UiLab, a Workbench for Conducting and Replicating Experiments in GUI Visual Design

NICOLAS BURNY, Université catholique de Louvain, Belgium
JEAN VANDERDONCKT, Université catholique de Louvain, Belgium

With the continuously increasing number and variety of devices, the study of visual design of their Graphical User Interfaces grows in importance and scope, particularly for new devices, including smartphones, tablets, and large screens. Conducting a visual design experiment typically requires defining and building a GUI dataset with different resolutions for different devices, computing visual design measures for the various configurations, and analyzing their results. This workflow is very time- and resource-consuming, therefore limiting its reproducibility. To address this problem, we present UiLab, a cloud-based workbench that parameterizes the settings for conducting an experiment on visual design of Graphical User Interfaces, for facilitating the design of such experiments by automating some workflow stages, and for fostering their reproduction by automating their deployment. Based on requirements elicited for UiLab, we define its conceptual model to delineate the borders of services of the software architecture to support the new workflow. We exemplify it by demonstrating a system walkthrough and we assess its impact on experiment reproducibility in terms of design and development time saved with respect to a classical workflow. Finally, we discuss potential benefits brought by this workbench with respect to reproducing experiments in GUI visual design and existing shortcomings to initiate future avenues. We publicly release UiLab source code on a GitHub repository.

CCS Concepts: • **Human-centered computing** → **Interactive systems and tools**; **User interface toolkits**; *Usability testing*; *Systems and tools for interaction design*; *Ubiquitous and mobile computing systems and tools*; • **Software and its engineering** → Graphical user interface languages; Software testing and debugging;

Additional Key Words and Phrases: Aesthetics, Usability evaluation, User interface evaluation, Visual design.

ACM Reference Format:
Nicolas Burny and Jean Vanderdonckt. 2021. UiLab, a Workbench for Conducting and Replicating Experiments in GUI Visual Design. *Proc. ACM Hum.-Comput. Interact.* 5, EICS, Article 196 (June 2021), 31 pages. https://doi.org/10.1145/3457143

1 INTRODUCTION

Visual design [14, 37] is a influencing factor of software quality [40] contributes to the usability of Graphical User Interfaces (GUI) [1, 55] by manipulating their visual components (*e.g.*, widgets, menus, contents, pictures, videos, banners), their properties (*e.g.*, size, color, typography), and their layout by relying on a variety of techniques borrowed from visual design [18], such as Gestalt properties [37], visual techniques [66], symbolic qualities [26], quantitative measures [49], and aesthetic properties [70]. Therefore, a significant portion of GUI studies concerns GUI visual aspects, which is the scope of this paper. Evaluating the visual design of a GUI is an evaluation

Authors' addresses: Nicolas Burny, Université catholique de Louvain, LouRIM Institute, Place des Doyens, 1, Louvain-la-Neuve, 1348, Belgium, nicolas.burny@uclouvain.be; Jean Vanderdonckt, Université catholique de Louvain, LouRIM Institute, Place des Doyens, 1, 1348, Louvain-la-Neuve, Belgium, jean.vanderdonckt@uclouvain.be.

2573-0142/2021/6-ART196 $15.00
https://doi.org/10.1145/3457143

method without any real user and any real GUI, according to Whitefield's taxonomy [67]. Other evaluation methods, such as mockup-based design [33], model-based evaluation [13, 36], and formal analysis [39], are not in the scope of this paper.

Many aspects of Visual Design for GUIs have been investigated, evaluated, and reported in the literature, as well as their impact on usability and user experience (UX). For example, visual appreciation is one of the first interaction people have with an interface when they first interact with it, an effect occurring within the first 500 milliseconds of the interaction [32, 42]. The impact of visual design on other dimensions such as credibility [72], usefulness [6, 60] or performance [56] has also been demonstrated. This phenomenon propagates itself to other properties, such as repurchase intention [45] or establishing the profile of highly-usable rated websites [29].

One particular research direction that grew considerably in the last few years is the experimental study of GUI visual design [14, 19, 26, 61, 71]. Conducting an experiment in this area roughly follows the same workflow: defining and building a GUI dataset for different configurations (*e.g.*, capturing screenshots in different resolutions on different devices, wireframes, or mockups =[33]), computing visual design measures for the various configurations, and analyzing their results. These experiments make heavy use of data to discover any correlation between GUI features, expressed through metrics or measures[1], and UX dimensions or to develop models predicting the user score for a particular UX facet [29, 50, 63, 73]. A common approach consists of comparing the values computed for the visual design measures to values attributed by humans to these properties.

Despite the growing interest in the field, the experimental study of GUI visual design encounters some problems that hinder its development. The difficulty of creating large GUI datasets with their related user data, or the lack thereof, coupled to the tedious process to build them, limits the reproducibility and verifiability of experiments [47]. Not only it is vital to support various experimental methods for studying GUIs [35], but also these methods should be reproducible [46]. This is a major problem because science is a cumulative process, in which facts are confronted with other facts and new experimental results consolidate theories or, on the contrary, invalidate them. Reproducibility is the cornerstone of the cumulative science and is a requirement in many research settings in order to assess the value of scientific claims [53].

This paper presents UiLab, a cloud-based web application for (semi-)automating the workflow of defining and conducting experiments on GUI visual design. For this purpose, the contributions of this paper are manifold:

(i) Section 2 conducts a comparative analysis of different software for conducting experiments of GUI visual design.
(ii) Section 3 defines a framework for reproducing experiments in GUI visual design based on incremental criteria.
(iii) Section 4 motivates UiLab by describing its underlying conceptual model resulting from a requirements elicitation.
(iv) Section 5 describes the services-oriented software architecture of UiLab.
(v) Section 6 exemplifies the UiLab workflow by demonstrating a system walkthrough.
(vi) Section 7 assesses the UiLab impact on experiment reproducibility by evaluating its impact on design and development workflow.
(vii) Finally, Section 8 provides some conclusions of this work, namely by discussing its contributions, the limitations of the current research work, and the potential improvements for the future.

[1]While the terms "metric" and "measure" both express a computational feature of a GUI, from now on, the term "measure" will be systematically used as it is the commonly adopted term in software testing.

2 RELATED WORK AND BACKGROUND

This section discusses the problem of reproducible research first in science and then in Human-Computer Interaction (HCI). We then review different applications existing in the field of GUI visual design via a Targeted Literature Review (TLR) [11, 12], including their functionalities and workflows.

2.1 Reproducibility of Research

2.1.1 Reproducibility in (Computer) Science, HCI, and GUI Evaluation. *Reproducibility* is the cornerstone of cumulative knowledge and a requirement to confirm scientific truth [24, 53], where new results are confronted with prior findings and theories. Reproducibility is supported by clear-cut description of procedures, methods, experimental designs, and publicly available data. In particular, the lack of public data hinders the reproducibility and verifiability of experiments, since recollecting data may be a tedious endeavour [47]. The reproducibility crisis [43] denotes an actual situation in which scientific experiments are difficult, if not impossible, to reproduce [54].

Many attempts have been made to raise awareness of the importance of reproducibility, to formalize and structure reproducibility in science [24, 44, 46], including in Computer Science (CS) [22] and Human-Computer Interaction (HCI) [25, 28, 68, 69]. For example, Peng [46] considered that a paper is not reproducible unless its artifacts are made available. After that, the next steps require releasing source code [20], and both code and data to increase the potential for reproducibility. This continuum defines a gold standard for reproducible papers. Patil *et al.* [44] suggested a visual notation for expressing to what extent reproducibility is explicitly addressed in a scientific paper. In 2020, ACM [2] updated its definitions for "repeatability" (same team, same experimental setup), "reproducibility" (different team, same experimental setup), and "replicability" (different team, different experimental setup) to characterize and foster reproducibility in experimental CS, but without any operationalization.

The HCI community has equally acknowledged the need for reproducible research. For example, RepliCHI [28, 68, 69] defined *replication* as the "attempt to confirm, expand, or generalize an earlier study's findings" (p. 3525). HCI also welcomes replication studies, *e.g.*, Claes *et al.* [16] replicated an in-the-wild study to evaluate the potential impact on ecological validity of a high fidelity prototype against the final user interface to find out that the prototype was very close to the final interface in terms of appreciation. However, according to the findings of Hornbæk *et al.* [28] on an outlet of 891 HCI papers, a replication rate of just 3% was found (28 studies). Most of the studies (22) replicated the work of other researchers, while the others replicated their own work (2) or did both (6 studies). By analyzing the nature of replications, Hornbæk *et al.* [28] revealed that earlier findings were confirmed, and that comparisons to prior work were often simple. In a follow-up, Hornbæk [27] encouraged the HCI community to embrace replication, especially when the outcomes are likely to question previous results, *e.g.*, "we must be more wrong in HCI research".

Following RepliCHI [68, 69], the term "replicability" has been widely adopted in HCI where "reproducibility" has been used for the same purpose in general science [24]. Although different terminologies co-exist in HCI and other fields to designate reproducible research, this confusion is generalized across the entire science spectrum. Gomez *et al.* [22] reported 27 different frameworks to classify reproducibility.

For example, Hornbæk *et al.* [28] distinguish three replication types (p. 3526): a strict replication reuses the same variables as the original study to repeat it; a partial replication introduces any possible variation in these variables to investigate how the original results still hold or vary in a different setup (*e.g.*, such as in a different context of use); and a conceptual replication applies other measurement methods to the original data to detect any effect. Goodman [24] specifies that

"reproducibility" refers to the ability of a researcher to duplicate the results of a prior experiment using the same materials as were used by the original investigator. Reproducibility is a minimum necessary condition for a finding to be believable and informative.'

Tsang & Kwan [62] classify replications into seven types along two dimensions: sources of data (same data set, same population, and different population change in method same measurement and analysis, different measurement and/or analysis). From this perspective, we are considering the same measurement and analysis. The second dimension is about the sources of data, *e.g.*, reanalyze the data (checking of analysis), collect data again from the same population, or from a different population. While the aforementioned classifications are useful, they do not specify which artefact should vary in order to fulfill the needs of a particular replication type.

In the ACM definitions [2], the "same experimental setup" does not specify whether the same participants should be involved. If it is the case, a participant involved in the original study will never be the same in any further replication because of carry-over effects. In addition, the ACM definitions do not cover the case "same team, different setup". The dimensions are not independent. To be consistent with RepliCHI, from now on, we will use "replication" as an umbrella term to refer to any form of reproducible research in HCI, although the two terms could be used alternatively.

2.2 Evaluation vs. Experimental Research

Evaluation and experimental research processes are quite different in many aspects [7] (Table 1) and the tools suitable for one process may not be suitable for the other.

Evaluation process usually aims at solving practical problems and its goal is motivated by the will to improve the current state of a given situation. The area of interest of the evaluation process is determined by decision-makers or stakeholders that may be external to the group of persons carrying the evaluation. Given that evaluation is targeted at solving problems in a specific context, the generalization of its outcome is limited. The outcome is also transmitted to a limited audience, usually the decision-makers and stakeholders that initiated the evaluation process. The evaluation process is assessed in terms of its relevance to decision-makers and its impact on the improvement of the current situation.

Experimental research is usually dedicated to the development of new knowledge and motivated by the will to improve the current understanding of a given topic. As opposed to the evaluation process, the decision-making process in experimental research is highly researcher-centric and the area of interest is defined by the team carrying the research process. As its results are meant to be published, the targeted audience of the process is larger than in the evaluation process. The assessment of the quality of the research process is often expressed in terms of external validity and reproducibility of results.

More specifically for GUI visual design, an evaluation consists of applying any evaluation method to a single GUI or a small set to determine their quality and choose the best design. It is not particularly intended to contribute to the body of experimental research. Software existing in this field has been implemented mostly for evaluation purposes and less for conducting experimental research with replicability in mind. Of course, any software that can be used for conducting an evaluation could be repeatedly used to conduct some experiment.

2.3 Software for Experimental Study and Evaluation in GUI Visual Design

Replicability is a matter of habits and good practices on the part of the researcher, but it is also related to the tools and technologies that makes the processes efficient and realistic. Considering the experimental study of GUIs visual design, there exists several tools allowing to partially automate of the dataset construction process. These tools mainly focus on either computing measures on GUIs or gathering data about users evaluation of GUIs and a lot of tedious tasks such as building

Criteria	Evaluation	Experimental Research
Aim	Solving practical problems	Knowledge development
Area of interest	Defined by decision-makers	Defined by researchers
Motivations	Improve current state of a situation	Improve understanding & develop theories
Generalisation	Limited, oriented towards specific situations	High, interest in generalisation to different situations
Output	Transmission of findings to decision-makers; limited audience	Publication of results; large audience
Assessment	Relevance to the decision-makers, impact of results on the current situation	External validity, reproducibility, test of hypotheses

Table 1. Differences between Evaluation & Experimental Research processes.

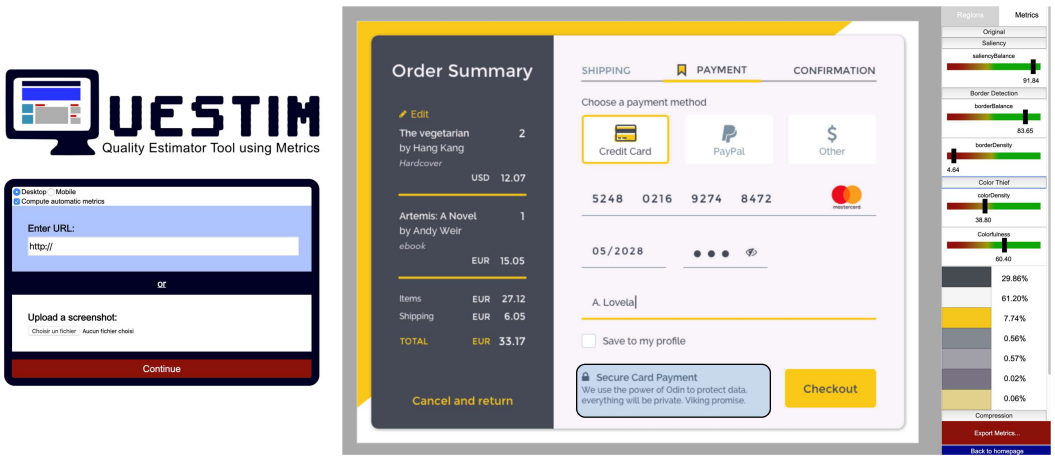

Fig. 1. Questim: (a) introductory screen; (b) evaluation screen.

experiments or capturing screenshots are left to the experimenter. Most of the tools available in the field of GUI visual design have been built with evaluation in mind, which makes them not fully suitable for experimental purposes.

