metrics calculation in order to provide a final decision about a given user interface design. Semantic

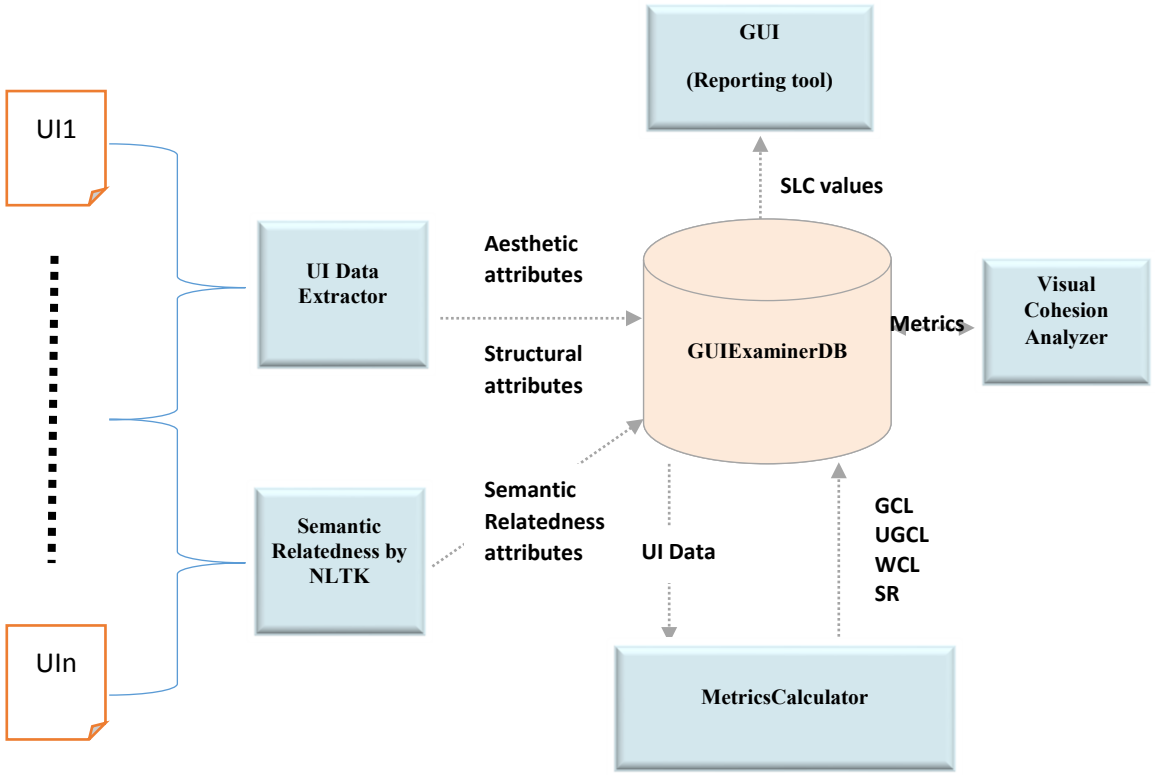

Figure 7.1. The architecture of GUIExaminer

Relatedness Extractor computes the similarity relatedness of the widgets which exist in the same group through using the Natural Language Toolkit (NLTK). All the GUIExaminer's functionalities are presented through a simple attractive GUI. Finally, the GUIExaminerDB is utilized to store the collected data and the calculated values of metrics. With these components, GUIExaminer can be utilized to calculate the SLC, which is used to evaluate the usability of user interface layouts whether the user interface is under development or a part of running applications.

7.4. An Illustrated Example of SLC Calculation

To illustrate how to calculate the Screen Layout Cohesion (SLC) metric, given two real user interface layouts for the same purpose are used: The interface of billing information of Ptel as shown in Fig 7.3 and the interface of billing information by ecwid as shown in Fig 7.4. The

SLC calculations have been performed for those two user interfaces using the GUIExaminer. The main objectives of this example are to illustrate how to compute the SLC for real user interfaces, and confirm the validity of this metric to predict the relative subjective rating for a given set of alternative user interfaces.

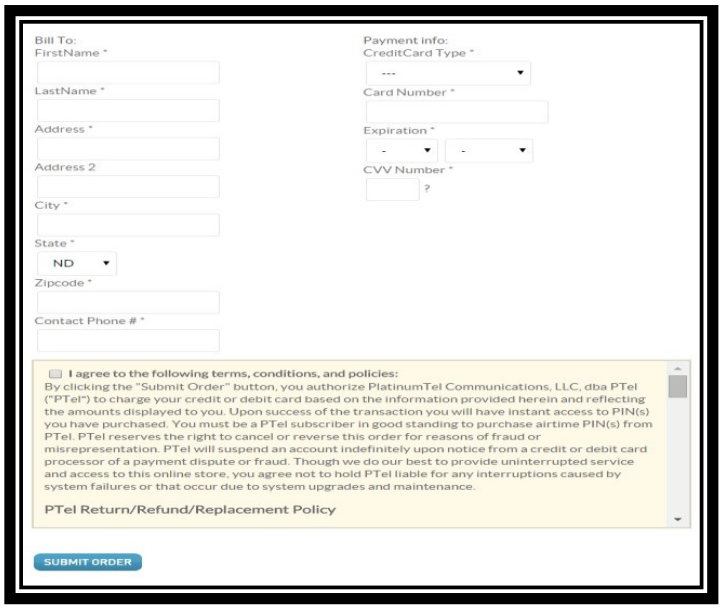

Figure 7.2. The user interface layout of billing information of www.ptel.com

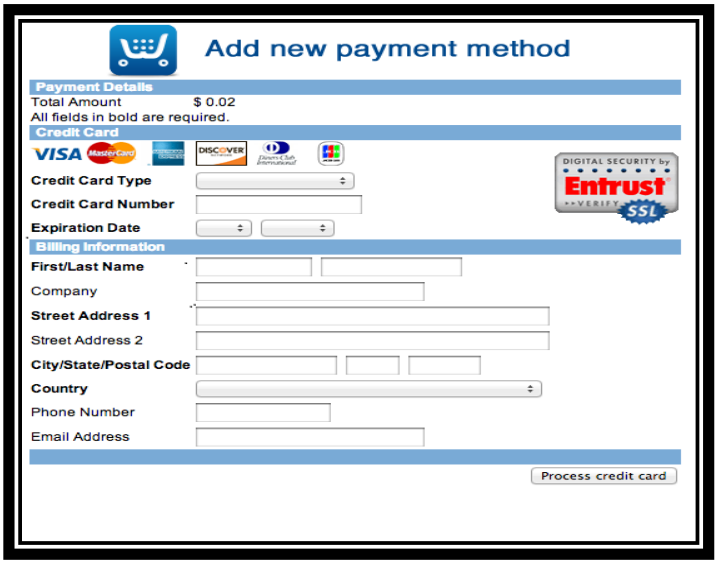

Figure 7.3. The user interface layout of billing information of www.ecwid.com

Table 7.1 shows the values of properties of the two user interfaces of billing information for both Ptel and ecwid. These values are in pixels. These properties that are required for SLC calculation are: The number of grouped and ungrouped widgets, total number of widgets, width and height to calculate the area of user interface, and the number of groups. For example, the number of grouped widgets is used for calculating the visual cohesion of groups taking into account the semantic, structural, and aesthetic relatedness between the grouped widgets on the interface.

To compute Group Layout Cohesion (GLC) and UnGroup Layout Cohesion (UGLC) metrics, the possible common properties among the widgets are identified for a given user interface layout. In this example, 17 properties are determined among the widgets (**h**: height, **w**: width, **l**: left margin, **r**: right margin, **b**: bottom margin, **t**: top margin, **c**: backcolor, **fs**: font size, **f**: font style, **fn**: font name, **br**: border style, **fc**: forecolor, **x1**: x1_location, **x2**: x2_location, **y1**: y1_location, **y2**: y2_location. **al**: text alignment). The values of these properties have been used to calculate the relationship among the widgets on the screen (**Rik**). There are two widget groups on the billing information interface by Ptel compared to three widget groups on the billing information interface by ecwid.

Table 7.2 shows the final relatedness value of each property for both group and ungroup categories. These values have been used to calculate GLC and UGLC taking into account the weight of each category. The values range is 0 to 1, where 0 means that there is no relatedness among a given set of widgets and 1 means that there is a full relatedness among a given set of widgets. The weights have been determined for GLC as the ratio of grouped widgets to the total number of widgets, and for UGLC as the ratio of ungrouped widgets to the total number of widgets.

