Towards Measuring User Experience based on Software Requirements

Issa Atoum[1], Jameel Almalki[2], Saeed Masoud Alshahrani[3], Waleed Al Shehri[4]
The World Islamic Sciences and Education University, Amman, Jordan[1]
Department of Computer Science, College of Computer in Al-Leith, Umm Al-Qura University, Makkah, Saudi Arabia[2, 4]
Department of Computer Science, College of Computing and Information Technology, Shaqra University, Shaqra, Saudi Arabia[3]

***Abstract*—User Experience (UX) provides insights into the users' product perceptions while using or intending to use an application. Software products are known for complexity and changeability, starting from requirements engineering until the product operation. Users often evaluate software UX based on a prototype; however, UX is semantically embedded in the software requirements, a crucial indicator for project success. The problem of current UX evaluation methods is their dependence on the actual involvement of users or experts, a time-consuming process. First, this paper builds a benchmark dataset of UX based on textual software requirements crowdsourcing several UX experts. Second, the paper develops a machine learning model to measure UX based on the dataset. This research describes the dataset characteristics and reports its statistical internal consistency and reliability. Results indicate a high Cronbach Alpha and a low root mean square error of the dataset. We conclude that the new benchmark dataset could be used to estimate UX instantly without the need for subjective UX evaluation. The dataset will serve as a foundation of UX features for machine learning models.**

***Keywords*—User experience; benchmark dataset; requirements engineering; UX evaluation; software engineering**

I. Introduction

The User Experience (UX) is a term used to indicate personnel perceptions and resulting emotions from systems or services [1]. Some of the concepts related to UX are included in the definition of Quality in Use (QinU); the user perceptions could result from a system or software's hedonic and pragmatic qualities. Very often, software UX is related to nonfunctional requirements and usability. Therefore, UX is a broader concept that includes usability, user satisfaction, emotions, and perceptions during the interaction [2]. On the other hand, the UI is considered a mechanism of functionality, usability, reliability, and satisfaction [3]. Consequently, usability is influenced by two orthogonal aspects, GUI and UX. However, the focus of this paper is evaluating the UX even if no UI is already built. Therefore, post evaluations of UX after product release (e.g., [4]) are not considered in this study.

The UX evaluation depends on components or factors of UX models. Some of the most commonly cited factors are proposed by Morville [5], known as the honeycomb model. The honeycomb model is based on balancing context, content, and users. Morville's honeycomb consists of seven factors: useful, usable, desirable, findable, accessible, credible, and valuable. However, Morville's model is generic to any product and is not focusing on software products. Recently, questionnaire-based approaches [6][7], heuristics models [8] [9], and hybrid models [13] were proposed for UX modeling and evaluation.

One of the frameworks that lay a foundation on software requirements and UX requirements is the UX-aware framework of Kashfi *et al.* [10]. The UX-aware framework shows that UX requirements are embedded quality requirements that describe the end user's satisfaction or pleasure of using the software. However, UX is generally evaluated based on a prototype(or a release); therefore, productive UX evaluation models could be used early during the requirements development [11]. Regrettably, the unavailability of measurement metrics or a benchmark dataset breaks the early evaluation. The automation of UX measurement is widely disregarded due to its complexity. In this era, an agent was proposed to automate UX testing for specific predefined tasks of objects finding scenarios in a game application [11].

Machine learning has been extensively adopted in many domains to predict and estimate target variables; however, a useful machine learning model depends on robust and reliable datasets. To the best of our knowledge, there are no existing datasets specialized for UX based on software requirements.

Therefore, the research gap is related to the unavailability of automated UX evaluation methods, which will result in that UX evaluators spending extensive efforts. Therefore, the UX evaluators are regretted using manual UX evaluation, including conducting surveys. Worst of all the evaluators must use an existing prototype to evaluate the UX (using existing evaluation methods). As a result, requirements engineers are not fully aware of any early UX software requirements before getting any prototype. Lacking large datasets of software requirements are still a major challenge for proper cost-effective and rapid application development[12]. Moreover, the elimination of UX-compliant requirements results in UX neglection in the final product [13] or poor resource planning [14]. Therefore, the automation of UX evaluation lacks datasets and proper evaluation models.

The main objective of this paper is to build a dataset that could be used for UX evaluation using machine learning techniques solely relying on textual requirements before an actual software gets developed. We employed four UX experts

to annotate user requirements to a widely accepted scale[1], the User Experience Questionnaire (UEQ) [15][16][17]. The proposed benchmark dataset is a PROMISE-based dataset[18], one of the non-commercial requirements datasets. The research uses the latest expanded PROMISE dataset[19], part of the Open Science Tera-PROMISE repository upgraded earlier [18].