We provide an analysis of of the different tools in the field of GUI visual design and their characteristics. We evaluate the functionalities of these tools regarding a list of features provided in Section 2.4 and we summarize this evaluation in Table 2.

2.3.1 Questim. Questim [70] is a web-based application for evaluating GUI measures. The tool is implemented in Java and is compiled for Javascript using Google Web Toolkit. In order to provide designers with an objective feedback regarding the visual design of their GUIs, such as for web pages, Questim enables the end user to specify a website URL or upload a file containing any GUI artefact, such as a screenshot, a wireframe, a sketch, a picture, a prototype (Fig. 1). After defining graphical regions of interest (*e.g.*, a widget, a group box, a menu, an image) by direct manipulation, visual design measures are automatically computed (Figure 1b). Any change in the location, sizing, and arrangement of a region triggers the re-computation of the measures displayed on a color gradient formatted depending on the distribution of the measure. For example, the density measure will orient to red when the value is low or high, but becomes green slight above the middle.

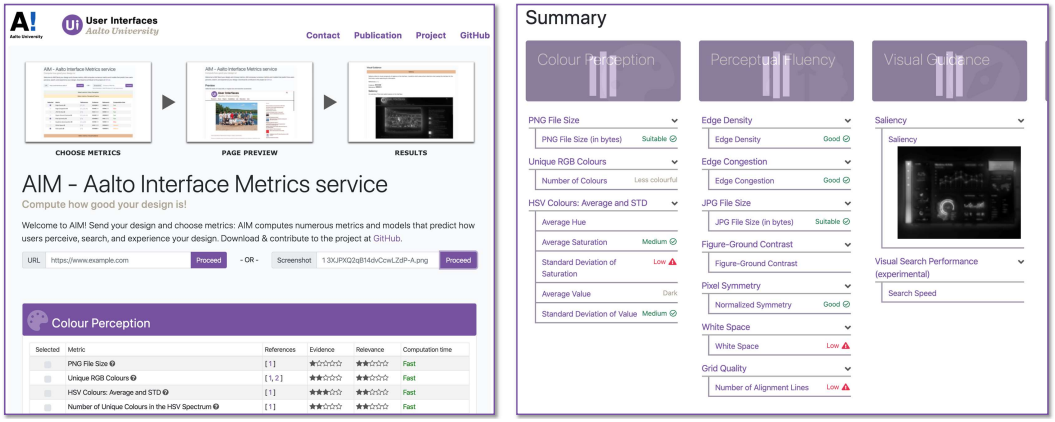

Fig. 2. Aalto Interface Metrics (AIM): (a) Measure selection; (b) Summary of computed measures.

2.3.2 Aalto Interface Metrics (AIM). AIM [41] is an online web application for the computational evaluation of GUIs. The goal of the application is to facilitate the use and appropriation of computational methods in design practices. After specifying the screenshot or an URL (Fig. 2a), the end user can choose the measures to be computed automatically on designated elements. The screen resulting from the computational evaluation (Fig. 2b) appears with the values of the selected measures in real-time.

2.3.3 Web UI Visual Analyzer (WUI). WUI [8] consists of an online web application integrating measures computed from different providers for evaluating GUI visual design. The application is composed of a front-end and a back-end, as for AIM and Questim. For this purpose, WUI is able to work with different remote services such as AIM as major providers for visual design measures. WUI allows capturing screenshots automatically by specifying the URL or GUI screenshot, but only one sample can be captured and analyzed at a time. As for AIM and Questim, WUI only considers the computation of measures. To conduct the experiment they report in the the original paper, authors had to run WUI a significant amount of time with different parameters. End users can add a new measure to the application by registering it in the system according to a specified procedure. The measures can be self-implemented or provided by a remote service. However, the whole system must be redeployed in order for changes to take place.

2.3.4 LabInTheWild. LabInTheWild is a platform hosting various types of experiments. It has been used, among other experiments, to collect user data about perceived visual aesthetics and to model several GUI features [50]. This platform provides some support for building the GUI dataset and the user data collection. Participants can take part to any experiment deployed on the platform and are instructed to fill in a pre-test survey about their socio-demographical profile before taking part to the experiment. Given the variety of experiments available on the platform, the existence of a template-based experiment generation engine behind the platform seems unlikely. Such capability would reduce the variety of the experiments that can be deployed, but would increase the reproducibility of their setup and reduce their development and deployment time.

2.3.5 BaLOReS. BaLOReS [23, 34] is a framework composed of five structural principles and their associated measures. These principles help designers to structure their mockups and produce well-designed, pleasing GUIs to improve user's subjective satisfaction. The framework comes with a prototyping tool, BGLayout, that allows assessing the quality of interfaces design through metrics. BGLayout has been developed in Java and is a desktop application, as opposed to Questim, AIM, and WUI that are web-based applications. BaLOReS builds a layout based on several "screen areas",

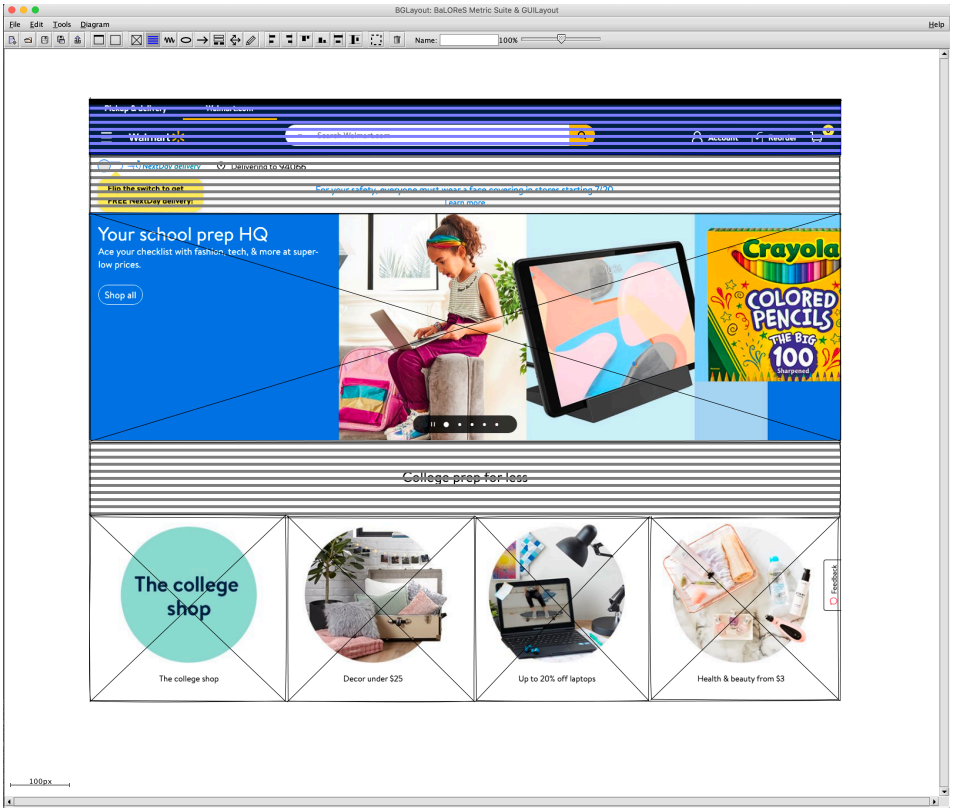

Fig. 3. Screenshot of the BGLayout interface.

which can be of different types (*e.g.*, text, image, form). The layout can be designed from scratch or based on a screenshot (Fig. 3). BGLayout allows computing five visual measures (*i.e.*, balance, regularity, linearity, sequentiality, and orthogonality) on the layout in order to quantify different aspects and improving its design by putting it in correspondence with users' perceptions about the visual appearance. However, this comparison with users' perception must be done externally as the application does not support collecting user-related data. Hence, the number of measures is fixed and the software does not accommodate any new measure in its current form.

2.3.6 PLAIN. PLAIN [57] is an Eclipse-based plugin that automatically computes eight measures, called *defects*, used to predict the usability of a mobile GUI: incorrect layout of widgets, overloaded GUI, complicated GUI, incorrect data presentation, incohesive GUI, difficult navigation, ineffective appearance of widgets, and imbalance. The measures are transposed from desktop GUIs to mobile GUIs based on Ngo's measures [38]. The source code for computing these measures is embedded in the plugin, thus requiring code modification and re-compilation if any measure should be added, deleted, or modified. PLAIN is based on the genetic algorithm technique used for the generation of evaluation rules. PLAIN uses the GUI source code to perform the computation of the measures and the evaluation of the interface, contrarily to Quesтim, AIM, and WUI which use a graphical representation of the GUI to compute the measures. Any user data collection has to be performed outside of the application. PLAIN later on proceeded to ADDET [10], which consists of two modules: a first module that statically computes thirteen visual design metrics and a second module that analyzes deviations from reference measures to detect defects in GUIs for mobile devices.

2.3.7 GUIEvaluator. GUIEvaluator [5] is a desktop application for evaluating the GUI complexity based on its structure. The application automatically computes five structural measures. The tool, developed in Visual Basic 2012, extracts the interface layout information using reflection techniques supported in the language. The tool can be used at design- or at run-time. The application does not allow collecting user-related data.

2.3.8 GUIExaminer. GUIExaminer [4] automatically computes the Screen Layout Cohesion (SLC) measure to predict its usability based on different aspects of the GUI to measure its quality. As for GUIEvaluator, the tool is developed in Visual Basic and is only applicable to Visual Basic GUIs since the source code of the interface is required to extract the widget properties and the layout type.

2.3.9 UI-CAT. UI-CAT [51] is a background service developed for the Android OS that allows computing automatically complexity metrics for Android mobile applications. The tool records users' touch events and take screenshots of the opened application upon interaction without the need to modify the application's source code. Aside the user touch events, the application does not allow the collection of user data and is focused on measures computation.

2.4 Comparative Analysis of Software

Table 2 compares the aforementioned tools and applications on the basis of a set of selected common functionalities and we qualify the extent to which eight comparison criteria are supported in the different tools via Harvey's balls [30]. The bottom line represents UiLab's coverage of these criteria, highlighting the differences between the functionalities of the systems "as-is" and the system "to-be". When a criterion is not applicable for comparison for a specific tool (*e.g.*, the corresponding feature is not implemented), "NA" will be mentioned in the corresponding cell.

From Table 2, we identify several shortcomings from the existing tools and application in the field of GUI visual design. All of them process only a single type of input at a time manually provided, be it a screenshot or a piece of source code, thus repeating the same process for conducting an experiment where a large number of GUIs need to be measured. Not all aspects of GUI visual design can be evaluated or measured at once using a single type of representation of the interface. Such a restriction could be considered as an important limitation when studying GUI visual design. These applications do not process inputs concurrently or in batches (*e.g.*, via a parallel computation of measures over a dataset of GUIs), thus lengthening the preparation phase. Perhaps this feature was not required for these applications as soon as they were primarily designed for evaluation and not for experimental research (see Section 2.2). If we consider experimental research of GUI visual design as a target, such applications are not suitable because they do not automatically process large datasets.

(1) **Inputs**
  (a) Diversity: refers to the input types recognized by the application.
    - ○ - Low: the application only recognizes one type of input.
    - ● - High: the application recognizes several types of input.
  (b) Scalability: refers the the ability to process inputs concurrently.
    - ○ - Low: the application only allows processing one input at a time.
    - ● - High: the application allows processing multiples inputs concurrently.

(2) **User Data Collection**
  (a) Diversity: refers to the variety of data types that can be collected.
    - ○ - Nonexistent: the application does not allow the collection of user-related data.
    - ◑ - Low: the application allows collecting only one type of user-related data
    - ● - High: the application allows the collection of a variety of user-related data

	Inputs		User data collection			Measures computation		
	Diversity	Scalability	Diversity	Customisation	Automation	Diversity	Extensibility	Automation
QUESTIM	○	○	○	NA	NA	○	○	○
AIM	○	○	○	NA	NA	○	○	●
WUI	○	○	○	NA	NA	○	○	●
LabInTheWild	NA	NA	●	●	○	NA	NA	NA
BaLOReS	○	○	○	NA	NA	○	○	○
PLAIN	○	○	○	NA	NA	○	○	●
GUIEvaluator	○	○	○	NA	NA	○	○	●
GUIExaminer	○	○	○	NA	NA	○	○	●
UI-CAT	○	○	◑	○	●	○	○	●
UiLab	○	●	●	●	●	●	●	●

Table 2. Level of support of visual variables expressed according to Harvey's Balls.

(b) Customization: refers to the extent to which the data collection process is customizable.
- ○ - Low: the data collection process accepts little or no parametrization.
- ● - High: the data collection process is highly parametrizable.

(c) Automation: refers to the ease of deployment of the data collection process.
- ○ - Low: the data collection process has to be deployed manually.
- ◑ - Medium: the deployment of the data collection process is partially automated.
- ● - High: the deployment of the data collection process is automated.

(3) **Measures computation**

(a) Diversity: refers to the variety of programming languages supported by the application for measures implementation.
- ○ - Low: the application only supports one programming languages for the implementation of measures
- ● - High: the application supports multiple languages for measures implementation.

(b) Extensibility: refers to the ease of extensibility of the application with new measures.
- ○ - Low: the addition of new measures is time-consuming / difficult. The application must be rebuilt / redeployed.
- ● - High: The addition of new measures can be done without any change in the application nor rebuild / redeployment.