Table 7.1. The two user interfaces of billing information and their properties

Property	Billing User Interface	
	By Ptel	By ecwid
Number of Widgets	28	39
Number of Grouped Widgets	26	36
Number of Ungrouped Widgets	2	3
Height	660	580
Width	578	550
Area	381480	319000
Number of Groups	2	3

Table 7.2. Calculations of properties of grouped and ungrouped widgets for the two billing user interfaces

Property	User Interface						
	By Ptel			By ecwid			
Grouping_ID	G1	G2	N	G1	G2	G3	N
WidthCal	0.175	0.000	0.000	0.000	0.047	0.165	0.000
HeightCal	0.467	0.356	0.000	1.000	0.485	0.264	0.000
X1Cal	0.308	0.089	1.000	0.333	0.327	0.099	0.000
X2Cal	0.092	0.067	0.000	0.333	0.029	0.000	0.000
Y1Cal	0.000	0.022	0.000	0.333	0.088	0.286	0.000
Y2Cal	0.000	0.022	0.000	0.333	0.023	0.176	0.000
BackcolorCal	0.467	0.444	0.000	1.000	1.000	1.000	0.333
ForecolorCal	0.467	0.444	0.000	1.000	0.485	0.560	0.333
FontsizeCal	1.000	0.800	1.000	1.000	1.000	1.000	0.333
FontNameCal	1.000	0.800	1.000	1.000	1.000	1.000	0.333
FontStyleCal	1.000	1.000	1.000	1.000	0.649	1.000	0.333
LeftMarginCal	1.000	1.000	1.000	1.000	1.000	1.000	1.000
RightMarginCal	1.000	1.000	1.000	1.000	1.000	1.000	1.000
BottomMarginCal	0.467	0.444	1.000	1.000	1.000	1.000	0.333
TopMarginCal	0.467	0.444	1.000	1.000	1.000	1.000	0.333
TextalignmentCal	0.467	0.444	0.000	1.000	0.485	0.330	0.000
BorderStyleCal	0.467	0.444	0.000	1.000	0.485	0.637	0.333

The third value that we need to calculate the SCL is the relatedness among the widget types. As mentioned previously, 17 common properties have been identified among the existing widgets in Visual Basic. For each widget type, the relatedness has been calculated based on the associated properties for that type as shown in Table 7.3. There are five widget types on the both billing interfaces by Ptel and ecwid. Ptel billing interface has the following widget types: TextBox (8 textboxes), Label (14 labels), ComboBox (4 comboboxes), ListBox (1 lisbox), and Button (1button). Ecwid billing interface has the following widget types: TextBox (11textboxes), Label (15 labels), NumericUpDown (4 numericUpDown widgets), PictureBox (8 pictureboxes), and Button (1 button). With having the values of relatedness of each type, the Widget Layout Cohesion (WLC) metric can be calculated.

The fourth value that we need for SLC calculation is the semantic relatedness. The semantic relatedness has been computed for each given interface. The semantic relatedness of Ptel billing interface is 0.61 with semantic values of the groups (G1 = 0.623 and G2= 0.60) and the semantic relatedness of ecwid billing interface is 0.68 with semantic values of the groups (G1 = 0.60, G2 = 0.56 and G3= 1).

Table 7.4 shows the values of the following metrics for both given user interfaces: GLC, UGLC, WLC, SR, and SLC. The ecwid has higher metric values than Ptel interface in terms of group cohesion, widget type cohesion, semantic relatedness, and overall layout cohesion. According to these results, the billing interface by ecwid with SLC value (0.715) is better than the billing interface by Ptel with SLC value (0.597).

Table 7.3. Calculations of properties of widget types for both billing user interfaces

Property	The Billing User Interface									
	By Ptel					By ecwid				
	ListBox	Combo Box	TextBox	Label	Button	TextBox	Numeric UpDown	Label	Button	Picture Box
WidthCal	0.000	0.000	0.583	0.000	0.000	0.127	0.000	0.010	0.000	0.536
HeightCal	0.000	1.000	1.000	0.846	0.000	1.000	1.000	0.867	0.000	0.536
X1Cal	0.000	0.000	0.333	0.282	0.000	0.509	0.500	0.533	0.000	0.000
X2Cal	0.000	0.000	0.306	0.013	0.000	0.073	0.000	0.019	0.000	0.000
Y1Cal	0.000	0.167	0.000	0.013	0.000	0.073	0.167	0.010	0.000	0.750
Y2Cal	0.000	0.167	0.000	0.013	0.000	0.073	0.167	0.010	0.000	0.536
BackcolorCal	0.000	1.000	1.000	1.000	0.000	1.000	1.000	1.000	0.000	1.000
ForecolorCal	0.000	1.000	1.000	1.000	0.000	1.000	1.000	0.867	0.000	1.000
FontsizeCal	0.000	1.000	1.000	0.846	0.000	1.000	1.000	0.867	0.000	1.000
FontNameCal	0.000	1.000	1.000	0.846	0.000	1.000	1.000	1.000	0.000	0.750
FontStyleCal	0.000	1.000	1.000	1.000	0.000	1.000	1.000	0.467	0.000	0.750
LeftMarginCal	0.000	1.000	1.000	1.000	0.000	1.000	1.000	1.000	0.000	1.000
RightMarginCal	0.000	1.000	1.000	1.000	0.000	1.000	1.000	1.000	0.000	1.000
BottomMarginCal	0.000	1.000	1.000	1.000	0.000	1.000	1.000	0.867	0.000	1.000
TopMarginCal	0.000	1.000	1.000	1.000	0.000	1.000	1.000	0.867	0.000	1.000
TextalignmentCal	0.000	1.000	1.000	1.000	0.000	1.000	1.000	1.000	0.000	1.000
BorderStyleCal	0.000	1.000	1.000	1.000	0.000	1.000	1.000	1.000	0.000	0.750

Table 7.4. Summary of SLC metric calculations of the two billing user interfaces

Metric	Billing User Interface	
	By Ptel	By ecwid
Cohesion of Grouped Widgets (GLC)	0.490	0.666
Weight of Grouped Widgets	0.929	0.923
Cohesion of Ungrouped Widgets (UGLC)	0.471	0.275
Weight of Ungrouped Widgets	0.071	0.077
Cohesion of Widget Types (WLC)	0.440	0.584
Semantic Relatedness (SR)	0.614	0.680
Weight of Widgets that have semantic relatedness	0.929	0.947
Screen Layout Cohesion (SLC)	**0.597**	**0.715**

7.5. Experiments

There is a need to provide an empirical validation of the effectiveness SLC metric to evaluate and predict the usability of user interfaces. Therefore, to validate the effectiveness of SLC provided by GUIExaminer, the two experiments have been performed. First, effectiveness of GUIExaminer for measuring the GUI usability for a given user interface. Second, effectiveness of GUIExaminer for selecting the best GUI design among a set of alternative designs. Those two experiments provide empirical findings of the effectiveness of SLC. Therefore, the basic assumption in these experiments is: High cohesive user interface leads to high usability rating by users and less time required for users to extract information from user interfaces. Several research studies have supported this assumption: Costantine [77] concluded that high layout coherence enhances ease of use and comprehension. Cheng-Mong [87] found that the high visual cohesion assists unskilled GUI designers to construct an improved user interface. Lee [88] found that the high coherence user interface design enhances the quality of user interface in the early software design phase.

7.5.1. Methodology

7.5.1.1. Participants

The study was conducted at North Dakota State University. As shown in Table 7.5, the participants were 96 student volunteers (41 females, 55 males), where 19.8% of participants were graduate students and 80.2% were undergraduate students. The participants who are majoring in computer science or related fields were 88.5%, and 11.5% were from other majors. The data that were collected about the participants show that 15.6% of participants reported having no experience in software development. But 69.8%, 9.4%, and 5.2% of participants reported having one or two years, three to six years, and six years and above, respectively.

Table 7.5. Participant characteristics

Characteristic	Measure	Number of Participants	Percentage of Participants
Gender	Male	55	0.573
	Female	41	0.427
Age	18-25	53	0.552
	26-35	33	0.344
	36-45	10	0.104
Major	Computer Science and Related Fields	85	0.885
	Other	11	0.115
Education Qualifications	Graduate	19	0.198
	Undergraduate	77	0.802
User Experience in software development	No Experience	15	0.156
	1-2 Years	67	0.698
	3-5 Years	9	0.094
	6 Years and Above	5	0.052

7.5.1.2. User Interfaces for Analysis

In the two experiments, 42 user interfaces were used as objects for this study from various sources. All the user interfaces have been developed in Visual Basic 2012. The number of widgets has been used to classify the user interfaces into Class A, Class B, and Class C. Briefly, user interfaces were categorized into three classes in terms of the number of widgets as follows: Class A (number of widgets <= 25), Class B (26 =< number of widgets <=50), and Class C (number of widgets >=51). Table 7.6 shows, for each user interface category, the number and percentage of user interfaces as follows: Class A consists of 16 user interfaces, which represent 38% of object user interfaces. Class B includes 12 user interfaces, which

represents 28.6% of object user interfaces, while Class C consists of 14 user interfaces, which represents 33.4% of object user interfaces.

7.5.1.3.Data Collection

This section presents a brief description of data collected during the study. A survey application has been developed to collect the data. The study took approximately 50 minutes. To exploit the power of empirical evaluation, the two experiments have been performed. First, effectiveness of GUIExaminer for measuring the GUI usability of a given user interface. Second, effectiveness of GUIExaminer for selecting the best GUI design among a set of alternative designs. At the beginning of the user study, the participants were asked to provide background information that includes the information in Table 7.5. The participant had the decision to select, randomly, which experiment to perform first. The two experiments should be performed by each participant.

The Experiment 1 has been run as follows. First, two examples were shown to the participants to explain the widget attributes that are considered in this research. Moreover, the participants have been shown how to rate the user interfaces. Second, the participants selected randomly the interface from a list provided in the survey application. And then, the participants were asked to rate the three design factors (Usefulness, Ease to use, and Attractiveness) and the overall user satisfaction using a 7-point Likert scale. The participants had to evaluate all the user interfaces that were provided in the experiment.

In Experiment 2, there were 11 pairs of user interfaces. Each pair consists of two alternative GUI designs for the same purpose. The participant selected the preferred user interface design from each pair of user interfaces. Participants used a 7-point Likert scale, which is provided by the survey application, to provide their evaluations.

Table 7.6. User interfaces for analysis and associated data

Category	Number of User Interfaces	Percentage of User Interfaces %
Class A <=25	16	38%
Class B 26-50	12	28.6%
Class C >=51	14	33.4%

The t-test, Pearson correlation test, and 2 sample two-tailed proportion test, specifically the z-test are used to statistically analyze the collected data. The statistical results provide evidence of the effectiveness of the metrics-model and its tool. Those two experiments provided a considerable deal of insight into the investigation of the SLC effectiveness. Therefore, to illuminate the strengths of applying the SLC metric in GUI layout evaluation, the following subsections report the results from an evaluation of 30 user interfaces for Experiment 1 and 11 pairs of user interfaces for Experiment 2 using the GUIExaminer.