The contributions of this study are critical to software requirements engineering and user acceptance success. The prepared benchmark dataset eases the UX evaluation using machine learning models instead of traditional approaches such as questionnaires or heuristic models. This study is also important for software developers to track the software quality over the software life cycle. Therefore, the overall advantage of using an automated UX system is important for both software consumers and developers. Thereby, saving time and efforts of software developers and providing early feedback to software consumers. The main research question that addresses the UX measurement is as follows:

How UX evaluation could be automated at the early stages of requirements engineering using machine learning?

The paper structure is as follows. Section 2 discusses related works. Then, the methodology of building the benchmark dataset is illustrated in Section 3. Next, Section 4 describes the benchmark dataset reliability and provides preliminary results of machine learning methods. After that, we show the implications of the proposed dataset and its limitations in Section 5. Finally, we conclude the paper in Section 6.

II. Related Work

The UX has been evaluated in several models. Morville [5], one of the leaders in UX, proposed the honeycomb model, which consists of seven factors: useful, usable, desirable, findable, accessible, credible, and valuable. A system is considered usable if its needs are delivered in a simple way considering the user learning curve. If the system fulfills the user's needs it is considered useful. However, a system is considered desirable based on its design and attractiveness. If further information is needed about the system it should be findable, easy to navigate. The system should be accessible even to a user with disabilities. Therefore, an application is considered credible if it is trustworthy. Integrating all of these factors provides a valuable system. However, Morville's model is generic to any product and is not focusing on software products.

Generally, UX evaluation models could be classified into three categories: questionnaire-based approaches, heuristics models, and hybrid frameworks. Questionnaire-based models [6], [7], [15], [20] are known for their simplicity in evaluation and aggregation. In such methods, the users are responsible for the system evaluation with a few questions based on their perceived use of products. For example, the benchmark of the User Experience Questionnaire (UEQ) [15] uses several question items to evaluate the UX. Each item of the UEQ consists of a pair of terms with opposite meanings (e.g., 'not understandable' vs. 'understandable', 'efficient' vs. 'inefficient') on a 7-point Likert scale that each ranges from -3 to +3. With UEQ, several users would evaluate a system prototype to calculate key performance indicators(KPIs) for each software application under study (e.g., [6]). Such KPIs are based on UEQ scores and simple average and summation statistics. Recently a questionnaire survey was developed to measure usability, usefulness, and satisfaction of a chatbot UX [21]. However, the approach of 19] was customized for conversational agents.

On the other hand, heuristics models depend on rules or checklist items prepared and evaluated by experts in the specific service domain. In this category, Yeratziotis and Zaphiris [9] proposed a set of rules (heuristics) to support human-computer interaction experts to evaluate website accessibility. Similarly, online travel agency applications' UX has been evaluated formally with 8 stages [2]; however, this model[2] is application-dependent.

UX has been evaluated in the context of students' applications. The UX has been seen from the angle of usefulness and effectiveness of students teaching systems [22] [23]. For example, Krouska *et al.* [23] proposed a set of rules for students' misconception of HTML to understand the user (student) experience while using an e-learning system. Based on the student experience with the system the proposed model —based on the repair theory— was able to suggest a student learner path. Although heuristic models enable experts to key in needed knowledge to help in UX evaluation, they are generally applied to certain application domains where evaluation checklists exist.

Hybrid models combine the previous methods to measure the UX such as measuring the physiological aspects of users during software usage [24]. The multimodal deep learning model UX framework of Hussain *et al.* [24] depends on sentiment analysis, user feedback, and visual analysis of user action using sensors that detect user's objects on the screen. Koonsanit and Nishiuchi [25] proposed a framework to measure software UX based on facial expression recognition and machine learning; however, their focus is on product sentiment analysis rather than measurement of UX from software requirements [26]. A similar model was proposed by Li and Liu [27] to analyze user eye-movement tracking along with user testing methods to suggest actions for a better user experience. Furthermore, a UX framework for business intelligence (BI) systems interfaces was proposed by Eriksson and Ferwerda [28] to support users' desires and data evaluation. The evaluation of their framework utilizes several KPIs: utility, usability, visual attractiveness, and hedonic quality that covers the whole system development lifecycle. Jang and Han [29] proposed a framework for UX understanding in blockchain services. The UX is defined based on the literature in UX generality and blockchain technological aspects. More specific UX frameworks were proposed for educational games. Leong *et al.* [21] considered the game flow, player context, usability, and learnability along with psychometrically to build an educational UX measurement model.

[1]Google Scholar shows 1,301 citations for UEQ paper (2008): 11/2021.