(c) Automation: refers to the level of automation of measures computation.
- ○ - Low: the computation of measures is partially automated but still requires human intervention.
- ● - High: the computation of measures is completely automated.

Only one application enables collecting user-related data with a high degree of diversity. Again, this may not be part of requirements of application built for evaluation purpose but experimental studies on GUI visual design require user-related data to analyze any potential correlation between these data and values of measures. Most applications implement their measures by hard-coding them in the application via a single programming language. Consequently, the set of languages for implementing the computation of measures is de facto limited to the language used to develop the application itself. In some cases, it also limits the type of GUIs to be analyzed, such as for Visual Basic GUIs. Any modification of existing measures, any addition of a new measure, or any combination of existing measures imply direct manual programming of the source code of the application and the need to re-compile, re-build, and re-deploy it in order for changes to take place.

In the next sections, we explain how UiLab aims at solving these issues by describing its underlying model, its flexible software architecture and by illustrating the system via a walkthrough of its functionalities.

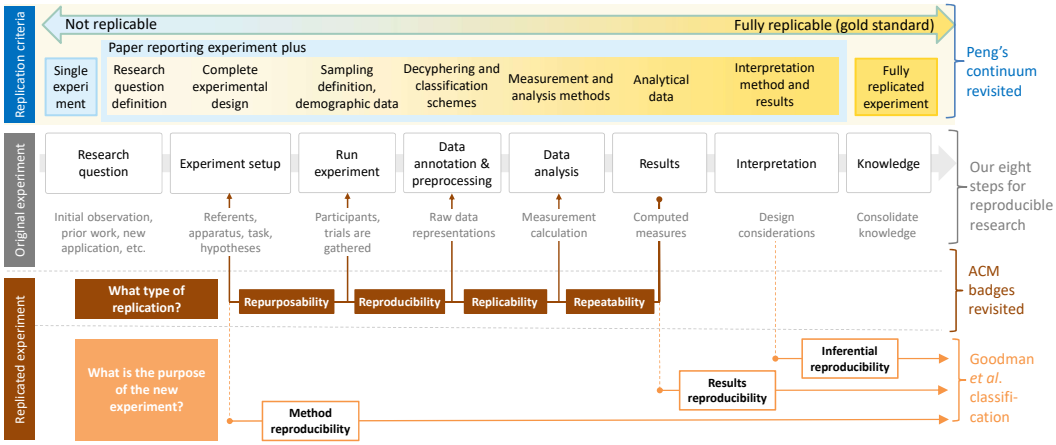

Fig. 4. Our framework for experiment replicability.

3 OUR FRAMEWORK FOR EXPERIMENT REPLICABILITY

Since the ACM definitions are not independent, we define four properties (see Figure 4, brown lane): "repeatability" (same team, same experimental setup), "replicability" (different team, different experimental setup), "reproducibility" (different team, same experimental setup), and "repurposability" (same team, different setup). Three original types of replication are recommended (see Figure 4, bottom part in brown): method reproducibility strives for making the raw data and the experimental setup totally explicit and accessible to repeat the same experiment; if the same results would be obtained by matching this setup, results reproducibility would be satisfied; and inferential reproducibility would reach to the same conclusion resulting from an independent replication or a reanalysis of the obtained results.

Finally, the upper blue part of Figure 4 depicts seven levels of possible replications by revisiting Peng's continuum for fully replicable research [46]. Similarly, we argue that six types of experiment artefacts should become explicit and accessible to enable progressively more elaborated levels of replication, similarly to the three levels of W3C accessibility[2]. For this purpose, we introduce the testable *replicability criteria*:

- Level **A**: the complete experimental setup should be made explicit and accessible. Various techniques exist for this purpose, such as templates for research questions and hypotheses development, such as the FINER criteria in PICOT format [21] and preregistration of CHI experiments [17].
- Level **B**: equals "Level A" plus the explicit definition of the population sampling, demographic data, the context of use, and the stimuli. All these data should be made available through electronic documents in commonly used formats [15]. Instead of capturing data by picture or video (*e.g.*, taking a manual screenshot), it is advisable to actually record GUIs on the very device used, which may require a common, shareable format. In absence of a commonly used format, log files and raw data should be made available in any repository or in an accessible database.
- Level **C**: equals "Level B" plus the publicly available definition of any decyphering or classification scheme used for clustering GUIs into groups. The classification criteria and their

[2]See https://www.w3.org/WAI/standards-guidelines/wcag/

potential values should be clearly defined without any overlapping. Again, a common software platform for managing GUIs and their evaluation would be welcome, like Galaxy for biomedical engineering [3], allowing the designer to semi-automatically explore and test alternate decyphering schemes, without the need to perform this classification manually.

- Level **D**: equals "Level C" plus the definition of the measurement and analysis methods used for analyzing data, *e.g.* through the formalization of measurement methods, tools, formula, and units, a common practice in software quality assurance [9].
- Level **E**: equals "Level D" plus the accessibility of any manual or software tool for applying the measurement methods. For instance, a software could automatically compute all the required GUI measures and provide the designer with some interpretation, like in WebTango [29].
- Level **F**: equals "Level E" plus the interpretation scheme followed to derive the conclusion from analytical data. Note that a same measurement method, perhaps through different formula, may reach to comparable conclusions (confirmation) or not (disconfirmation).

A fully replicated experiment is obtained when "Level F" is reached, which is quite demanding but rewarding. We note that most experiments in GUI visual design reach only levels A or B, but not further away. While recommendations for replications in HCI [28] are still valid and applicable, our replicability criteria are aimed at establishing criteria that are located at a higher level of methodological abstraction than these guidelines. Greiffenhagen & Reeves [25] observed that CHI privileges novelty over consolidation. In our context, this means that merely performing "yet another GUI visual design evaluation" would not be considered as a new contribution [48]. For these reasons, the replicability criteria are cumulative to foster incremental experimentation instead of a large pool of separate, independent experiments, which are poorly related to each other. This is indeed the case for experiments based on GUI visual design measures: a significant body of papers exists that compute various sets of GUI measures, but they are rarely, if ever, put in perspective with another paper, thus making them hard to compare. All these initiatives are commendable, but they do not support any replication.

4 THE UILAB WORKBENCH

4.1 Introduction and Motivations

Existing softwares described in section 2.3 mainly focus either on the computation of measures on user interfaces or on the collection of data about users perceptions of different facets of GUIs visual design. These tools fit well in the GUI evaluation process but when it comes to the construction of datasets for experimental studies, a lot of time-consuming tasks such as experiment design and deployment or screenshot collection still have to be performed manually. In this section, we describe UiLab, a software aimed at facilitating the workflow for experimental studies in GUI visual design, by automating feasible stages, such as the construction of datasets, and supporting explicitly the stages which cannot be automated.

4.2 Definition of UiLab Conceptual Model

The application is designed based on a conceptual model, expressed as an Extended Entity-Relationship (EER) diagram [59], composed of seventeen concepts as follows:

Gallery: A gallery is a set of screenshots that are logically grouped together. A gallery can be associated to 0, 1 or more experiments. Screenshots contained in the gallery will be subject to user evaluation in the experiments based on that gallery. Galleries can also be associated to workflows by the intermediate of "Run" entities.

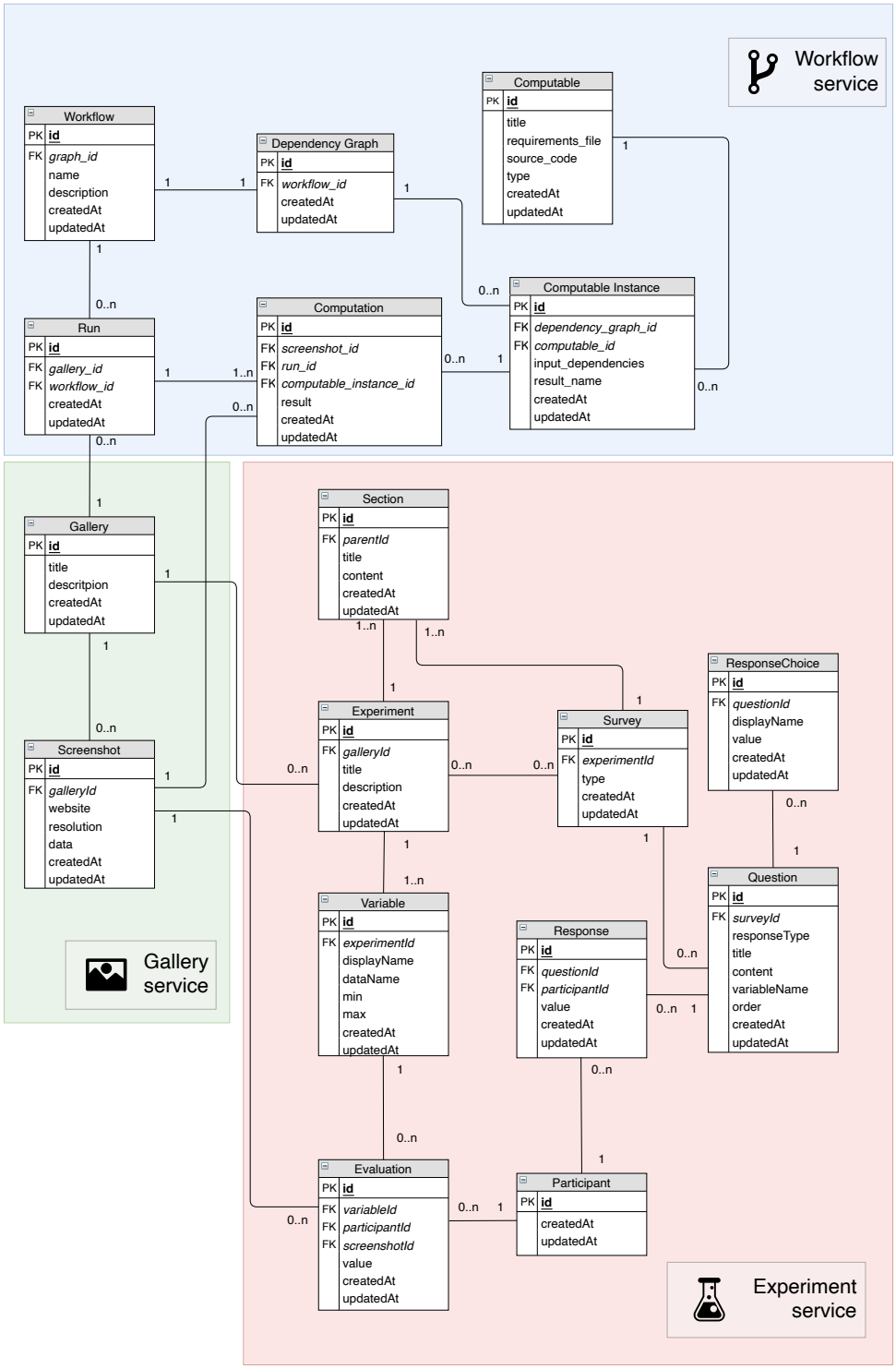

Fig. 5. UiLab- Extended Entity-Relationship (EER) conceptual model.

Screenshot: A screenshot is the captured graphical representation of an user interface. A gallery is composed of screenshots captured from different URLs and of various resolutions and densities.

Experiment: The experiment encompasses the elements that are required to design and deploy an online experiment allowing users to evaluate screenshots of the associated gallery. An experiment is characterized by a title and a description and can be associated with different sections, surveys, and variables. An experiment must be associated to one and only one gallery.

Section: A section is a paragraph of text displayed to the user during an experiment. It can be used to introduce the experiment, explain the objectives of the experiment to the participant, etc. A section must be part of an experiment or a survey and is described by a title and contents.

Survey: A survey is an optional step of an experiment used to gather information about participants. For example, it can be used to collect socio-demographical information such as age, gender, etc. Experiments have usually 2 optional surveys (a pre-test survey occurring before the experiment and a post-test survey occurring after the experiment). A survey is characterized a set of sections and a set of questions to be asked to the participants.

Question: A question is asked to the participant in the context of a survey. The question is characterized by a title, a content describing the object of the question, its order among all the other questions asked in the survey and the type of response displayed by the question (text, number, drop-down list, radio buttons, etc.).

Response Choice: A choice represents the different items proposed to the user as a potential response in the context of a question displayed the different possible responses as a drop-down list of radio buttons. A typical example would be the possible answers to a particular question.

Response: A response corresponds to a participant's answer to a specific question of a survey. It is characterized by the question and participant it is related to and the actual value of the response given by the participant.

Variable: This entity represents a variable that an experimenter wants to be evaluated by participants in the context of an experiment. Several variables may be evaluated for each screenshot during an experiment. For example, an experimenter could ask participants to evaluate both the complexity and the beauty of a user interface, both being represented by a variable during the configuration of the experiment by the experimenter. Variables are characterized by a display name which will be displayed to the participant at evaluation time (*e.g.* "Complexity") and a name, which will refer to the variable in datasets (*e.g.* "complexity_variable"). They are also characterized by the minimum and maximum values they can take in the context of an evaluation by an user. Variables will typically be displayed to participants as a Likert scale [31] with minimum and maximum values enforced by the application.

Participant: A participant represents an individual taking part to experiments in the application.

Evaluation: An evaluation represents the evaluation of a specific screenshot by a participant regarding a specific variable in the context of an experiment. As experimenters can ask participants to evaluate several variables for each screenshot during an experiment, it can lead to several evaluations per screenshot, one for each variable evaluated. An evaluation is characterized by the participant, the screenshot and the variable it is related to and the actual value given by the participant for the screenshot and the variable considered.

Run: A run corresponds to the request of an user to compute a given workflow on a gallery. Each time an user asks the application to compute a workflow on a gallery, a run entity is created.

Computation: A computation represents the actual computation of a computable instance on a specific screenshot. This entity is characterized by the screenshot and the computable instance it is related to and the result of the actual computation.