7.5.2. Experiment 1: Validation of GUIExaminer Tool and SLC Metric

7.5.2.1. Results and Analysis

The main goal of this experiment is to investigate the effectiveness of the SLC metric given by GUIExaminer in evaluation of the user interface usability. Therefore, the investigation addresses this research question:

RQ7: Is the SLC metric effective to evaluate the usability of a given user interface?

The hypotheses associated with RQ7 are:

H13: given a specific user interface, there is no significant difference between the SLC value of GUIExaminer and the user rating.

H14: given a specific user interface, there is a strong positive correlation between the user rating and the SLC value of GUIExaminer.

To test these two hypotheses, the t-test was performed on (*H13*) and the Pearson correlation test on (*H14*) for both the participants' rating and the GUIExaminer, at a significance level of 0.05, on 30 user interface layouts. The collected data have met the assumptions for a t-test. First, the data are continuous which were collected based on the 7-point Likert scale. Second, the participants are randomly sampled and the user interfaces are randomly selected in this experiment. Third, the sample sizes are equal for the two populations (n=30). Fourth, the collected data are normally distributed. If the t-test does not indicate a substantial difference between the participants' ratings and the GUIExaminer rating, we can assume the null hypothesis and conclude that the means of variances of the two populations are equal. With respect to the Pearson correlation test assumptions, the collected data have met these assumptions, where the collected data are quantitative and have linear association with no outliers.

The mean of the SLC values given by GUIExaminer is 0.61 while the means the participants' ratings of the usefulness, easy to use, attractiveness, and overall users' satisfaction for the same set of user interfaces are 0.62, 0.60, 0.63 and 0.63, respectively. To examine the hypothesis (*H13*), the t-test has been performed as indicated in Table 7.7, for a significance level of 0.05. The findings indicate that there is no significant difference between the means of the usefulness of user interfaces by the participants' rating and the SLC values by GUIExaminer, where t-value = 0.199 with p-val=0.843. Furthermore, the t-values between SLC values by GUIExaminer and the easy to use and attractiveness of user interfaces are 0.418 with p-val=0.679 and -1.295 with p-val=0.206, respectively. These findings also lead us to the same conclusion, which is that there is no substantial divergence among the SLC values and the participants' ratings of the easy to use and attractiveness factors. Moreover, the overall users' satisfaction and the SLC values have no significant difference, where the t-value is -1.023 with

p-val=0.315. Therefore, the hypothesis (*H13*) is strongly supported through the findings of t-test between the participants' ratings of the usability factors (usefulness, easy to use, attractiveness, and overall user satisfaction) and SLC values.

To examine the hypothesis (*H14*), the Pearson correlation test has been performed at a significance level of 0.05. Table 7.7 shows strong positive correlations between the participants' rating of the usability factors and the SLC values by GUIExaminer, where the R value between the usefulness and SLC is 0.83, the R value between the easy to use and SLC is 0.82, the R value between the attractiveness and SLC is 0.79, and the R value between the overall users' satisfaction and SLC is 0.897.

Despite the strong correlation between the overall participants' ratings and the SLC values with R-value is 0.897, but the usability factors (usefulness, easy to use, and attractiveness) have less positive correlation with the SLC values with R-values range (0.79 - 0.83). This signifies that the participants may rate the usability factors and overall users' satisfaction individually or the participants may rate the overall user satisfaction if they are satisfied with any two or all of the three usability factors. Despite the existence of these differences, the findings of the Pearson correlation test provide statistical evidence that confirms the acceptance of the hypothesis (*H14*).

Figure 7.5 depicts the relationship between the SLC values by GUIExaminer and the usefulness rating by the participants of 30 user interface layouts. A strong positive correlation has been observed between the SLC values and the usefulness scores, as the SLC values of GUIExaminer increase, the usefulness scores trend to increase. One of the interesting results that can be observed in Fig 7.5 is two of user interfaces have inconsistent values with other user interface values. This may influence, negatively, the correlation between the SLC metric and the

Usefulness factor. Despite the inconsistent values of two user interfaces, but the scatter plot in Fig 7.5 describes a positive trend for 28 out of 30 user interfaces, which is a proof that the GUIExaminer can be effective to accurately predict the usefulness of user interfaces.

The scatter plot in Figure 7.5 describes the relationship between the SLC metric by GUIExaminer and the easy to use rating by the participants of 30 user interface layouts. A strong positive correlation has been observed among the SLC values and easy to use scores. As the SLC

Table 7.7. The results of the pearson correlation test and t-test among the GUI usability factors and the SLC scores for 30 given user interfaces

GUI Usability Factor by the participants' Rating	**SLC by GUIExaminer Ratings**			
	Pearson Correlation Test		**t-test**	
	R	p-val.	t-value	p-val.
Usefulness	0.83	< 0.00001	0.199	0.843
Easy to Use	0.82	< 0.00001	0.418	0.679
Attractiveness	0.79	< 0.00001	-1.295	0.206
Overall Satisfaction	0.897	< 0.00001	-1.023	0.315

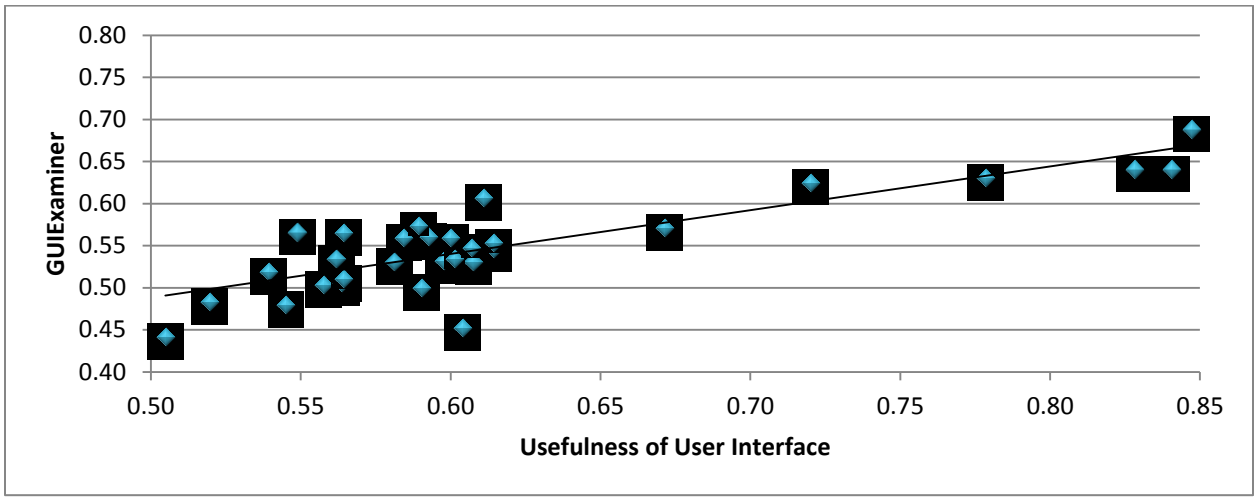

Figure 7.4. Correlation between the values of usefulness factor and GUIExaminer ratings for 30 user interfaces

values of GUIExaminer increase, the easy to use scores trend to increase. The R-val (0.82) between the SLC values and easy to use scores may be affected, negatively, by the inconsistent values for two user interfaces, which are observed in the Fig 7.5. This interesting result is linked to the user interfaces U1 (SLC value = 0.442 and easy to use score = 0.542) and U26 (SLC value = 0.625 and easy to use score = 0.586). However, the scatter plot in Fig 7.5 depicts a positive trend, which is a proof that the GUIExaminer can be utilized to accurately predict the easy to use of user interfaces.

The scatter plot in Figure 7.6 presents the relationship between the SLC values by GUIExaminer and the attractiveness rating by the participants of 30 user interface layouts. A strong positive correlation has been observed among the SLC values and attractiveness scores. As a result, as the SLC values of GUIExaminer increase, the attractiveness scores trend to increase. We can observe three user interfaces have inconsistent values with other the user interfaces. These inconsistent values may influence the correlation between the SLC values and the attractiveness scores for a given set of user interfaces. However, for a given user interface, the GUI attractiveness can be predicted, accurately, the GUIExaminer depending on the positive trend observed in Fig 7.6.