Workflow: A workflow represents a set of computable instances grouped together that can be computed on screenshots belonging to a gallery. The computation of a workflow on a given gallery implies the computation of all the computable instances contained in the workflow on all screenshots contained in the gallery. For each screenshot, the order of execution of computable instances is determined by the dependency graph associated to the workflow.

Dependency Graph: A dependency graph is an entity containing the dependencies between the computable instances of a workflow. A dependency graph indicates which functions are to be computed on a given gallery and in which order these functions must be computed.

Computable: A computable represents an object that can be computed on screenshots or derived results. A computable can be associated to workflows by the intermediate of their dependency graph. A given workflow can be associated multiple times with the same computable by adding several computable instances of the computable to the dependency graph. A computable can take several parameters as input and returns a result. This result will be passed to subsequent computable instances according to the dependency graph of the workflow. Not all computables are computed directly on screenshots as some may require intermediate computations in order to be executed with the rights parameters.

Computable Instance: A computable instance is an entity representing the actual integration of a computable into a workflow. A computable instance is represented by the dependency graph and the computable it is associated to and the computable instances it depends on. A dependency graph can have multiple computable instances . Computables can be related to multiple computable instances, in different dependency graphs but also inside the same dependency graph.

4.3 Requirements Elicitation

UILAB has been built to meet most of the criteria of the replicability framework described in section 3. Each level of replicability is supported in UILAB by a series of requirements elicited hereafter and represented graphically using goal-oriented requirement specification [64, 65]. A detailed listing of the requirements [52] can be found in the appendix.

4.3.1 Level A requirements. The level A of the replicability framework we propose in section 3 is related to the explicit definition and accessibility of the experimental setup. The application UILAB supports this level of replicability by allowing the user to create experimental setups. The application also allows the user to download the meta-data of the setup in JSON format to make it easily accessible and replicable by other users. The creation of experimental setup implies the ability for the user to specify the content for the different entities composing an experiment in UILAB, namely the experiment, section, survey, question, and response choice entities depicted in Figure 5. The requirements for level A are depicted in Figure 6.

4.3.2 Level B requirements. The level B of the replicability framework is related to the explicit definition of population sampling, demographic data about experiment participants, the stimuli, etc. These sources of information can be though of in UILAB as the data used by and generated by an experiment. UILAB supports the level B of the proposed framework by making accessible to the user the captured screenshots with their resolution (i.e the stimuli), the responses of participants to

surveys' questions (*i.e.* the demographic data / population sampling) and the evaluation of participants recorded in the context of experiments. The access to these sources of information implies that users must be able to participate to experiments for generating data. This also implies that the platform must be able to capture screenshots in order for them to be evaluated by participants in the context of experiments. Another requirement is the ability for the user to download the responses and evaluations of the participants through the application. Finally, the support by UiLab of level B of the replicability framework implies the application supports the level B of replicability in the first place. The requirements for level A are depicted in Figure 7.

4.3.3 Level C requirements. The level C of replicability framework concerns the definition of classification schemes for GUI clustering. The clustering of GUI screenshots in UiLab is implemented by the use of galleries. Gallery entities aggregate screenshots with common properties into collections. UiLab allows CRUD operations on galleries and for the download of the data related to a given gallery. The requirements for level C are depicted in Figure 8.

4.3.4 Level D requirements. Level D of replicability framework is related to the definition of measurements and analysis methods. UiLab is only concerned by the definition of measures as the purpose of the application is not to provide analysis tools to the user but rather to easily collect data to be analyzed in an external application. UiLab meets the requirements of level D by providing the user with the possibility to compute measures on screenshots. Meeting the requirements for level D implies that the requirements for level C are fulfilled and that the user is allowed to perform SCRUD operations on entities related to measures computation (workflow, computable, computable instance, and run). The requirements for level D are depicted in Figure 9.

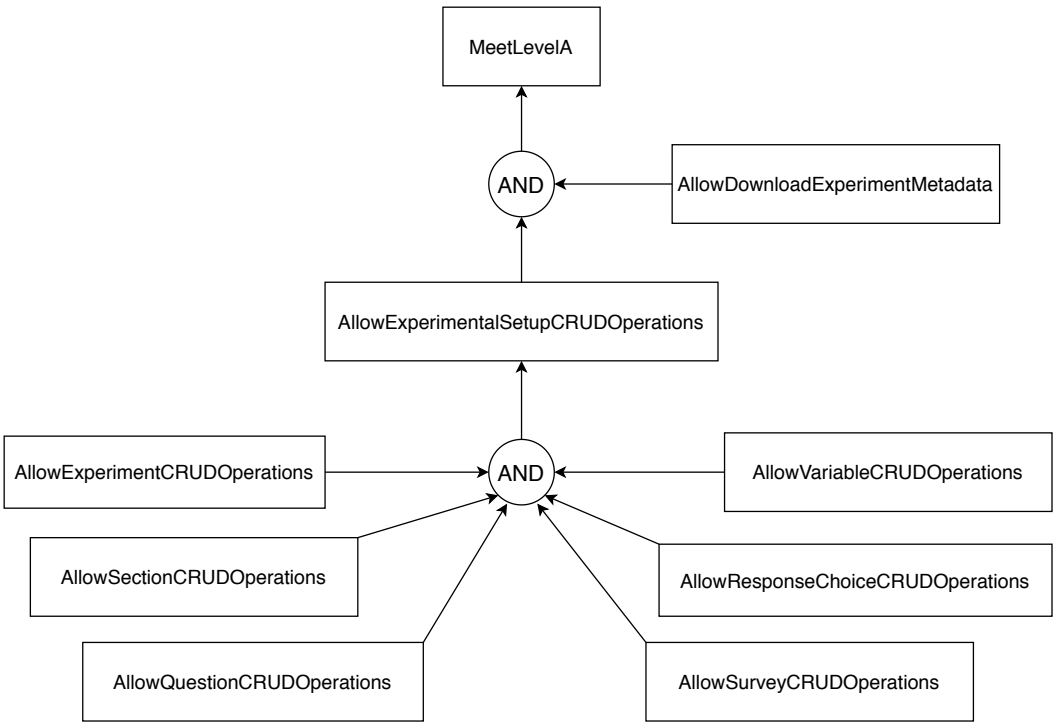

Fig. 6. Requirements for level A of replicability framework.

4.3.5 Level E requirements. The level E of replicability framework is related to the documentation of the tool and procedures. Regarding this, UiLab meets the requirements by providing its documentation and source code on the following GitHub repository : https://github.com/uilab-app.

4.3.6 Level F requirements. The level F of replicability framework is related to the interpretation of results and the scheme followed to derive conclusions from analytical data. UiLab is not meant to be an analytical platform and thus does not implement the requirements to reach level F of replicability. Reaching this level of replicability is the responsability of the researcher using the platform, as she must clearly document the reasoning scheme and its conclusions in a scientific document.

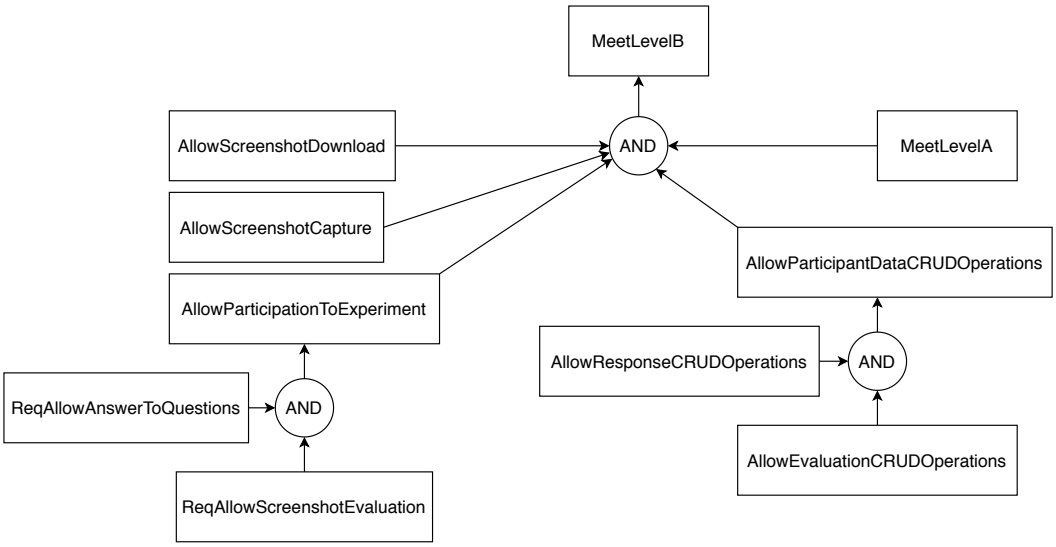

Fig. 7. Requirements for level B of replicability framework.

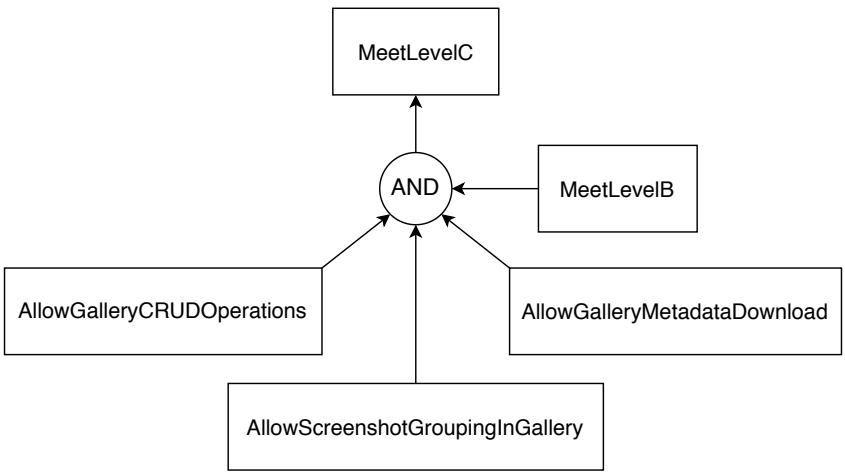

Fig. 8. Requirements for level C of replicability framework.

5 WORKBENCH IMPLEMENTATION

We have considered several architectures and technological stacks during the conception of UiLab. We retained the MERN stack (MongoDB-Express-React-Node.js) [58]. The back-end side of our application UiLab is built according to a service-oriented architecture. The computation of measures is outsourced to the Amazon cloud[3] by using the AWS Elastic Container Service (ECS)[4]. This service allows for the creation of container-based computational instances. We use custom Docker images for measure computation and screenshot capture. The front-end side of UiLab is divided in two different applications. The first application allows experimenters to administrate their galleries of screenshots, experiments, and workflows. The second application allows participants to take part to deployed experiments. Each experiment possesses its own URL shared among participants. We give a detailed explanation of each of the services composing the back-end of UiLab.

Gallery service: this service is in charge of managing galleries and screenshots. Upon request coming from the gateway, it will create or delete galleries and will take screenshots according to the URL and resolution specified in the request. The captured screenshots are stored in AWS S3.

Experiment service: this service is in charge of managing the creation of experiments by experimenters and is also in charge of allowing participants to take part to deployed experiments.

Workflow service: the workflow service is in charge of managing workflows, dependency graphs, computables and computable instances. It is also responsible for the communication with AWS ECS for the delegation of measure computations.

[3]See https://aws.amazon.com/ec2/

[4]See https://aws.amazon.com/ecs/features/

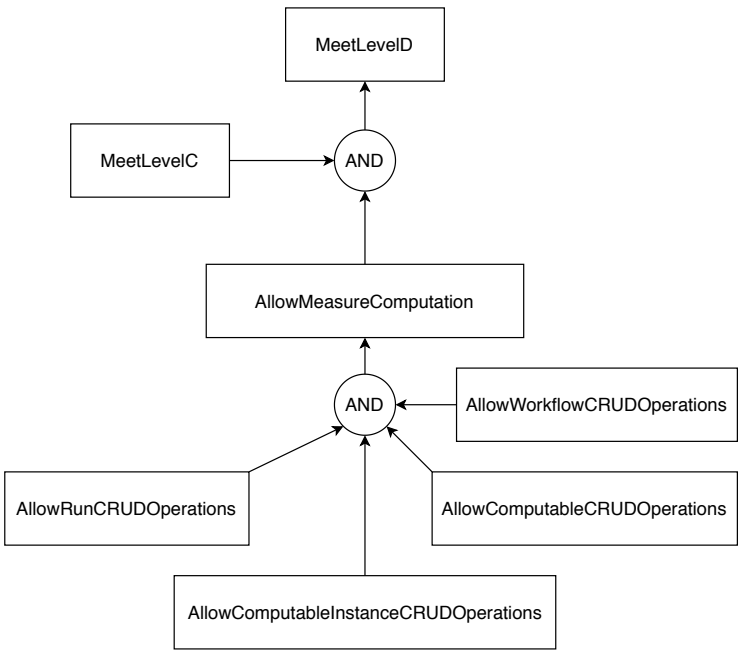

Fig. 9. Requirements for level D of replicability framework.

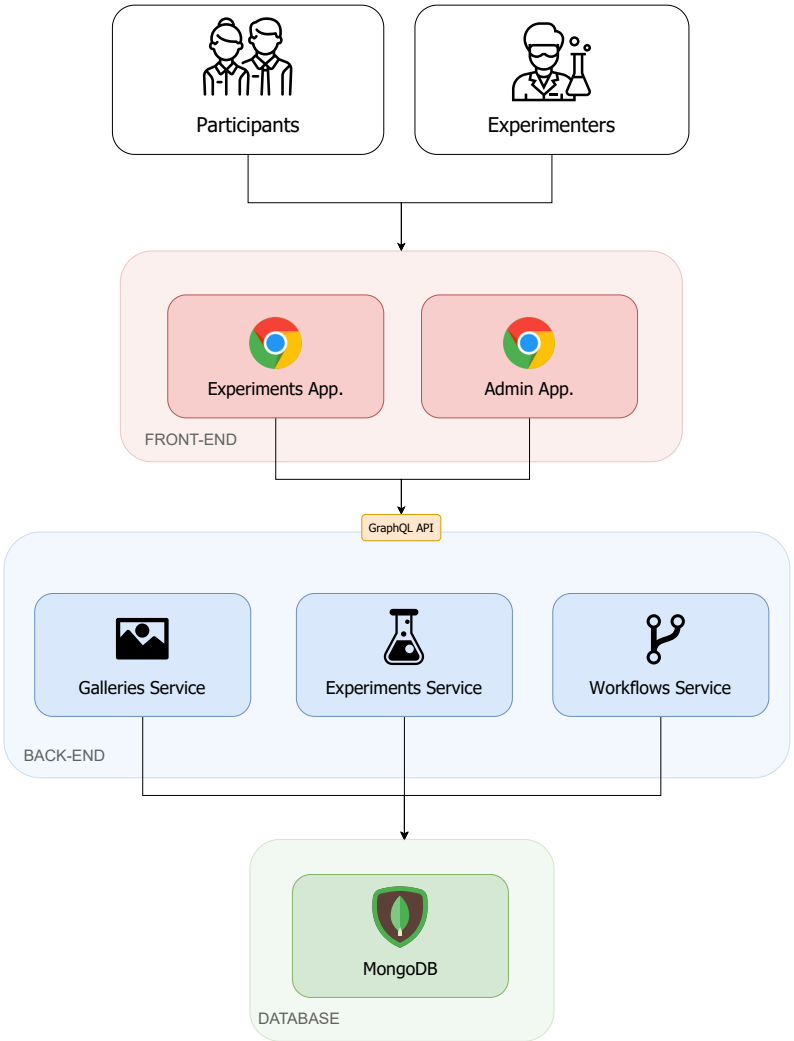

Fig. 10. UiLab- Services-oriented Architecture Diagram.

Figure 5 describes the partition of the data model among the different services.

6 SYSTEM WALKTHROUGH

6.1 Construction of Galleries: Automation of the Capture Process of Screenshots

In UiLab, experimenters have to create a gallery prior to any experiment design. Experimenters have to specify a title and a description for the gallery before validation (Figure 11). Once the gallery is created, experimenters have the possibility to add web sites and new resolutions to the gallery by specifying the URL and width, height, pixel density and if the screenshot has to be taken on an emulated mobile device or not respectively (Figure 12a). Upon validation, the back-end services will take care of capturing the screenshots with the specified parameters. Once the screenshots captured, the experimenter manages them in the corresponding gallery page (Figure 12b).

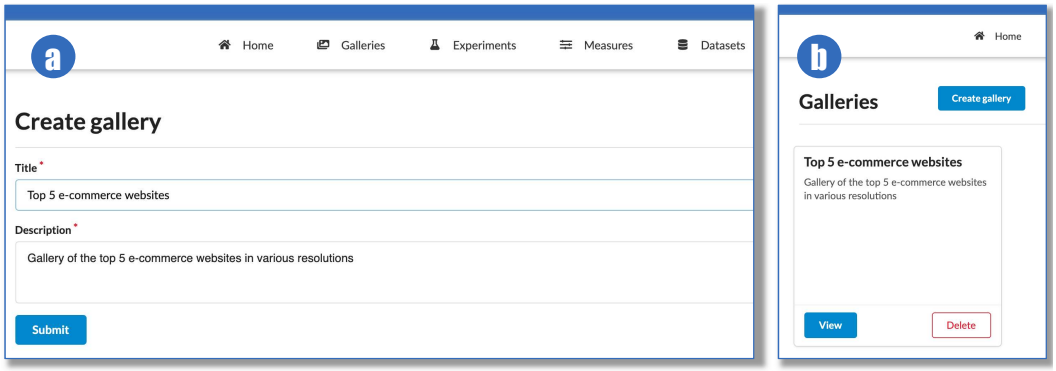

Fig. 11. UiLab: (a) Gallery creation screen, (b) Galleries overview screen.

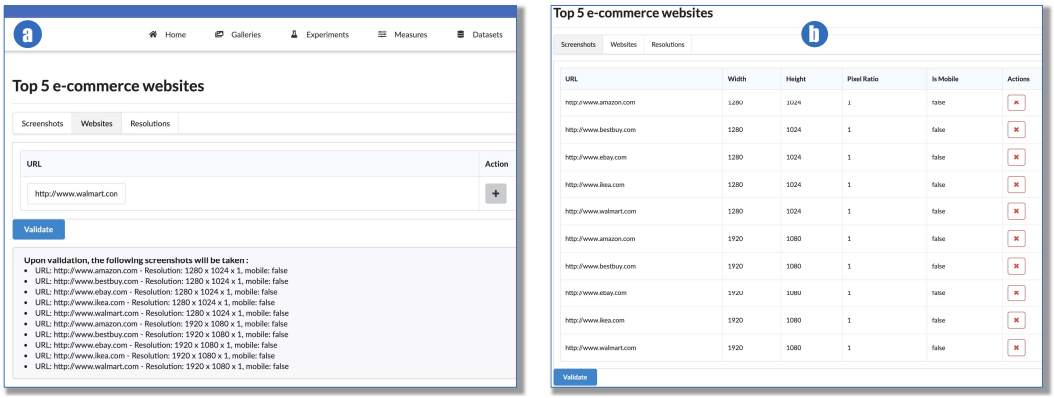

Fig. 12. UiLab: (a) Screenshots form screen, (b) Screenshots overview screen.

6.2 Design and Deployment of Experiments

UiLab implements a workflow that enables experiments to design their experiments according to a generic template and deploy them easily. Experiments deployed are readily available for users and data collection does not require long design and development phases. In addition to that, experimental setups can be saved for further reuse in experiments of similar configuration. The experimenter has to provide all the details of the experiment, namely the title, the description, and the gallery of the experiment, the different sections of the experiment, the optional information related to surveys including sections and questions and finally the variables to be measured during the experiment (Figures 13a to 13g). Upon validation, the experiment data are sent to the services for creation by the mean of the API gateway. After creation by the back-end services, the experiment is deployed and ready to be shared with participants. A link to each experiment is provided to the experimenter on the experiments summary page (Figure 13h).

6.3 Participation to Deployed Experiments

UiLab allows for users to participate to experiments that have been designed and deployed by experimenters. Aside of the web application available for experimenters, there is an application dedicated to participants that allows them to take part in these experiments. Each deployed experiment has its own URL and users only need to know the URL to participate to the specific experiment.

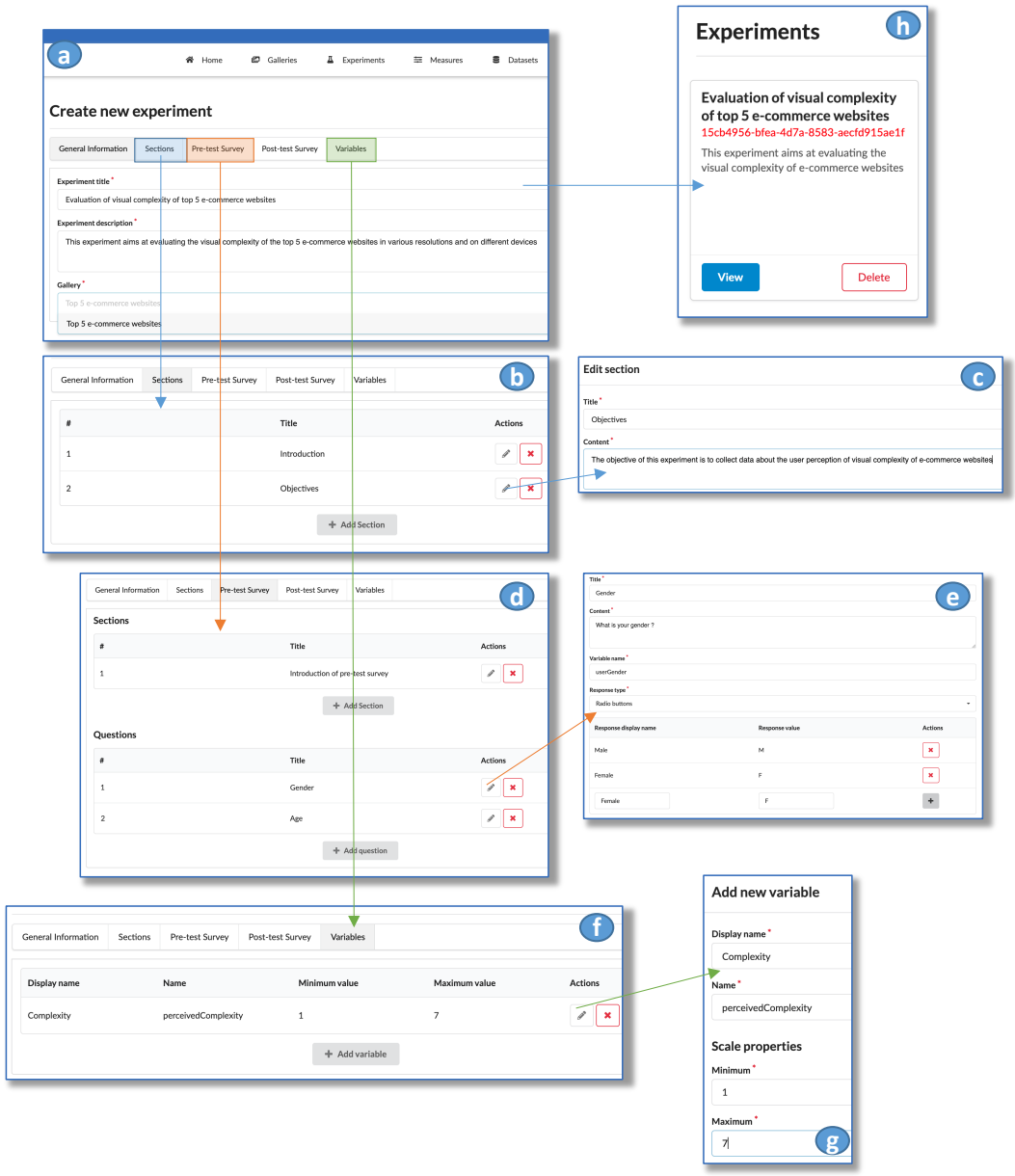

Fig. 13. UiLab: Experiment creation process.

When visiting the URL of an experiment, participants are provided with the introductory sections as specified by experimenter at design time (Figures 14a and b). Then, participants will have the opportunity to answer the pre-test survey if the experimenter specified any question during the conception of the experiment (Figures 14c to e). After answering the questions of the pre-test survey, participants will be invited to take part in the experiment by evaluating the screenshots of the gallery specified by the experimenter (Figure 14f). Finally, once all screenshots have been evaluated, participants are invited to answer questions of the post-test survey (if a post-test survey was defined at design time).

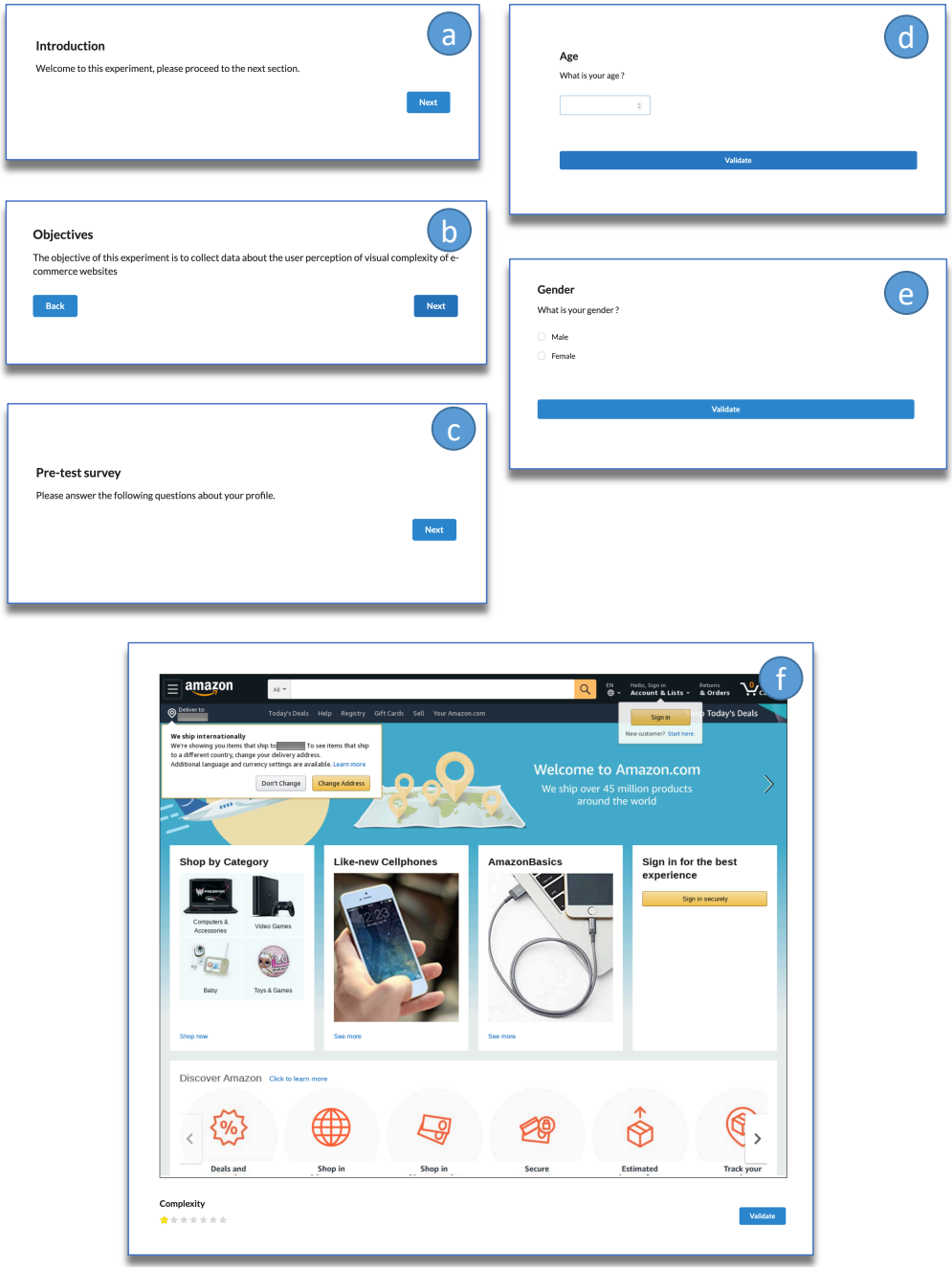

Fig. 14. UiLab- Experiment participation process.