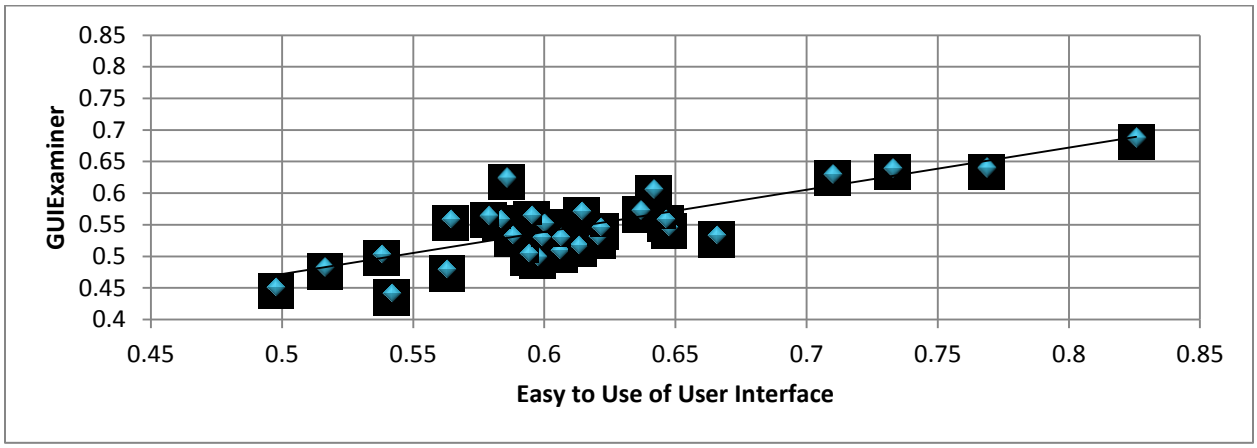

Figure 7.5. Correlation between the values of easy to use factor and GUIExaminer ratings for 30 user interfaces

Figure 7.7 shows a scatter plot that depicts the relationship between the SLC values by GUIExaminer and the participants' satisfaction ratings of 30 user interface layouts. A strong positive correlation has been observed among the SLC values and the participants' satisfaction ratings. The value of R is 0.897, which means as the SLC values of GUIExaminer increase, the participants' satisfaction scores trend to increase. To sum up, the scatter plot in Fig 7.7 describes a positive trend, where data points are tightly clustered around the line, which provides evidence that the GUIExaminer can be used to accurately predict the users' satisfaction of user interfaces.

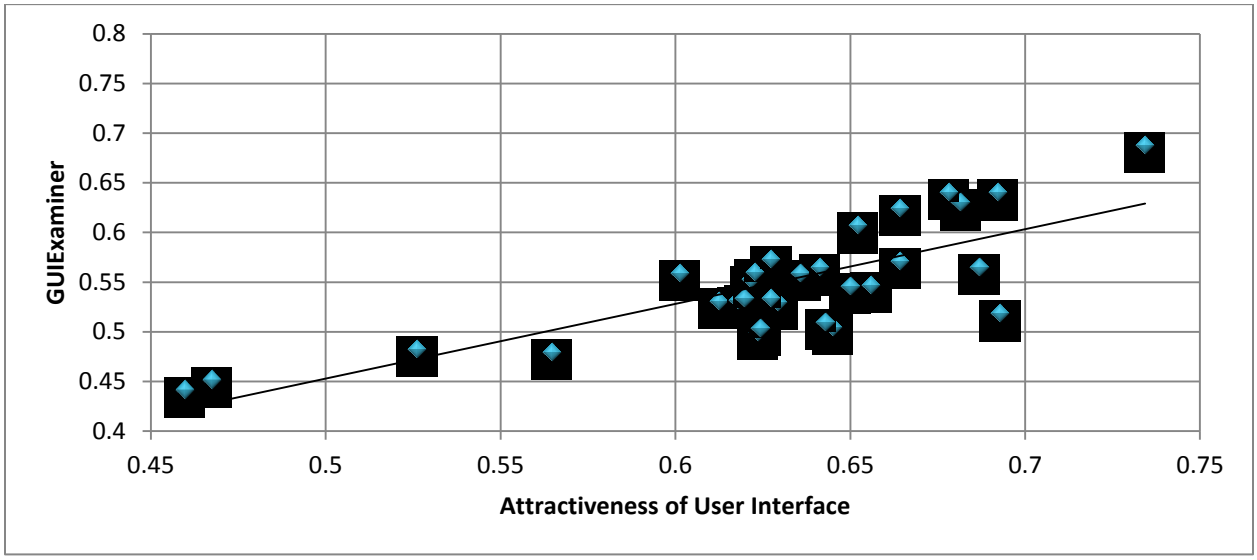

Figure 7.6. Correlation between the values of attractiveness factor and GUIExaminer ratings for 30 user interfaces

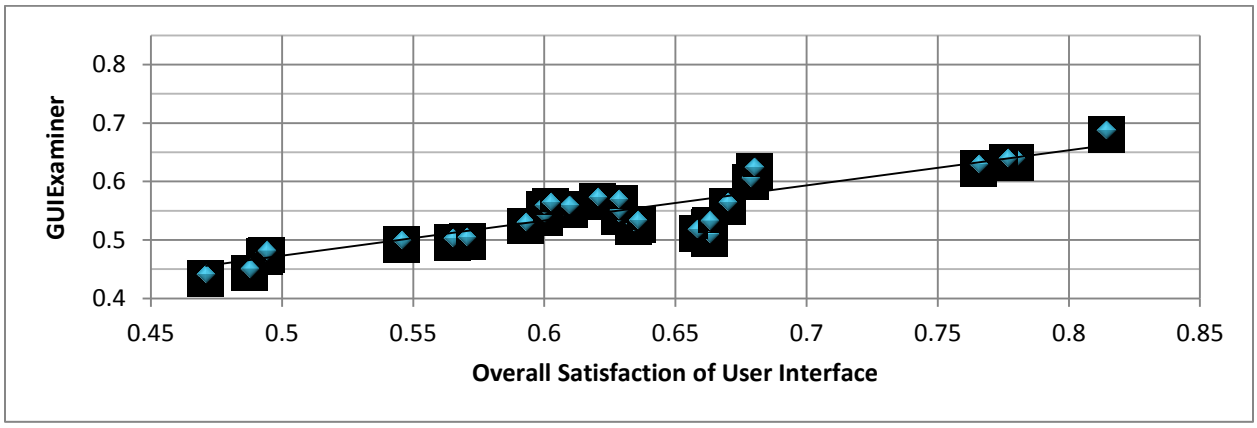

Figure 7.7. Correlation between the values of overall participants' satisfaction and GUIExaminer ratings for 30 user interfaces

7.5.2.2. Discussion

With reference to the findings of this experiment, with additional consideration of its data, to derive practical implications of these findings. The findings of the t-test show that the SLC ratings have no difference with the of the participants' rating of the usefulness, easy to use, attractiveness, and overall satisfaction for 30 user interfaces. Furthermore, the most of the scores of the participants' ratings and the values of SLC are between 50 and 70, but nevertheless, the findings of the Pearson correlation test show that the SLC values by the GUIExaminer have a strong positive correlation with the GUI usability factors as indicated in the Fig.7.4, Fig.7.5, Fig.7.6, and Fig.7.7. These graphs illustrate the shared positive trends among the three usability factors and overall user satisfaction. Through the Experiment 1, we can observe that two or three user interfaces may influence, negatively, the correlation coefficient R among the SLC values and the scores of usefulness, easy to use, and attractiveness of GUIs. Consequently, the scores of these user interfaces could be considered as outliers that need to be removed in order to enhance the performance of SLC to predict effectively the quality of given user interfaces. Regarding the practical implications of these findings, I learned that typically the hybrid metrics are more efficient compared with other metrics. In addition, I learned that various aspects related to the GUI layout design have disparate impacts on the final SLC value for a given user interface. Furthermore, using various aspects could affect the precision of the GUI evaluation process. Thus, adopting various metrics-based tools considering such GUI aspects is potentially a practical approach for GUI designers who have time pressure with GUI release, due to the budget constraint and competitive market. To sum up, this experiment provides results that help us to inform that the SLC metric can predict, accurately, the user preferences with the respect to the usability of GUIs.

7.5.3. Experiment 2: Effectiveness of GUIExaminer for Selecting the Best Alternative GUI Design

7.5.3.1. Results and Analysis

The objective of Experiment 2 is to investigate the effectiveness of the SLC metric to predict the subjective rating of selecting the best GUI layout among alternative GUI designs. Therefore, the investigation was performed to address this research question:

RQ8: For a given pair of user interfaces for the same purpose, is the SLC metric effective to determine the better user interface design?

The hypotheses associated with RQ6 are:

H15: for a given pair of user interfaces, the SLC metric provided by GUIExaminer can predict, precisely, the subjective rating for selecting the better user interface design.

A controlled experiment has been designed and performed to answer the RQ8. To examine the hypothesis (*H2*), the z-test was performed between the participants' rating and the SLC values, at a significance level of 0.05, on 11 pairs of user interfaces. The z-test has been performed in this experiment to determine whether the difference between the proportions of subjective rating and SLC values is significant or not. Therefore, if there is no significant difference between the two proportions, the z-test fails to reject the null hypothesis, which assumes that the means of two sample proportions are equal.

Table 7.8 presents the findings of the participants' ratings of the usefulness factor and the SLC ratings of 11 pairs of alternative user interfaces. The findings indicate that the SLC metric has rated the 11 pairs of user interfaces at the same as the participants' evaluation of the GUI usefulness with accuracy (100%). Furthermore, Table 7.8 presents the findings of two sample z-test of the proportions of the participants' and SLC ratings, at a significance level of 0.05. As

shown in this table, there is no significant difference between the proportions of participants' rating and the SLC values for the 11 pairs of user interfaces. Thus, the null hypothesis can be accepted, which states that the proportions of the participants' ratings of GUI usefulness and SLC values are equal. Moreover, the difference range between the two proportions of 11 pairs of user interfaces is 0.025 to 0.417. To sum up, the findings provide sufficient evidence that more than 50% of participants have supported the SLC to predict the usefulness of a given set of user interfaces. By using the GUIExaminer, the findings confirm the acceptance of the hypothesis (*H15)* in terms of the usefulness of GUIs.

Table 7.8. The results of two sample z-test between the proportions of the participants' ratings of "usefulness" of user interfaces and the SLC values provided by GUIExaminer at a significance level 5%

Pair of User Interfaces	**The Preferred User Interface By**		**Participants' Rating Proportion**	**GUIExaminer Proportion**	**Difference**	**z-value**	**p-val**
	Participants' Rating	**GUIExaminer**					
UI1 and UI2	U2	U2	0.975	1.000	0.025	0.159	0.873
UI3 and UI4	U4	U4	0.786	1.000	0.214	0.521	0.603
UI5 and UI6	U6	U6	0.819	1.000	0.181	0.469	0.638
UI7 and UI8	U8	U8	0.822	1.000	0.178	0.465	0.646
UI9 and UI10	U9	U9	0.726	1.000	0.274	0.613	0.542
UI11 and UI12	U11	U11	0.675	1.000	0.325	0.692	0.490
UI13 and UI14	U14	U14	0.636	1.000	0.364	0.754	0.453
UI15 and UI16	U16	U16	0.788	1.000	0.212	0.518	0.603
UI17 and UI18	U18	U18	0.827	1.000	0.173	0.457	0.646
UI19 and UI20	U20	U20	0.667	1.000	0.333	0.705	0.478
UI21 and UI22	U22	U22	0.583	1.000	0.417	0.842	0.401

Table 7.9 indicates the findings of the participants' rating of the easy to use scores and the SLC values among 11 pairs of alternative user interfaces. The findings show that the SLC has rated nine pairs of interfaces out of 11 pairs of user interfaces at the same as the participants' rating of the easy to use of GUIs with accuracy (82%). Table 7.9 presents the findings of two sample z-test of the proportions of the participants' rating and the values of SLC, at a significance level of 0.05. On the one hand, there is no significant difference between the proportions of the participants' ratings and the SLC values for the nine pairs of user interfaces. On the other hand, Table 7.9 shows that there is a significant difference between the proportions of the participants' rating and the SLC values for the user interfaces (U9, U10, U19, and U20). Therefore, the null hypothesis can be accepted, which states that the proportions of the participants' ratings of easy to use factor and SLC values are equal because of the majority of ratings.

Moreover, the difference range between the participants' and SLC proportions of the 11 pairs of user interfaces is 0.05 to 0.60. To sum up, the findings provide adequate evidence that more than 50% of participants have supported the effectiveness of SLC to predict the easy to use for nine out of 11 pairs of user interfaces. By using the GUIExaminer, the findings in Table 7.9 support the acceptance of the hypothesis (*H15)* in terms of the easy to use of GUIs.

Table 7.9. The results of two sample z-test between the proportions of the participants' ratings of "easy to use" of user interfaces and the SLC values provided by GUIExaminer at a significance level 5%

Pair of User Interfaces	The Preferred User Interface By		Participants' Rating Proportion	GUIExaminer Proportion	Difference	z-value	p-val
	Participants' Rating	GUIExaminer					
UI1 and UI2	U2	U2	0.950	1.000	0.050	0.228	0.818
UI3 and UI4	U4	U4	0.741	1.000	0.259	0.591	0.555
UI5 and UI6	U6	U6	0.796	1.000	0.204	0.506	0.610
UI7 and UI8	U8	U8	0.660	1.000	0.340	0.715	0.472
UI9 and UI10	U10	U9	0.396	1.000	0.604	3.655	<0.001*
UI11 and UI12	U11	U11	0.600	1.000	0.400	0.814	0.418
UI13 and UI14	U14	U14	0.748	1.000	0.252	0.579	0.562
UI15 and UI16	U16	U16	0.688	1.000	0.312	0.672	0.503
UI17 and UI18	U18	U18	0.806	1.000	0.194	0.489	0.624
UI19 and UI20	U19	U20	0.417	1.000	0.583	3.517	<0.001*
UI21 and UI22	U22	U22	0.941	1.000	0.059	0.250	0.803

Table 7.10 presents the findings of the participants' rating of the GUI attractiveness and the SLC values among 11 pairs of alternative user interfaces. The findings indicate that the SLC has rated the 11 pairs of user interfaces at the same as the participants' rating of the GUI attractiveness with accuracy (100%). Furthermore, Table 7.10 presents the findings of two sample z-test of the proportions of the participants' rating and the values of SLC, at a significance level of 0.05. As shown in this table, there is no significant difference between the proportions of participants' rating and the SLC values for the 11 pairs of user interfaces. Therefore, the null hypothesis can be accepted, which states that the proportions of the participants' rating of GUI attractiveness and SLC values are equal. Moreover, the range of differences between the participants' rating and the SLC values of 11 pairs of user interfaces is

0.114 to 0.455. To summarize this, the findings provide substantial evidence that more than 50% of participants have supported the effectiveness of SLC to predict the attractiveness of a given set of user interfaces. Through using the GUIExaminer, the findings in Table 7.10 confirm the acceptance of the hypothesis (*H15)* in terms of the GUI attractiveness.

Table 7.10. The results of two sample z-test between the proportions of the participants' ratings of "attractiveness" of user interfaces and the SLC values provided by GUIExaminer at a significance level 5%

Pairs of User Interfaces	**The Preferred User Interface By**		**Participants' Rating Proportion**	**GUIExaminer Proportion**	**Difference**	**z-value**	**p-val**
	Participants' Rating	**GUIExaminer**					
UI1 and UI2	U2	U2	0.858	1.000	0.142	0.406	0.682
UI3 and UI4	U4	U4	0.806	1.000	0.194	0.490	0.624
UI5 and UI6	U6	U6	0.696	1.000	0.304	0.659	0.509
UI7 and UI8	U8	U8	0.649	1.000	0.351	0.733	0.465
UI9 and UI10	U10	U10	0.721	1.000	0.279	0.622	0.535
UI11 and UI12	U11	U11	0.617	1.000	0.383	0.786	0.430
UI13 and UI14	U14	U14	0.820	1.000	0.180	0.468	0.638
UI15 and UI16	U16	U16	0.719	1.000	0.281	0.624	0.535
UI17 and UI18	U18	U18	0.886	1.000	0.114	0.358	0.719
UI19 and UI20	U20	U20	0.545	1.000	0.455	0.909	0.363
UI21 and UI22	U22	U22	0.714	1.000	0.286	0.631	0.529

Table 7.11 shows the findings of the scores of overall satisfaction provided by the participants and the SLC values among 11 pairs of alternative user interfaces. The findings show that the SLC metric has rated 10 pairs of interfaces out of 11 pairs of user interfaces as the same as the participants' rating of the overall satisfaction of GUIs with accuracy (91%). Table 7.11 presents the findings of two sample z-test of the proportions of the overall participants' satisfaction and the values of SLC metric, at a significance level of 0.05. On the one hand, there is no significant difference between the proportions of the overall satisfaction scores and the SLC values for the 10 pairs of user interfaces. On the other hand, Table 7.11 shows that there is a significant difference between the proportions of the overall satisfaction scores and the SLC values for the pair of user interfaces (U7 and U8). Therefore, the null hypothesis can be accepted, which states that the proportions of both the overall satisfaction scores and the SLC values are equal because of the majority of ratings.

Moreover, the difference range between the two proportions of the 11 pairs of user interfaces is 0.089 to 0.60. To sum up, the findings provide sufficient evidence that more than 50% of participants have supported the effectiveness of SLC metric to predict the participants' satisfaction for 10 out of 11 pairs of user interfaces. By using the GUIExaminer, the findings support the acceptance of the hypothesis (*H15)* in terms of the overall user satisfaction of GUIs.

7.5.3.2. Discussion

The findings of Experiment 2 strongly support the conclusion that states the SLC metric provided by the GUIExaminer can be utilized to determine the best user interface design among a set of alternative designs. The findings of the z-test indicate that the SLC metric can predict the usefulness and attractiveness of a given user interface with accuracy 100% and the easy to use of user interface with accuracy 82% of 11 pairs of user interfaces.

Table 7.11. The results of two sample z-test between the proportions of the participants' ratings of "overall user satisfaction" of user interfaces and the SLC values provided by GUIExaminer at a significance level 5%

Pairs of User Interfaces	The Preferred User Interface By		Participants' Rating Proportion	GUIExa-miner Proportion	Difference	z-value	p-val
	Participants' Rating	GUIExaminer					
UI1 and UI2	U2	U2	0.911	1.000	0.089	0.312	0.757
UI3 and UI4	U4	U4	0.770	1.000	0.230	0.545	0.582
UI5 and UI6	U6	U6	0.736	1.000	0.264	0.597	0.549
UI7 and UI8	U7	U8	0.400	1.000	0.600	3.625	<0.001*
UI9 and UI10	U9	U9	0.610	1.000	0.390	0.797	0.424
UI11 and UI12	U11	U11	0.647	1.000	0.353	0.737	0.459
UI13 and UI14	U14	U14	0.789	1.000	0.211	0.517	0.603
UI15 and UI16	U16	U16	0.732	1.000	0.268	0.603	0.549
UI17 and UI18	U18	U18	0.769	1.000	0.231	0.548	0.582
UI19 and UI20	U20	U20	0.903	1.000	0.097	0.327	0.741
UI21 and UI22	U22	U22	0.654	1.000	0.346	0.726	0.465

Moreover, the SLC can predict the overall users' satisfaction of user interfaces with accuracy 91%. Thus, these findings provide sufficient evidence regarding the usefulness of GUIExaminer in the user interface evaluation. But the collected data about the easy to use factor of the user interfaces (U9, U10, U19, and U20) show a substantial divergence between the subjective evaluation and the GUIExaminer rating. The reasons that may cause of this difference are: The final values of the easy to use factor for the pair of user interfaces (U9 and U10) are 0.605 and 0.612, respectively. Thus, the participants' preferences of these two user interfaces are somewhat equal. Although the difference is very small, but the user interface (U10) is better than the user interface (U9) based on the participants' evaluation. On the other hand, the final values of SLC of U9 and U10 are 0.63 and 0.61. The difference between the two SLC values is caused

by the divergence between the cohesion values of the grouped widgets of U9 and U10, which are 0.54 and 0.49, respectively. The U9 layout consists of two widget groups while the U10 layout consists of three widget groups. Furthermore, the difference between the two SLC values is extremely small, which means the SLC evaluation is commensurate with the participants' ratings. The causes of the difference between the participants' rating of the easy to use factor and SLC rating for the pair of interfaces (U19 and U20) may be related to the difference between the values of coherence of the grouped widgets, where the values are 0.54 and 0.41, respectively. On the other hand, the participants may not take into account the grouping factor when they evaluated this pair of user interfaces. Moreover, the findings indicate a significant difference between the participants' ratings of the overall satisfaction and SLC values for the pair of user interfaces (U7 and U8). This difference may be occurring because the SLC values of U7 and U8 are 0.534 and 0.536, respectively, and the participants may prefer to see several colors on the user interface layout.