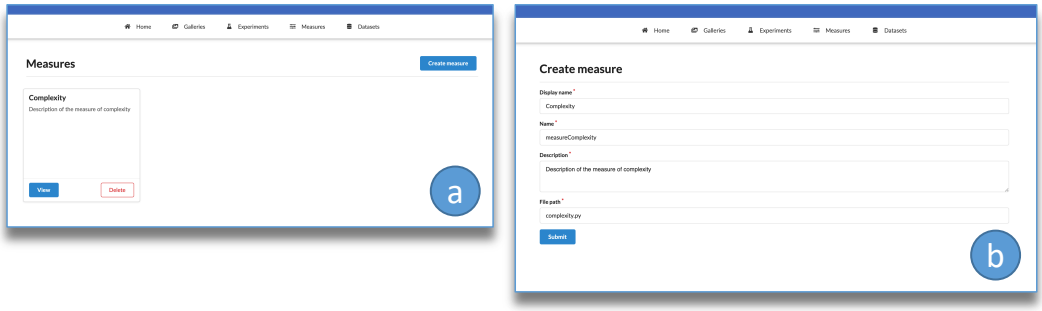

Fig. 15. UiLab: Measure creation process.

6.4 Incremental Development of New Measures

UiLab allows users to define measures that can be computed over GUIs. Measures such as balance or equilibrium [38] require intermediate computations (*e.g.* the segmentation of a UI into zones) that are not themselves considered as measures. For this reason, UiLab defines the concept of *computable* which is broader than the concept of measure (see section 4.2). Users can create computables in UiLab by providing the source code and dependencies file (Figures 15a and 15b). These measures must implement a uniform interface (such as measures in AIM [41]).

Once created, a computable can be associated to a workflow for later computation on galleries of screenshots. However, computables are not directly added to workflows. Instead, UiLab uses the concept of *computable instance* (see section 4.2 for the definition). Computable instances are organized in a workflow by the intermediate of a dependency graph (also see section 4.2), which is a Directed Acyclic Graph (DAG) indicating the dependencies between the computable instances inside a workflow.

When creating a computable instance, the user has to specify the underlying computable, the name of the output (under which it should appear in a data set), and the list of dependencies with, for each dependency, the related parameter name in the source code of the computation instance being created. This parameter mapping is required due to the fact that several computables instances may have a particular computable instance as common dependency, while expecting the output of that dependency under different parameter names.

6.5 Computation of Workflows on Galleries

The computation of measures on GUIs in UiLab is performed through the creation of runs (see section 4.2). When creating a run, the user has to specify the underlying workflow and gallery the run is based on. The backend of UiLab will then create a computation entity for each pair of computable instance and screenshot that will contain the computation result.

The lifecycle of each computation can be described by a finite state machine such as illustrated in figure 16 (A). Figure 16 also illustrates the flow between computations as their state changes. The different states are as follows:

Idle : When a run is first created, all the associated computations are set in the idle state. After creating all the required computations, the backend set all of computations without dependencies to the check state.

Check : When a computation enters the check state, the UiLab backend verifies that all of its dependencies are in the completed state, if any. If so, the backend put the computation in the launchable state, otherwise it put the computation back in the idle state.

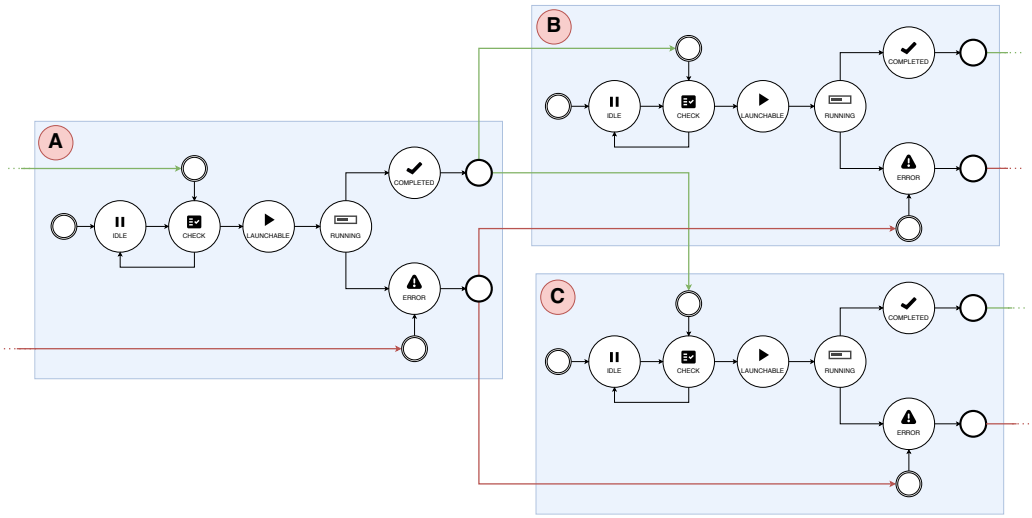

Fig. 16. UiLab computations: Execution flow.

Launchable : When a computation enters the LAUNCHABLE state, the backend aggregates the result of the computation of its dependencies in a file that is made available online on AWS S3. The backend then triggers the creation of an AWS ECS task. This task will run a Docker container that will gather all the required information for the computation (i.e the parameters previously aggregated, the considered materials of the considered screenshot and computable). When the computation finishes or if an error occurs, the backend will put the computation in either COMPLETED or ERROR state respectively, along with the result of the computation if applicable.

Completed : When a computation is marked as COMPLETED, the backend of UiLab will put all of the computations depending on it in the CHECK state. This is illustrated by the green arrows of Figure 16.

Error : When a computation is marked as ERROR, the backend of UiLab will put all of the computations depending on it in the ERROR state, leading to a chain reaction of error propagation across computations. This is illustrated by the red arrows of Figure 16.

6.6 Datasets as Unions of Data Sources

UiLab's datasets must be seen as the union of different data sources with a pivot entity (*e.g.*, the screenshot being the pivot entity between user-related data and measures-related data). Each data source (*e.g.*, measures computed on screenshots, user-related data gathered through experiments) can be downloaded separately in UiLab. The purpose of the application is not to provide the experimenter a full-fledged, ready-to-use dataset. The goal of the application is rather to provide experimenters with a quick and easy way to collect the different data sources, allowing them to compose their dataset easily. Moreover, with the development of new modules inside of the application, the number of data sources will potentially increase. This reinforces the need for experimenters to be able to easily gather the different data sources they are interested in in order to compose a dataset tailored to their needs.

7 IMPACT ON EXPERIMENT DESIGN TIME: EVALUATION OF THE APPLICATION

In order to evaluate the impact of the UiLab application on the time taken to build an experiment, we performed an evaluation consisting of building an experiment, both manually and with UiLab. The experiment designed has to meet several requirements:

- The experiment should evaluate 50 e-commerce websites.
- The screenshots of the selected websites should be taken with various resolutions due to the variety of devices participants can use to take part in the experiment.
- The screenshots displayed to a given participant should have the largest width smaller than the participant device's width.
- The variables evaluated by the participants regarding the displayed screenshots are the visual appeal and complexity of the GUIs represented in the screenshots. They should be evaluated on a 7-items Likert scale.
- For each participant, screenshots should be displayed in a random order and each screenshot should be evaluated only once by each participant.
- The experiment should allow to collect socio-demographical information about the user.
- The experiment process should also include the computation of measures on the captured screenshots and should gather them in a separate dataset.

In order to ensure the feasibility of the manual experiment design, we limited the number of captured resolutions to 3, the number of computer measures to 3 and the socio-demographical information about users collected through the experiment to age and gender.

The goal of these requirements was to ensure that the experiments built would allow to collect the same kind of data with the same kind of stimuli. The performance indicator through this evaluation is the "time to deployment", *i.e.* the delay between the beginning of the experiment inception phase and the end of the deployment phase as well as the time to compute measures on collected screenshots. The data collection process was not relevant for the evaluation of the UiLab application as the purpose is to design and deploy experiments faster and more easily. For this reason, no data were collected during this evaluation.

7.1 Methodology

We asked two developers to develop independently an application implementing such kind of experimental setup as described by the requirements hereabove. The applications could be developed by using any technological stack. We divided the development process in several steps and we asked each developer to record the time he/she took to complete the step. We then asked the developers to build the same experimental setup by using the application UiLab and again to measure the time taken to complete each step. The experiment design process was decomposed into several steps:

Web sites selection & screenshot capture: this step concerns the selection of the web sites to be evaluated in the context of the experiment and the capture of the screenshots of these websites with different resolutions. The capture process can be manual or automated through a script.

Experiment design: this step is related to the conception of the experimental setup and the development of the application that will implement this setup.

Experiment deployment: this concerns the deployment of the application implementing the experimental setup designed previously. The deployment is required to make it available to participants.

Measures computation: this step concerns the computation of measures, manually by users or automatically by a platform.

	Experiment 1		Experiment 2	
	Manual	UiLab	Manual	UiLab
Screenshots capture	8h	45min	6h	45min
Experiment design	80h	45min	100h	1h
Experiment deployment	8h	0	4h	0
Measures computation	6h	30min	8h	45min
Total	102h	2h	118h	2h30

Table 3. Recorded time for each configuration and step of the UiLab evaluation process.

We integrated the measure computation as a step in the design process because this is a feature of the application UiLab that needs to be tested. This part about measure computation is rather related to the dataset construction process and is not related to the design of experimental setup strictly speaking.

7.2 Results

The recorded times for each step and for each configuration (manual and with UiLab) are displayed in Table 3 and illustrated in Figure 17.

We can see that the time required to design and build a complete experimental setup drastically decreases with the use of the application UiLab. The main bottleneck in the manual process of experiment design is the development of the application hosting the experiment. This process involves a lot of different tasks such as front-end application development, back-end server development, database design and deployment, etc. All these tasks may require different technologies which use may be tedious if one are not familiar with them. Measure computation is also time-consuming as computing measures with the help of tools such as AIM [41] does not allow processing multiple screenshots simultaneously (Table 2). The application UiLab abstracts all these tasks by providing a high-level, template-based, service for experiment building and a computation service for measures computation. Experimenters can easily interact with these engines through the application GUI. This explains the decrease in design and conception time in both cases. We also observe the absence of deployment time with the use of UiLab as experiments are automatically deployed.

The applications built manually had some differences in their internal working but also in the way information was presented to the user. This may introduce variability in the results obtained for a data collection process using these applications. The template engine of UiLab allows standardizing the presentation layer of experimental setups which reduces potential variability in data due to this factor.

8 CONCLUSION

In this document, we presented UiLab, our contribution to the domain of experimental studies on GUI visual design. UiLab is a service-oriented web application that aims at facilitating the collection of data and the construction of datasets related to the study of GUI visual design. We discussed the reproducibility issues encountered in the domain of Human-Computer Interactions and more specifically in the field of GUIs evaluation. We listed the different tools available allowing users to compute measures on UIs or to conduct experiments related to GUIs evaluation. We discussed their functionalities and we summarized the comparison of these tools into a comparative table. We provided an in-depth description of the model underlying UiLab with a definition of all relevant concepts and their relationships. We then provided an overview of UiLab by describing each of the principal functionalities illustrated with captures of the related application screens. Finally, we presented the evaluation of the application by comparing the conception time of several

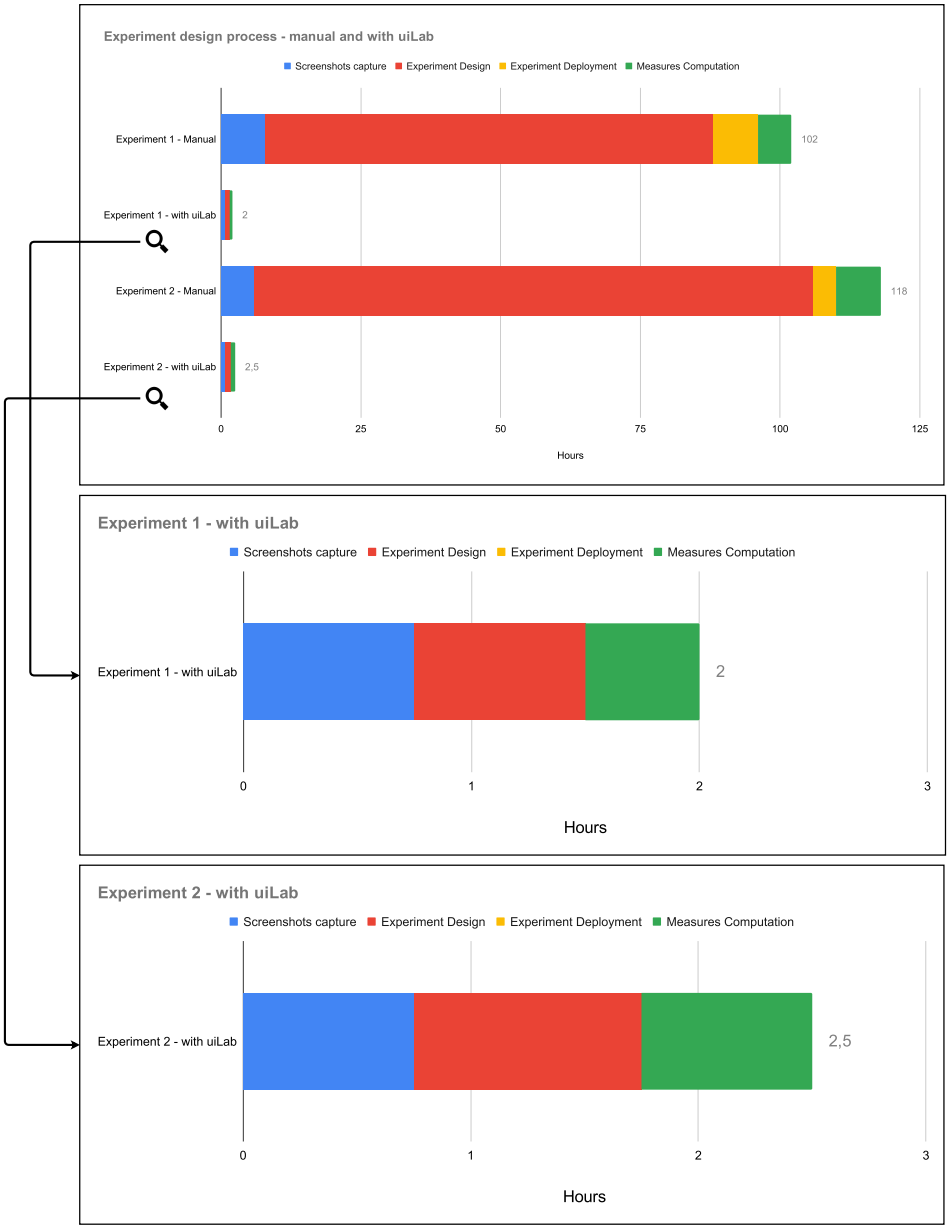

Fig. 17. Evaluation of UiLab.

experiments built manually and by using UiLab. We noticed a important decrease in design and development time of experiments, which is the main bottleneck of such process.