In general, the main conclusion of Experiment 2 is that whether the GUIExaminer or the user rating is used to decide the best GUI design among alternative GUI designs during the design phase, the GUI designers will reach the same conclusion. The two experiments proved that higher screen layout cohesion score corresponds with higher usability for a given user interface. Therefore, the findings confirm the effectiveness of SLC given by GUIExaminer to be used during early stages in software development to measure the layout cohesion, which is used to predict the usability of a given user interface.

7.6. Threats to Validity

In general, any user study is subject to threats to validity. Therefore, these threats must be taken into account in order to estimate their impact on the findings of the experiments of SLC

effectiveness evaluation. In addition to the threats to validity that have been discussed in Chapter 5 which may affect on the findings of SLC evaluation experiments, more threats will be discussed in this section. The possible internal and construct threats are described in the following subsections.

7.6.1. Internal Validity

One of the popular internal threats is the potential faults of the software tool that was used for computing the semantic relatedness. This threat may affect the conclusions that are made regarding the effectiveness of the SLC metric and GUIExaminer. To avoid this threat, the results of semantic relatedness were compared to the results of similar purpose tools such the online tool [89].

7.6.2. Construct Validity

The widget attributes which were used in this research are not the only possible attributes. 17 widget attributes have been considered in this research to calculate the cohesion values for the grouped widget, ungrouped widgets, and widget types in addition to the semantic relatedness of the grouped widgets. The computed cohesion values and the semantic relatedness are used to calculate the SLC. To avoid this threat, future studies will be performed using other possible attributes to calculate the SLC.

CHAPTER 8. CONCLUSIONS AND FUTURE WORK

This dissertation presented and evaluated two metric-models for evaluating the complexity and usability of GUIs. These metric-models have been proposed based on the complexity and visual cohesion of GUIs. The first metric-model is developed based on a set of structural factors of user interfaces: Alignment, grouping, size, density, and balance. Each factor is supported by a metric to calculate its complexity degree. The total sum of these five metrics represents the overall score of GUI complexity. These metrics have been used to construct the metrics-model, which is supported by a complementary tool called GUIEvaluator. The GUIEvaluator calculates, automatically, the values of the complexity metrics that are defined in the metrics-model. The findings of the experiments in Chapters 5 and 6 confirm the effectiveness of the metrics-model and GUIEvaluator to measure the complexity of interface layouts and predict the best user interface design among alternative user interface designs. Moreover, this metrics-model has been useful to predict the GUI usability for a given user interface.

In the second model, the SLC metric has been proposed based on the cohesion concept. The SLC metric provides a numeric value that predicts the usability of a given user interface. This metric can be considered as a hybrid metric because it is measured based on the structural, aesthetic, and semantic aspects of GUI layouts taking into account the type of widget and whether the widget is grouped or not. GUIExaminer, a tool, was developed to support the SLC calculation. The findings show that the SLC metric is effective to predict the usability of user interfaces, where the SLC values had a strong positive correlation with the usability factors: Usefulness, easy to use, attractiveness, and overall user satisfaction at significance level 0.05. Therefore, the metrics-model and SLC metric could be used to support designers in making effective design decisions and mitigate the effort, costs, and time of the GUI design.

To sum up, the best resulting model can predict the user evaluation of a given user interface with an accuracy (89%). So, it is fair to claim that the main objective of this work was achieved.

In this dissertation, there are a number of contributions to GUI evaluation research and practice. First, a unified model for measuring the complexity of GUIs has been proposed, allowing the GUI designers to measure the complexity of GUIs in the early stages of software development using the structural measures of GUIs. This model provides complexity values for each structural aspect and overall complexity score for a given user interface. The provided complexity values will help the GUI designers determine which design factor has a high complexity score. This may assist to mitigate the effort and time of the GUI evaluation process later.

Second, two unified metric-models for predicting the usability of GUIs compare to the user evaluations. These models can help to predict these usability measures: Usefulness, easy to use, attractiveness, and the overall user satisfaction of GUIs. According to the literature, those usability factors play a significant role in the GUI evaluation process. The proposed models are sufficient and accurate to predict the usability of desktop and web GUIs. Also, These models can be used and calibrate in software development organizations because these models provide an option to the GUI designers to adjust the weight of design factors to be locally fitted into their needs.

Third, this work evidences the importance of using hybrid metrics to evaluate GUIs. The SLC can be considered as a hybrid metric, which combine the aesthetic, structural, and semantic aspects of a given GUI. As shown above, the accuracy of hybrid metrics to predict the user evaluation of a given GUI is greater than the accuracy of single-aspect metrics.

Finally, two complementary tools were developed to support the metrics calculations. These tools can be used to evaluate the GUIs whether the code exists or not. The findings of this work highlight the importance of developing metric-based tools in order to be used in the early stages of software development. Therefore, there are a number of issues in automating the GUI evaluation still need more investigation.

There are a number of directions for future work that are worth exploring. First, there is a need to investigate the influences of different weighting techniques through controlled experiments. In this dissertation, the number of widgets is the main coefficient that has been used to add weight for the metric values in both metric-models. Therefore, some of other possible factors can be used for the metric weighting such as widget sizes. Another weighting issue is the influence of the complexity score of each design factor in the first metric-model. Absolutely, they do not have the same impact on the final GUI complexity score. In this dissertation, the participants were asked to rate the importance of each design factor during the experiments. These ratings have been used to weight the five design factors. To solve this issue, we can use the optimization techniques to set threshold values of the weights of these factors.

Second, extending the proposed models to measure the complexity and usability of other types of user interfaces, such as mobile interfaces, taking into account the size of screen and new interactive techniques. Therefore, there is a need to take into account the dynamic aspects of GUIs.

Third, the scope of this dissertation was limited to predicting the complexity and usability of GUIs based on structural, aesthetic, and semantic aspects. This limitation was imposed by the objectives of this dissertation. Therefore, future work would be extending the proposed metric-models through introducing new metrics that focus on other aspects of user

interface, such as code-based metrics and task-based metrics. These metrics can be used to predict the complexity of GUIs, GUI testing effort, and GUI evolution. For example, these metrics may help to predict which user interface would be changed or simplified among a set of design alternatives.

Fourth, The proposed metrics can be used as a foundation to propose estimation and prediction models. Therefore, future studies will be performed to investigate some capabilities of the metric-models such as: Estimating the effort and time of the GUI development, predicting the number of test cases that will be required for a given user interface, and estimating the productivity of users using a given user interface. The future studies are not limited to just these points.

Fifth, comparison studies with other GUI evaluation methods such as usability testing, inspection, or prototyping, can be considered as future direction in order to provide evidence of the effectiveness metric-models compared to other evaluation methods.

Finally, future studies will be performed using other possible aesthetic and structural attributes to calculate the SLC metric. In this dissertation, 17 aesthetic and structural attributes have been used to calculate the SLC metric. These attributes are provided by Visual Basic. Therefore, there is a need to investigate the other possible attributes that might be provided by other programming languages. Furthermore, the impact of each attribute on the overall SLC score, may need to be investigated in order to provide more accurate scores that meet the user expectations.

Therefore, this work is the start point of developing a framework that consists of tools and metrics to support the evaluation of user interfaces during the early stages of software development. To sum up, this work addresses the lack of effective tools and techniques to

evaluate the complexity of user interfaces and predict the GUI usability without performing extensive subjective evaluations during the GUI design. So, the automated tools can play a significant role in evaluating the quality of user interfaces.

REFERENCES

[1] T. Comber and J.R. Maltby, 1997, "*Layout complexity: does it measure usability?,*" in S Howard, J Hammond & G Lindgaard (eds), *Human-computer interaction: Interact '97, International Conference on Human-computer Interaction, S*ydney, Australia, 14-18 July, Chapman Hall, London, pp. 623-626.

[2] C. Stickel, M. Ebnar, and A. Holzinger, "*The XAOS metric: understanding visual complexity as measure of usability,*" In Proc. Of USAB'10, the 6th international conference on HCI in work and learning, life and leisure: workgroup human-computer interaction and usability engineering, Springer-Verlag Berlin, Heidelberg, pp.278-290.

[3] A. Sears, "*Layout Appropriateness: A Metric for Evaluating User Interface Widget Layout*," Journal IEEE Transactions on Software Engineering, Volume 19 Issue 7, July 1993 Page 707-719 IEEE Press Piscataway, NJ, USA.

[4] X. Zheng, I. Chakraborty, J. Lin, and R. Rauschenberger, "*Developing Quantitative Metrics to Predict Users' Perceptions of Interface Design,*" Proceedings of the Human Factors and Ergonomics Society Annual Meeting 2008 52: 2023.

[5] H. G. Kang and P. H. Seong, "*An Information Theory-Based Approach for Quantitative Evaluation of User Interface Complexity,*" IEEE TRANSACTIONS ON NUCLEAR SCIENCE, VOL. 45, NO. 6, DECEMBER 1998, 3165 – 3174.