We hope that UiLab will allow researchers to build datasets more easily and will contribute in this way to improve the reproducibility of experimental results in the field of GUI visual design.

9 FUTURE WORK

Although UiLab is complete enough to meet the requirements elicited earlier in this document, there are situations that the application is not currently able to manage. For example, it is currently

not possible to evaluate websites or web applications through real interactions between users and the web sites or to measure real-time usage metrics to evaluate usability. We are currently limited to the evaluation of static screenshots of websites and web applications. However, the service-based architecture of UiLab enables anybody to contribute to the development of the application. In this way, we hope the platform will continue to develop with each person developing new services or modifying the existing services to meet their particular needs.

REFERENCES

[1] Alain Abran, Adel Khelifi, Witold Suryn, and Ahmed Seffah. 2003. Usability Meanings and Interpretations in ISO Standards. *Software Quality Journal* 11, 4 (Nov. 2003), 325–338. https://doi.org/10.1023/A:1025869312943

[2] ACM. 2018. Artifact Review and Badging. (April 2018). https://www.acm.org/publications/policies/artifact-review-badging

[3] Enis Afgan, Dannon Baker, Bérénice Batut, Marius Van Den Beek, Dave Bouvier, Martin Ech, John M. Chilton, Dave Clements, Nate Coraor, Björn A. Grüning, Aysam Guerler, Jennifer Lynne Jackson, Saskia Hiltemann, Vahid Jalili, Helena Rasche, Nicola Soranzo, Jeremy Goecks, James Taylor, Anton Nekrutenko, and Daniel Blankenberg. 2018. The Galaxy platform for accessible, reproducible and collaborative biomedical analyses: 2018 update. *Nucleic Acids Research* 46, W1 (2 7 2018), W537–W544. https://doi.org/10.1093/nar/gky379

[4] Khalid Alemerien and Magel Kenneth. 2015. SLC: a visual cohesion metric to predict the usability of graphical user interfaces. 1526–1533. https://doi.org/10.1145/2695664.2695791

[5] Khalid Alemerien and Kenneth Magel. 2014. GUIEvaluator: A Metric-tool for Evaluating the Complexity of Graphical User Interfaces. In *SEKE*.

[6] Farah Alsudani and Matthew Casey. 2009. The Effect of Aesthetics on Web Credibility. In *Proceedings of the 23rd British HCI Group Annual Conference on People and Computers: Celebrating People and Technology (BCS-HCI '09)*. British Computer Society, Swinton, UK, 512–519. http://dl.acm.org/citation.cfm?id=1671011.1671077

[7] Raluca Antonie. 2009. Concepts of Research Methods and Statistics Used in Program Evaluation. *Transylvanian Review of Administrative Sciences* 26E (06 2009).

[8] Maxim Bakaev, Sebastian Heil, Vladimir Khvorostov, and Martin Gaedke. 2019. Auto-Extraction and Integration of Metrics for Web User Interfaces. *Journal of Web Engineering (JWE)* 17 (03 2019), 561–590. https://doi.org/10.13052/jwe1540-9589.17676

[9] Pablo Becker, Philip Lew, and Luis Olsina. 2012. Specifying Process Views for a Measurement, Evaluation, and Improvement Strategy. *Advanced Software Engineering* 2012 (2012), 949746:1–949746:28. https://doi.org/10.1155/2012/949746

[10] Narjes Bessghaier, Makram Soui, Christophe Kolski, and Mabrouka Chouchane. 2021. On the Detection of Structural Aesthetic Defects of Android Mobile User Interfaces with a Metrics-Based Tool. *ACM Trans. Interact. Intell. Syst.* 11, 1, Article 3 (March 2021), 27 pages. https://doi.org/10.1145/3410468

[11] Josette Bettany-Saltikov. 2010. Learning how to undertake a systematic review: part 1. *Nursing standard (Royal College of Nursing (Great Britain) : 1987)* 24 (08 2010), 47–55; quiz 56. https://doi.org/10.7748/ns2010.08.24.50.47.c7939

[12] Josette Bettany-Saltikov. 2010. Learning how to undertake a systematic review: Part 2. *Nursing standard (Royal College of Nursing (Great Britain) : 1987)* 24 (08 2010), 47–56; quiz 58, 60. https://doi.org/10.7748/ns2010.08.24.51.47.c7943

[13] Elodie Bouzekri. 2018. Model-Based Approach to Design and Develop Usable and Dependable Recommender Systems. In *Proceedings of the ACM SIGCHI Symposium on Engineering Interactive Computing Systems, EICS 2018, Paris, France, June 19-22, 2018*. ACM, 17:1–17:7. https://doi.org/10.1145/3220134.3220147

[14] Murilo C. Camargo, Rodolfo M. Barros, and Vanessa T. O. Barros. 2018. Visual Design Checklist for Graphical User Interface (GUI) Evaluation. In *Proceedings of the 33rd Annual ACM Symposium on Applied Computing (SAC '18)*. Association for Computing Machinery, New York, NY, USA, 670–672. https://doi.org/10.1145/3167132.3167391

[15] Jon F. Claerbout and Martin Karrenbach. 2005. *Electronic documents give reproducible research a new meaning*. 601–604. https://doi.org/10.1190/1.1822162 arXiv:https://library.seg.org/doi/pdf/10.1190/1.1822162

[16] Sandy Claes and Andrew Vande Moere. 2017. Replicating an In-The-Wild Study One Year Later: Comparing Prototypes with Different Material Dimensions. In *Proceedings of the 2017 Conference on Designing Interactive Systems (DIS '17)*. Association for Computing Machinery, New York, NY, USA, 1321–1325. https://doi.org/10.1145/3064663.3064725

[17] Andy Cockburn, Carl Gutwin, and Alan Dix. 2018. HARK No More: On the Preregistration of CHI Experiments. In *Proceedings of the ACM International Conference on Human Factors in Computing Systems (CHI '18)*. Association for Computing Machinery, New York, NY, USA, Article Paper 141, 12 pages. https://doi.org/10.1145/3173574.3173715

[18] D.A Dondis. 1973. *A Primer of Visual Literacy*. The MIT Press, MA, USA.

[19] Sophie Dupuy-Chessa, Yann Laurillau, and Eric Céret. 2016. Considering Aesthetics and Usability Temporalities in a Model Based Development Process. In *Actes de La 28ième Conference Francophone Sur lâĂŹInteraction Homme-Machine*

(IHM '16). Association for Computing Machinery, New York, NY, USA, 25–35. https://doi.org/10.1145/3004107.3004122

[20] Florian Echtler and Maximilian Häuundefinedler. 2018. Open Source, Open Science, and the Replication Crisis in HCI. In *Extended Abstracts of the 2018 CHI Conference on Human Factors in Computing Systems (CHI EA '18)*. Association for Computing Machinery, New York, NY, USA, Article alt02, 8 pages. https://doi.org/10.1145/3170427.3188395

[21] Patricia Farrugia, Bradley A. Petrisor, Forough Farrokhyar, and Mohit Bhandari. 2010. Practical tips for surgical research: Research questions, hypotheses and objectives. *Canadian Journal of Surgery* 53, 4, Article PMC2912019 (Aug. 2010), 278–281 pages. https://canjsurg.ca/wp-content/uploads/2013/12/53-4-278.pdf

[22] Omar S. Gómez, Natalia Juristo, and Sira Vegas. 2010. Replications Types in Experimental Disciplines. In *Proceedings of the 2010 ACM-IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM '10)*. Association for Computing Machinery, New York, NY, USA, Article Article 3, 10 pages. https://doi.org/10.1145/1852786.1852790

[23] Salvador González, Francisco Montero, and Pascual González. 2012. BaLOReS: A Suite of Principles and Metrics for Graphical User Interface Evaluation. In *Proceedings of the 13th International Conference on Interacción Persona-Ordenador (Interaccion '12)*. Association for Computing Machinery, New York, NY, USA, Article 9, 2 pages. https://doi.org/10.1145/2379636.2379645

[24] Steven N. Goodman, Daniele Fanelli, and John P.A. Ioannidis. 2016. What does research reproducibility mean? *Science Translational Medicine* 8, 341 (June 2016), 341ps12. https://doi.org/10.1126/scitranslmed.aaf5027

[25] Christian Greiffenhagen and Stuart Reeves. 2013. Is Replication Important for HCI?. In *Proceedings of the ACM CHI '13 Workshop on the Replication of HCI Research* (April 27-28, 2013) *(CEUR Workshop Proceedings)*, Max L. Wilson, Ed H. Chi, David Coyle, and Paul Resnick (Eds.), Vol. 976. CEUR-WS.org, 8–13. http://ceur-ws.org/Vol-976

[26] Edward Hartono and Clyde W. Holsapple. 2019. Website Visual Design Qualities: A Threefold Framework. *ACM Trans. Manage. Inf. Syst.* 10, 1, Article Article 1 (April 2019), 21 pages. https://doi.org/10.1145/3309708

[27] Kasper Hornbæk. 2015. We Must Be More Wrong in HCI Research. *Interactions* 22, 6 (Oct. 2015), 20–21. https://doi.org/10.1145/2833093

[28] Kasper Hornbæk, Søren S. Sander, Javier Andrés Bargas-Avila, and Jakob Grue Simonsen. 2014. Is Once Enough? On the Extent and Content of Replications in Human-Computer Interaction. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '14)*. Association for Computing Machinery, New York, NY, USA, 3523âĂŞ3532. https://doi.org/10.1145/2556288.2557004

[29] Melody Y. Ivory and Marti A. Hearst. 2002. Statistical Profiles of Highly-rated Web Sites. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '02)*. ACM, New York, NY, USA, 367–374. https://doi.org/10.1145/503376.503442

[30] Vince Kellen. [n. d.]. *Business Performance Measurement – At the Crossroads of Strategy, Decision-Making, Learning and Information Visualization*. Technical Report. DePaul University.

[31] Rensis Likert. 1932. A technique for the measurement of attitudes. *Archives of Psychology* 22, 140 (1932), 55–. http://psycnet.apa.org/record/1933-01885-001

[32] Gitte Lindgaard, Gary Fernandes, Cathy Dudek, and Judith M. Brown. 2006. Attention web designers: You have 50 milliseconds to make a good first impression! Behaviour and Information Technology, 25(2), 115-126. *Behaviour IT* 25 (03 2006), 115–126. https://doi.org/10.1080/01449290500330448

[33] Adriana Lopes, Anna Beatriz Marques, Simone Diniz Junqueira Barbosa, and Tayana Conte. 2015. Evaluating HCI Design with Interaction Modeling and Mockups - A Case Study. In *ICEIS 2015 - Proceedings of the 17th International Conference on Enterprise Information Systems, Volume 3, Barcelona, Spain, 27-30 April, 2015*, Slimane Hammoudi, Leszek A. Maciaszek, and Ernest Teniente (Eds.). SciTePress, 79–87. https://doi.org/10.5220/0005374200790087

[34] Salvador González López, Francisco Montero Simarro, and Pascual González López. 2013. *BaLOReS: A Framework for Quantitative User Interface Evaluation*. Springer London, London, 127–143. https://doi.org/10.1007/978-1-4471-5445-7_10

[35] Nadine Mandran and Sophie Dupuy-Chessa. 2018. Supporting experimental methods in information system research. In *Proceedings of 12th International Conference on Research Challenges in Information Science (RCIS '18)*. 1–12. https://doi.org/10.1109/RCIS.2018.8406654

[36] Anna Beatriz Marques, Simone D. J. Barbosa, and Tayana Conte. 2016. A comparative evaluation of interaction models for the design of interactive systems. In *Proceedings of the 31st Annual ACM Symposium on Applied Computing, Pisa, Italy, April 4-8, 2016*, Sascha Ossowski (Ed.). ACM, 173–180. https://doi.org/10.1145/2851613.2851679

[37] Kevin Mullet and Darrell Sano. 1995. *Designing Visual Interfaces*. Pearson Education, USA.