[6] A. Sears, "*AIDE: A tool to assist in the design and evaluation of user interfaces,*" 2001.

[7] J. Xing, (2004). "*Measure of information complexity and the implications for automation design,*" Washington, DC : U.S. DOT FAA, Tech. Report No: DOT/FAA/AM-04/17.

[8] T. Miyoshi and A. Murata, "*A method to evaluate properness of GUI design based on complexity indexes of size, local density, aliment, and grouping,*" Systems, Man, and Cybernetics, 2001 IEEE International Conference on , vol.1, no., pp.221-226 vol.1, 2001

[9] F. Fongling, C. Shao-Yuan, and H. S. Chiu, 2007. "*Measuring the screen complexity of web pages,*" In Proceedings of the 2007 conference on Human interface: Part II, Michael J. Smith and Gavriel Salvendy (Eds.). Springer-Verlag, Berlin, Heidelberg, 720-729.

[10] A. Parush, R. Nadir, and A. Shtub, "*Evaluating the Layout of Graphical User Interface Screens: Validation of a Numerical Computerized Mode,*" International Journal of Human-Computer Interaction 10(4), 343-360 (1998).

[11] D.C.L Ngo, L.S. Teo, and J.G. Byrne, "*Formalizing Guidelines for the Design of Screen Layouts,*" Display 21, 3–15 (2000)

[12] D.C.L Ngo, L.S. Teo, and J.G. Byrne, "*Modeling Interface Aesthetics,*" Information Science 152, 25–46 (2003)

[13] S. González, F. Montero, and P. González, "*BaLOReS: a suite of principles and metrics for graphical user interface evaluation,*" in Proceedings of INTERACCION '12, 13th International Conference on Interacción Persona-Ordenador, 2012.

[14] L.L. Constantine, (1996) "*Usage-Centered Software Engineering: New Models, Methods, and Metrics,*" M. (ed.) Software Engineering: Education & Practice. Los Alamitos, CA: IEEE Computer Society Press.

[15] J. Vanderdonckt, X. Gillo, *Visual techniques for traditional and multimedia layouts*, Proceedings of the workshop on Advanced visual interfaces, p.95-104, June 01-04, 1994, Bari, Italy

[16] M. Kurosu, K. Kashimura, *Apparent usability vs. inherent usability: experimental analysis on the determinants of the apparent usability,* Conference Companion on Human Factors in Computing Systems, p.292-293, May 07-11, 1995, Denver, Colorado, USA

[17] M. Zen, *Metric-based evaluation of graphical user interfaces: model, method, and software support*, EICS '13 Proceedings of the 5th ACM SIGCHI symposium on Engineering Interactive Computing Systems, P. 183-186, 2013.

[18] Montero, F., González, P., Lozano, M., & Vanderdonckt, J. (2005, September). *Quality models for automated evaluation of web sites usability and accessibility.* In International COST294 workshop on User Interface Quality Models (UIQM 2005) in Conjunction with INTERACT.

[19] Barry, C., Lang, M.: *A survey of multimedia and web development techniques and methodology usage.* IEEE 8, 2 (2001)

[20] Brajnik, G., *Comparing Accessibility Evaluation Tools: Results from a Case Study.* In: Ardissono, L., Goy, A.(eds.), Proc. of Symposium on Human-Computer Interaction HCITALY'2003 (2003)

[21] Ivory, M. Y., Sinha, R. R., & Hearst, M. A. (2001, March). *Empirically validated web page design metrics.* In Proceedings of the SIGCHI conference on Human factors in computing systems (pp. 53-60). ACM.

[22] Montero, F. and López-Jaquero, V. *Guilayout++: Supporting Prototype Creation and Quality Evaluation for Abstract User Interface Generation,* Proc. of the 1st Workshop on USer Interface eXtensible Markup Language UsiXML'2010, Berlin, Thalès, 2010, pp. 39--44.

[23] Magel K. and Izzat A., "*GUI Structural Metrics and Testability Testing,*" *in Proceedings of IASTED SEA*, USA, pp. 159-163, 2007.

[24] Alsmadi, I., & Al-Kabi, M. (2011). *GUI structural metrics.* The International Arab Journal of Information Technology,8 (2), 124–129.

[25] Comber, T & Maltby, JR 1995, *"Evaluating usability of screen designs with layout complexity,"* in H Hasan & C Nicastri (eds) , Proceedings of HCI, a light into the future : OZCHI '95 , CHISIG Australia, Downer, ACT.

[26] Silva, P., and Paton, N., *Improving UML Support for User Interface Design: A Metric Assessment of UMLi,* Proc. Workshop on Bridging the Gaps Between Software Engineering and Human-Computer Interaction at International Conference on Software Engineering (ICSE '03), 2003.

[27] Coskun, E., and Grabowski, M., *Impacts of User Interface Complexity on User Acceptance in Safety-Critical Systems,* Proceedings of the Tenth Americas Conference on Information Systems, New York, New York, August 2004.

[28] Sobiesiak, R., and Diao, Y., (2010, June 7). *Quantifying Software Usability Through Complexity Analysis*. In IBM Design: Papers and Presentations. {Online}. Available: http://www.ibm.com/software/ucd/otherresources.html

[29] Kokol, P., Rozman, I., and Venuti, V., ***User interface metrics***, Newsletter ACM SIGPLAN Notices, Volume 30 Issue 4, April 1995 Pages 36 – 38.

[30] Tullis, T., *A system for evaluating screen formats,* in R. Hartson & D. Hix (eds.) Advances in Human-Computer Interaction, Ablex Publishing: Norwood, NJ, 1988.

[31] Kim W. and Foley J., *Providing high-level control and expert assistance in the user interface presentation design*, Proceedings of INTERCHI'93 (1993), New York: ACM, 430-437.

[32] Shazia, M., Shoaib, Shah, A., and Majeed, F., *SOFTWARE DESIGN QUALITY METRICS FOR WEB BASED APPLICATIONS*, Pakistan Journal of Science (Vol. 63 No. 1 March, 2011)

[33] Seffah, A., Donyaee, M., Kline, R.B., and Padda, H.K, *Usability measurement and metrics: A consolidated model,* Journal Software Quality Control, V.14 No.2, 2006, 159-178.

[34] Ivory, M. Y. and Hearst, M. A. 2001. *State of the art in automating usability evaluation of user interfaces.* ACM Computing Surveys 33(4): 470-516.

[35] Rosic, M., Mladenovic, S., and Borojevic, L., *Information System User Interface Complexity, Lecture Notes in Computer Science,* 2010, Volume 6389, HCI in Work and Learning, Life and Leisure, Pages 509-512.

[36] Riglise, E. *Modeling visual complexity in image architectures.* Technical Report, Heriot-Watt University (1998).

[37] Geissler, G. L., Zinkhan, G. M., & Watson, R. T. (2006). *The influence of home page complexity on consumer attention, attitudes, and purchase intent.* Journal of Advertising, 35(2), 69-80.

[38] Carroll, J. M. (1997). *Human–computer interaction: Psychology as a science of design.* International Journal of Human-Computer Studies, 46(4), 501-522.

[39] Donderi, D. C. (2006). *Visual complexity: a review.* Psychological Bulletin,132(1), 73.

[40] Tuch, A. N., Bargas-Avila, J. A., Opwis, K., & Wilhelm, F. H. (2009). *Visual complexity of websites: Effects on users' experience, physiology, performance, and memory.* International Journal of Human-Computer Studies, 67(9), 703-715.

[41] Rauterberg, M., *Quantitative Test Metrics to Measure the Quality of User Interfaces,* In Proceedings of 4th European Conference SOFTWARE TESTING ANALYSIS & REVIEW- EuroSTAR'96, 2-6 December 1996, Amsterdam.

[42] Xing, J. (2004). *Measure of information complexity and the implications for automation design*. Washington, DC: U.S. DOT FAA, Tech. Report No: DOT/FAA/AM-04/17.

[43] Victor M. R. Penichet, C.Calero, Maria D. Lozano, M. Piattini. *Using WQM for Classifying Usability Metrics*. Proceedings of the IADIS International Conference WWW/Internet 2006 - ICWI 2006; ISBN: 972-8924-19-4. Murcia, Spain; 05 Oct 2006.

[44] Lo, R., Webby, R., and Jeffery, R., *Sizing and Estimating the Coding and Unit Testing Effort for GUI Systems,* METRICS '96 Proceedings of the 3rd International Symposium on Software Metrics: From Measurement to Empirical Results, 166 – 173, 1996.

[45] McQuaid, Patricia A., *A Complexity Metric to Aid the Software Quality Process,* World congress for software quality, VOL. 1 NO. 0, p 1-10, 1995.

[46] Schneiderman, B., Plaidant, C.: *Designing the User Interface: Strategies for Effective Human-Computer Interaction.* 4th edn. Pearson Education, Inc (2005)

[47] Sobiesiak, R., and Diao, Y., (2010, June 7). Quantifying Software Usability Through Complexity Analysis. In IBM Design: Papers and Presentations. {Online}. Available: http://www.ibm.com/software/ucd/otherresources.html

[48] Coskun, E., and Grabowski, M., *Impacts of User Interface Complexity on User Acceptance in Safety-Critical Systems,* Proceedings of the Tenth Americas Conference on Information Systems, New York, New York, August 2004.

[49] Rauterberg, M. (1992) *An empirical comparison of menu-selection (CUI) and desktop (GUI) computer programs carried out by beginners and experts.* Behaviour and Information Technology 11(4), 227-236.

[50] Rauterberg, M. (1993) *A product oriented approach to quantify usability attributes and the interactive quality of user interfaces.* In: H. Luczak, A. Cakir & G. Cakir (Eds.) Work With Display Units 92. Amsterdam: North-Holland, pp. 324-328.