[38] David Chek Ling Ngo, Lian Seng Teo, and John G. Byrne. 2003. Modelling Interface Aesthetics. *Inf. Sci.* 152, 1 (June 2003), 25–46. https://doi.org/10.1016/S0020-0255(02)00404-8

[39] Raquel Oliveira, Sophie Dupuy-Chessa, and Gaëlle Calvary. 2015. Plasticity of User Interfaces: Formal Verification of Consistency. In *Proceedings of the 7th ACM SIGCHI Symposium on Engineering Interactive Computing Systems (EICS '15)*. Association for Computing Machinery, New York, NY, USA, 260–265. https://doi.org/10.1145/2774225.2775078

[40] International Standard Organization. 2011. ISO/IEC 25010:2011, Systems and software engineering âĂŤ Systems and software Quality Requirements and Evaluation (SQuaRE) âĂŤ System and software quality models. (2011). https://www.iso.org/standard/35733.html

[41] Antti Oulasvirta, Samuli De Pascale, Janin Koch, Thomas Langerak, Jussi Jokinen, Kashyap Todi, Markku Laine, Manoj Kristhombuge, Yuxi Zhu, Aliaksei Miniukovich, Gregorio Palmas, and Tino Weinkauf. 2018. Aalto Interface Metrics (AIM): A Service and Codebase for Computational GUI Evaluation. In *The 31st Annual ACM Symposium on User Interface Software and Technology Adjunct Proceedings (UIST '18 Adjunct)*. ACM, New York, NY, USA, 16–19. https://doi.org/10.1145/3266037.3266087

[42] Eleftherios Papachristos and Nikolaos Avouris. 2011. Are First Impressions about Websites Only Related to Visual Appeal?. In *Human-Computer Interaction – INTERACT 2011*, Pedro Campos, Nicholas Graham, Joaquim Jorge, Nuno Nunes, Philippe Palanque, and Marco Winckler (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 489–496.

[43] Harold Pashler and EricâĂŞJan Wagenmakers. 2012. Editors' Introduction to the Special Section on Replicability in Psychological Science: A Crisis of Confidence? *Perspectives on Psychological Science* 7, 6 (2012), 528–530. https://doi.org/10.1177/1745691612465253 arXiv:https://doi.org/10.1177/1745691612465253 PMID: 26168108.

[44] Prasad Patil, Roger D. Peng, and Jeffrey T. Leek. 2019. A visual tool for defining reproducibility and replicability. *Nature Human Behaviour* 3, 7 (2019), 650–652. https://doi.org/10.1038/s41562-019-0629-z

[45] Lg Pee, James Jiang, and Gary Klein. 2018. Signaling effect of website usability on repurchase intention. *International Journal of Information Management* 39 (04 2018), 228–241. https://doi.org/10.1016/j.ijinfomgt.2017.12.010

[46] Roger D. Peng. 2011. Reproducible Research in Computational Science. *Science* 334, 6060 (2011), 1226–1227. https://doi.org/10.1126/science.1213847 arXiv:https://science.sciencemag.org/content/334/6060/1226.full.pdf

[47] Stefan Pröll, Kristof Meixner, and Andreas Rauber. 2016. Precise Data Identification Services for Long Tail Research Data. (01 2016). https://doi.org/10.6084/M9.FIGSHARE.3847632

[48] Ghulam Jilani Quadri and Paul Rosen. 2019. You Can't Publish Replication Studies (and How to Anyways). (2019). arXiv:cs.HC/1908.08893

[49] Matthias Rauterberg. 1996. How to measure and to quantify usability attributes of man-machine interfaces. In *Proceedings 5th IEEE International Workshop on Robot and Human Communication (ROMAN '96)*. 262–267. https://doi.org/10.1109/ROMAN.1996.568839

[50] Katharina Reinecke and Krzysztof Z. Gajos. 2014. Quantifying Visual Preferences Around the World. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '14)*. ACM, New York, NY, USA, 11–20. https://doi.org/10.1145/2556288.2557052

[51] Andreas Riegler and Clemens Holzmann. 2015. UI-CAT: Calculating User Interface Complexity Metrics for Mobile Applications. In *Proceedings of the 14th International Conference on Mobile and Ubiquitous Multimedia (MUM âĂŹ15)*. Association for Computing Machinery, New York, NY, USA, 390âĂŞ394. https://doi.org/10.1145/2836041.2841214

[52] Suzanne Robertson and James Robertson. 2012. *Mastering the Requirements Process: Getting Requirements Right* (3rd ed.). Addison-Wesley Professional.

[53] Geir Kjetil Sandve, Anton Nekrutenko, James Taylor, and Eivind Hovig. 2013. Ten Simple Rules for Reproducible Computational Research. *PLoS Computational Biology* 9, 10 (2013). http://dblp.uni-trier.de/db/journals/ploscb/ploscb9.html#SandveNTH13

[54] Jonathan W. Schooler. 2014. Metascience could rescue the "replication crisis". *Nature* 515, 9 (Nov. 2014), 9. https://doi.org/10.1038/515009a

[55] Ahmed Seffah, Mohammad Donyaee, Rex Kline, and Harkirat Padda. 2006. Usability measurement and metrics: A consolidated model. *Software Quality Journal* 14 (06 2006), 159–178. https://doi.org/10.1007/s11219-006-7600-8

[56] Andreas Sonderegger and Jürgen Sauer. 2009. The influence of design aesthetics in usability testing: Effects on user performance and perceived usability. *Applied ergonomics* 41 (11 2009), 403–10. https://doi.org/10.1016/j.apergo.2009.09.002

[57] Makram Soui, Mabrouka Chouchane, Ines Gasmi, and Mohamed Wiem Mkaouer. 2017. PLAIN: PLugin for predicting the usAbility of mobile user INterface. In *Proceedings of the 12th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP 2017) - Volume 1: GRAPP, Porto, Portugal, February 27 - March 1, 2017*, Ana Paula Cláudio, Dominique Bechmann, and José Braz (Eds.). SciTePress, 127–136. https://doi.org/10.5220/0006171201270136

[58] Vasan Subramanian. 2019. *Pro MERN Stack: Full Stack Web App Development with Mongo, Express, React, and Node.* Apress. https://www.oreilly.com/library/view/pro-mern-stack/9781484243916/

[59] Toby J. Teorey, Dongqing Yang, and James P. Fry. 1986. A Logical Design Methodology for Relational Databases Using the Extended Entity-Relationship Model. *ACM Comput. Surv.* 18, 2 (June 1986), 197–222. https://doi.org/10.1145/7474.7475

[60] N Tractinsky, A.S Katz, and D Ikar. 2000. What is beautiful is usable. *Interacting with Computers* 13, 2 (2000), 127 – 145. https://doi.org/10.1016/S0953-5438(00)00031-X

[61] Stefan Trausan-Matu and Brahma Dathan. 2016. Perceived aesthetics of user-modifiable layouts: a comparison between an unspecified design and a GUI. In *13th International Conference on Human Computer Interaction, RoCHI 2016, Iasi, Romania, September 8-9, 2016*, Adrian Iftene and Jean Vanderdonckt (Eds.). Matrix Rom, 22–25. http://rochi.utcluj.ro/proceedings/en/articles-RoCHI2016.php

[62] Eric W.K. Tsang and Kai-Man Kwan. 1999. Replication and Theory Development in Organizational Science: A Critical Realist Perspective. *The Academy of Management Review* 24, 4 (October 1999), 759–780. https://doi.org/10.2307/259353

[63] Silvia Uribe, Federico Álvarez, and José Manuel Menéndez. 2017. User's Web Page Aesthetics Opinion: A Matter of Low-Level Image Descriptors Based on MPEG-7. *ACM Trans. Web* 11, 1, Article 5 (March 2017), 25 pages. https://doi.org/10.1145/3019595

[64] Axel van Lamsweerde. 2001. Goal-oriented requirements engineering: a guided tour. In *Proceedings Fifth IEEE International Symposium on Requirements Engineering*. 249–262. https://doi.org/10.1109/ISRE.2001.948567

[65] Axel van Lamsweerde. 2003. Goal-Oriented Requirements Engineering: From System Objectives to UML Models to Precise Software Specifications. In *Proceedings of the 25th International Conference on Software Engineering (ICSE '03)*. IEEE Computer Society, USA, 744âĂŞ745.

[66] Jean Vanderdonckt and Xavier Gillo. 1994. Visual Techniques for Traditional and Multimedia Layouts. In *Proceedings of the ACM Int. Conf. on Advanced Visual Interfaces (AVI '94)*, Maria Francesca Costabile, Tiziana Catarci, Stefano Levialdi, and Giuseppe Santucci (Eds.). ACM, 95–104. https://doi.org/10.1145/192309.192334

[67] Any Whitefield, Frank Wilson, and John Dowell. 1991. A framework for human factors evaluation. *Behaviour & Information Technology* 10, 1 (1991), 65–79. https://doi.org/10.1080/01449299108924272

[68] Max Wilson, Wendy Mackay, Ed Chi, Michael Bernstein, and Jeffrey Nichols. 2012. RepliCHI SIG: From a Panel to a New Submission Venue for Replication. In *CHI '12 Extended Abstracts on Human Factors in Computing Systems (CHI EA '12)*. Association for Computing Machinery, New York, NY, USA, 1185–1188. https://doi.org/10.1145/2212776.2212419

[69] Max L. Wilson, Ed H. Chi, Stuart Reeves, and David Coyle. 2014. RepliCHI: The Workshop II. In *CHI '14 Extended Abstracts on Human Factors in Computing Systems (CHI EA '14)*. Association for Computing Machinery, New York, NY, USA, 33–36. https://doi.org/10.1145/2559206.2559233

[70] Mathieu Zen and Jean Vanderdonckt. 2014. Towards an evaluation of graphical user interfaces aesthetics based on metrics. In *Proc. of IEEE 8th International Conference on Research Challenges in Information Science (RCIS '14)*, Marko Bajec, Martine Collard, and Rébecca Deneckère (Eds.). IEEE, 1–12. https://doi.org/10.1109/RCIS.2014.6861050

[71] Mathieu Zen and Jean Vanderdonckt. 2016. Assessing User Interface Aesthetics Based on the Inter-Subjectivity of Judgment. In *Proceedings of the 30th International BCS Human-Computer Interaction Conference (BCS-HCI '16)*, Shamal Faily, Nan Jiang, Huseyin Dogan, and Jacqui Taylor (Eds.). BCS. http://ewic.bcs.org/content/ConWebDoc/56903

[72] Ping Zhang and Na Li. 2005. The importance of affective quality. *Commun. ACM* 48 (09 2005), 105–108. https://doi.org/10.1145/1081992.1081997

[73] Xianjun Sam Zheng, Ishani Chakraborty, James Jeng-Weei Lin, and Robert Rauschenberger. 2009. Correlating Low-level Image Statistics with Users - Rapid Aesthetic and Affective Judgments of Web Pages. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '09)*. ACM, New York, NY, USA, 1–10. https://doi.org/10.1145/1518701.1518703

A FUNCTIONAL REQUIREMENTS OF UILAB

Id	Name	Description	Dependencies
1	MeetLevelA	The application must meet the level A of replicability framework	2, 8
2	AllowExperimentalSetupCRUDOperations	The application must allow the user to create experimental setups	3, 4, 5, 6, 7
3	AllowSectionCRUDOperations	The application must allow the user to perform CRUD operations on sections	/
4	AllowSurveyCRUDOperations	The application must allow the user to perform CRUD operations on surveys	/
5	AllowQuestionCRUDOperations	The application must allow the user to perform CRUD operations on questions	/
6	AllowResponseChoiceCRUDOperations	The application must allow the user to perform CRUD operations on response choices	/
7	AllowVariableCRUDOperations	The application must allow the user to perform CRUD operations on response choices	/
8	AllowDownloadExperimentMetadata	The application must allow the user to download the meta-data related to an experiment	/

Table 4. Requirements for level A of replicability framework.

Id	Name	Description	Dependencies
9	MeetLevelB	The application must meet the level B of replicability framework	1; 10; 11; 12; 15
10	AllowScreenshotCapture	The application must be able to capture screenshots of different websites and resolutions in an automatic fashion	/
11	AllowScreenshotDownload	The application must be able to download the captured screenshots	/
12	AllowParticipationToExperiment	The application must be allow the participants to take part into deployed experiments	13; 14
13	AllowAnswerToQuestions	The application must be allow the participants to answer to survey questions in the context of an experiment	/
14	AllowScreenshotEvaluation	The application must be allow the participants to evaluate screenshots regarding given variables in the context of an experiment	/
15	AllowParticipantDataCRUDOperations	The application must be allow the experimenter to download participant-related data for a given experiment	16; 17
16	AllowResponseCRUDOperations	The application must be allow the experimenter to download data related to responses provided by participants	/
17	AllowEvaluationCRUDOperations	The application must be allow the experimenter to download data related to evaluation of screenshots	/

Table 5. Requirements for level B of replicability framework.

Id	Name	Description	Dependencies
18	MeetLevelC	The application must meet the level C of replicability framework	9; 19; 20; 21
19	AllowGalleryCRUDOperations	The application must be allow the experimenter to perform CRUD operations on gallery entities	/
20	AllowScreenshotGroupingInGallery	The application must be allow the experimenter to group captured screenshots into galleries	/
21	AllowGalleryMetadataDownload	The application must be allow the experimenter to download the metadata related to a gallery	/

Table 6. Requirements for level C of replicability framework.

Id	Name	Description	Dependencies
22	MeetLevelD	The application must meet the level D of replicability framework	18; 23
23	AllowMeasureComputation	The application must allow the experiment to compute measures on galleries of screenshots	24; 25; 26; 27
24	AllowWorkflowCRUDOperations	The application must allow the experiment to create workflow entities	/
25	ReqAllowComputableCRUDOperations	The application must allow the experiment to create computable entities	/
26	ReqAllowComputableInstanceCRUDOperations	The application must allow the experiment to create computable instance entities	/
27	ReqAllowRunCRUDOperations	The application must allow the experiment to create run entities	/

Table 7. Requirements for level D of replicability framework.

Id	Name	Description	Dependencies
28	MeetLevelE	The application must meet the level E of replicability framework	22; 29; 30
29	ReqProvideDocumentation	The documentation related to the application must be publicly accessible	/
30	ReqProvideSourceCode	The source code of the application must be publicly accessible	/

Table 8. Requirements for level E of replicability framework.

Received February 15th, 2020; revised October 23rd, 2020; accepted October 30th, 2020.