[51] Rauterberg, M. (1995a) *Four different measures to quantify three usability attributes: 'feedback', 'interactive directness' and 'flexibility'.* In: P. Palanque & R. Bastide (eds.) Design Specification and Verification of Interactive Systems'95. Wien New York: Springer, pp. 209-223.

[52] J. C. Silva, J. C. Campos and J. A. Saraiva, *"Gui inspection from source code analysis,"* Electronic Communications of the EASST33, 2010.

[53] J. C. Silva, C. E. Silva, R. D. Gonçalo, J. Saraiva, J. C. Campos: *The GUISurfer tool: towards a language independent approach to reverse engineering GUI code.* EICS 2010: 181-186

[54] T. DeMarco, *Controlling Software Projects: Management, Measurement, and Estimates,* Prentice Hall PTR, Upper Saddle River, NJ, 1986

[55] J. Noble and L. L. Constantine *"Interactive design metric visualization: Visual metric support for user interface design", OzCHI Proceedings*, 1996.

[56] Mamillapally, N., Mulukutla, T., and Bojja, A. (2013). *A comparative Study of Redesigned Web site Based on Complexity Metrics. International Journal of Computer Engineering and Technology*.

[57] J. C. Silva (2010). *GUISURFER: A Generic Framework for Reverse Engineering of Graphical User Interfaces.* Ph.D. Thesis, University of Minho, Barcelos, Portugal.

[58] Bessiere, K., Ceaparu, I., Lazar, J., Robinson, J., Shneiderman, B.: *Understanding Computer user frustration: Measuring and Modeling the disruption from poor designs* (2003)

[59] Rosic, M., Mladenovic, S., and Borojevic, L., *Information System User Interface Complexity,* USAB'10 Proceedings of the 6th international conference on HCI in work and learning, life and leisure: workgroup human-computer interaction and usability engineering, 2010, Pages 509-512

[60] N. E. Fenton and S. L. Pfleeger. *Software Metrics: A Rigorous and Practical Approach*.

International Thomson Publishing, 1997.

[61] S. L. Pfleeger and C. McGowan. *Software metrics in the process maturity framework.* The Journal of Systems and Software, 1990.

[62] Zhang, Q. (2006) *Improving Software Development Process and Project Management with Software Project Telemetry,* Ph.D. Thesis, University of Hawai'i, USA.

[63] Genero, M. 2002. *Defining and Validating Metrics for Conceptual Models,* Ph.D. Thesis, University of Castilla- La Mancha, 2002.

[64] Loconsole, A. 2007. *Definition and Validation of Requirements Management Measures,* Ph.D. Thesis, UMEA University, Sweden, 2007.

[65] Kitchenham, B.A., Pfleeger, S.L., and Fenton, N. 1995. *Towards a Framework for Software Measurement Validation.* IEEE Transaction on Software Engineering, 21, 12 (Dec. 1995), 929–944.

[66] Donyaee, M. K..; Seffah, A. and Rilling, J., *Benchmarking Usability of Early Designs Using Predictive Metrics,* SMC, 2007, pp 2514-2519.

[67] Melody Y. Ivory, Marti A. Hearst, *The state of the art in automating usability evaluation of user interfaces,* ACM Computing Surveys (CSUR), v.33 n.4, p.470-516, December 2001.

[68] Bertoa, M., & Vallecillo, A. (2004, June). *Usability metrics for software components.* In 8th International Workshop on Quantitative Approaches in Object-Oriented Software Engineering (QAOOSE'2004), Oslo, Norway.

[69] http://www.socscistatistics.com/Default.aspx

[70] http://epitools.ausvet.com.au/content.php?page=home

[71] Davis, F. D., *"Perceived Usefulness, Perceived Ease of Use, and User Acceptance of Information Technology,"* MIS Quarterly, Vol. 13, No. 3, pp.319-339, September 1989.

[72] B. J. Fogg, Hsiang Tseng, *"The elements of computer credibility,"* Proceedings of the SIGCHI conference on Human factors in computing systems: the CHI is the limit, pp.80-87, May 15-20, 1999, Pittsburgh, Pennsylvania, United States.

[73] Adams, D.A., Nelson, R.R. and Todd, P.A. *"Perceived usefulness, ease of use, and usage of information technology: a replication,"* MIS Quarterly, 1992, pp. 227-247.

[74] Van der Heijden, Hans. *"Factors influencing the usage of websites: the case of a generic portal in The Netherlands." Information & Management* 40.6 (2003): 541-549.

[75] Kim, Changsu, Mirsobit Mirusmonov, and In Lee. *"An empirical examination of factors influencing the intention to use mobile payment."* Computers in Human Behavior 26.3 (2010): 310-322.

[76] Selamat, Mohd Hasan, Noraini Che Pa, and Rusli Abdullah. *"E-learning User Interface Acceptance Based on Analysis of User's Style, Usability and User Benefits."* Jurnal Sistem Informasi 9.1 (2013): 6-12.

[77] Constantine, L.L., *"Visual coherence and usability: a cohesion metric for assessing the quality of dialogue and screen designs,"* Proceedings. Sixth Australian Conference on Computer-Human Interaction, vol., no., pp.115-121, 24-27 Nov 1996

[78] Mitrović, N., & Mena, E. *Improving user interface usability using mobile agents.* In Interactive Systems. Design, Specification, and Verification, Springer Berlin Heidelberg. p. 273-287, 2003.

[79] Li, H. F., and Cheung, W. K. *An empirical study of software metrics,* IEEE Transactions on Software Engineering, (6), 697-708, 1987.

[80] Ma, Y. T., He, K. Q., Li, B., Liu, J., & Zhou, X. Y. *A hybrid set of complexity metrics for large-scale object-oriented software systems.* Journal of Computer Science and Technology, 25(6), 1184-1201. 2010.

[81] Bonsiepe, G. A. *A method of quantifying order in typographic design.* Journal of Typographic Research, 2, 1968, 203-220.

[82] Tullis, T. S. (1988a). *Screen design.* In M.Helander (Eds.), Handbook of Human-Computer Interaction Elsevier Science Publishers B.V. (North-Holland).

[83] Kumar, D. S., and Rana, A., *"Assessment of usability metrics for object-oriented software system".* ACM SIGSOFT Software Engineering Notes, 35.6, (2010), pp. 1-4.

[84] S. González, F. Montero, and P. González, *"BaLOReS: A Framework for Quantitative User Interface Evaluation"*, Human–Computer Interaction Series, Springer-Verlag London, pp 127-143, 2013.

[85] Montero, F., López Jaquero, V., "*GUILayout++: Supporting Prototype Creation and Quality Evaluation for Abstract User Interface Generation"*. In: Workshop of ACM SIGCHI Symposium on Engineering Interactive Computing Systems, Berlin, Germany, 2010.

[86] N. James and L. Constantine. *"Interactive design metric visualization: Visual metric support for user interface design",* In Proc. Sixth Australian Conference on Computer-Human Interaction, IEEE, 1996.

[87] L. Chang-Mong and C. OK-Bae, *"Interface Design Technique Considering Visual Cohesion-Rate by Object Unit"*, In Proc. 12th International conference, HCI International 2007, Springer Berlin Heidelberg, PP 72-81, 2007.

[88] L. Sang-Young, *"Abstract Clustering based User Interface Design."* In Proc. International Conference on Convergence Technology, 2(1), pp 445-452, 2013.

[89] P. Ted and M. Jason, *"Wordnet Similarity",* http://maraca.d.umn.edu/cgi-bin/similarity/similarity.cgi, 21 August, 2014.

[90] S. Bird, K. Ewan, and L. Edward, "*Natural language processing with Python*", O'Reilly Media, Inc., 2009.

[91] J. Morato, M. A. Marzal, J. Lloréns, and J. Moreiro, *"WordNet applications",* In GLOBAL WORDNET CONFERENCE, Vol.2, pp 270-278, 2004.

[92] S. Bird, *"NLTK: the natural language toolkit",* In Proc. of the COLING/ACL on Interactive presentation sessions. Association for Computational Linguistics, 2006.

[93] P. Ted, P. Siddharth, and M. Jason, "*WordNet::Similarity - Measuring the Relatedness of Concepts"*, In Proc. of the Nineteenth National Conference on Artificial Intelligence (AAAI) pp 1024-1025, 2004.

[94] *"Natural Language ToolKit (NLTK)",* http://www.nltk.org/howto/corpus.html, 15 Sept, 2014.

[95] M. Niemelä and S. Pertti, *"Layout attributes and recall."* Behaviour & information technology Vol.22 No.5, 353-363, 2003

[96] H. Tim and J. H. Anthony, *"The effects of semantic grouping on visual search",* CHI'08 Extended Abstracts on Human Factors in Computing Systems ACM, pp. 3471-3476, 2008.

[97] H. Tim and J. H. Anthony, "*A computational model of "active vision" for visual search in human–computer interaction"*, Human–Computer Interaction*,* 26(4), pp. 285-314, 2011.

[98] C. J. Mueller, D. E. Komogortsev, L. Feldman, *"Using designer's effort for user interface evaluation"*, In Proc. IEEE International Conference on Systems, Man and Cybernetics, SMC 2009, pp. 480-485, 2009.

[99] J. Rubin, and D. Chisnell, *"Handbook of Usability Testing: How to Plan, Design, and Conduct Effective Tests"*. Indianapolis, IN, USA: Wiley Publishing, Inc., 2008.

[100] J. S. Dumas and J. C. Redish, *"A Practical Guide to Usability Testing*", Portland, OR, USA: Intellect Books, 1999.