Journal Pre-proof

Predicting interaction effort in web interface widgets

Juan Cruz Gardey, Julián Grigera, Andrés Rodríguez, Gustavo Rossi, Alejandra Garrido

PII: S1071-5819(22)00139-2
DOI: https://doi.org/10.1016/j.ijhcs.2022.102919
Reference: YIJHC 102919

To appear in: *International Journal of Human - Computer Studies*

Received date : 26 March 2021
Revised date : 28 March 2022
Accepted date : 24 August 2022

Please cite this article as: J.C. Gardey, J. Grigera, A. Rodríguez et al., Predicting interaction effort in web interface widgets. *International Journal of Human - Computer Studies* (2022), doi: https://doi.org/10.1016/j.ijhcs.2022.102919.

Revised manuscript (clean version)

Predicting Interaction Effort in Web Interface Widgets

Juan Cruz Gardey[a,b,*], Julián Grigera[a,b,c], Andrés Rodríguez[a], Gustavo Rossi[a,b] and Alejandra Garrido[a,b]

[a]*LIFIA, Fac. de Informática, Univ. Nac. La Plata, La Plata, CP 1900, Argentina*
[b]*CONICET, Argentina*
[c]*CICPBA, Argentina*

ARTICLE INFO

Keywords:
interactivity
user interaction metrics
user experience
web usability
UX refactoring

ABSTRACT

The product of good design should render a tool invisible for a user who is executing a task. Unfortunately, web applications are often far from invisible to users, who struggle with poor design of websites and processes in them. We are particularly interested in web processes that involve form filling, so we have been studying how people interact with web forms. Besides cataloguing user interaction problems that are common in web forms, we have noticed that, in many cases, there is a single form element or widget to blame for a certain interaction problem, because such widget is not the most appropriate one for the required input in that particular context. This unfitness of the widget causes an extra burden to the user, which we call *interaction effort*. In this work we propose measuring the interaction effort of a widget with a unified score based on micro-measures automatically captured from interaction logs. We present the micro-measures that were found relevant to predict the interaction effort in 6 different types of web forms widgets. We describe a large data collection process and prediction models, showing that it is indeed possible to automatically predict a widget interaction effort score by learning from expert human ratings. We consequently believe that the interaction effort could be used as an effective metric to compare small variations in a design in terms of user experience.

1. Introduction

The design of an artifact determines how we interact with it and how much additional effort it requires while helping us complete our goal. Norman says that "good designs fit our needs so well that the design is invisible, serving us without drawing attention to itself" (Norman, 2013). Many authors have studied user interaction on software systems and the extra effort that it requires to interact with software artifacts. For instance, in the context of learning systems, research on cognitive load theory calls "extraneous processing" to a learner's extra cognitive effort which does not support the learning objective but on the contrary, deviates learners from it, as it may be the case with a confusing content layout or poor design (DeLeeuw and Mayer, 2008); in contrast, Janlert and Stolterman define "implicit interaction" as the interaction not requiring user attention (Janlert and Stolterman, 2017).

Most of the studies on web interaction focus on obtaining a measure of the interaction on a website, a whole page, or a complete user session. They do so by analyzing user interface events. For example, with a large experiment, Akers proved that analyzing backtracking events can be as effective as performing user testing in order to measure the usability of a web application (Akers et al., 2012). However, analyzing backtracking events requires manual observation, which is expensive and may be cost-prohibitive for some organizations. Therefore, automating the analysis of user interaction becomes important, as it could present a more affordable option for website evaluation. Along this line, Speicher et al. created a platform that captures different kinds of interaction events automatically while users are performing a task and presents a questionnaire at the end to ask for explicit usability ratings; the collected interactions plus these ratings are used to train models for predicting the usability of a website (Speicher et al., 2014). However, their experiments show that, for an accurate prediction, it is imperative to capture some context information like page structure, user intention, and even display resolution.

*Corresponding author
jcgardey@lifia.info.unlp.edu.ar (J.C. Gardey); julian.grigera@lifia.info.unlp.edu.ar (J. Grigera); arodrig@lifia.info.unlp.edu.ar (A. Rodríguez); gustavo@lifia.info.unlp.edu.ar (G. Rossi); garrido@lifia.info.unlp.edu.ar (A. Garrido)
ORCID(s):

The narrow focus on the instrumental qualities supported by usability measures was challenged when other dimensions of interactive products became relevant, and consequently the multidimensional notion of User Experience (UX) emerged (Hassenzahl and Tractinsky, 2006). ISO defines UX as "user's perceptions and responses that result from the use and/or anticipated use of a system, product or service" (ISO, 2019). For ISO, UX is a consequence of both hedonic aspects (brand image, user' internal state resulting from prior experiences and attitudes) and instrumental aspects (functionality, interactive behavior, user's skills, context of use) ((ISO, 2019), note 2). Several other definitions of UX share this hedonic plus instrumental pattern (Law et al., 2009). Hassenzahl et al (Hassenzahl et al., 2021) have advocated for some hierarchical relationship of hedonic aspects over the instrumental/usability ones, as components of the UX, but that debate exceeds the scope of this work. Here we talk broadly of UX in terms of the ISO definition. To improve the UX, a Human or User-Centered Design (UCD) approach has been recommended (Sharp et al.; ISO, 2019). However, agile methods have become the preferred choice for software development (Hoda et al., 2018), and their organization into short, timeboxed development cycles makes UCD practices hard to accommodate (Brhel et al., 2015; Da Silva et al., 2018).

Our long-term goal is to provide technology that allows attending to UX issues even in short iterations and when resources are insufficient to hire subjects for user testing. Thus, we aim at assisting UX experts approach UX issues incrementally, by comparing design variations in production through controlled experiments that may be run automatically. This can be compared to an A/B testing approach that focuses on UX instead of measuring revenue (Firmenich et al., 2019; Gardey and Garrido, 2020). That is, while manual evaluations continue to be important and should be pursued whenever possible, we believe that automating the diagnosis of some factors affecting UX is essential to boost their importance, especially in small/medium-sized companies with limited resources.

In this work, we are particularly interested in web forms which, besides navigation, demand the most interactions from web users; they may range from simple login forms to complex multi-stepped checkout processes. Forms are worth studying since statistics show that while they are the most common lead generation tool, 81% of people abandon them after beginning to fill them out (WPForms, 2020). Moreover, the experience in studying user interaction during form filling processes made us realize that in many cases a single form widget can be responsible for a certain interaction problem, because it is not the most appropriate for the required input.

The described unfitness of a widget is not only related to the type of data or format the input expects, but also to other aspects that can cause user annoyance or discomfort as well as demand extra time and effort from them. We call it widget *interaction effort* (Grigera et al., 2019). Interaction effort is a score assigned by UX experts based on their subjective analysis of the target interaction. For instance, it may be related either to one or many of the following: a confusing layout or label, its position in relation to the rest, unclear input format, unknown options or constraints, too many options, duplicated content or actions, how far a required date is, etc. That is, while it may be related to hedonic factors (such as aesthetics, perceptions, or comfort), it may also be related to instrumental qualities (e.g., usability measures such as efficiency), both affecting UX (ISO, 2019). In this work, rather than arguing about the precise source of the interaction effort, we are interested in providing a practical instrumentation to measure it.

In general, the interaction effort cannot be discovered by simply checking guidelines statically but by observing user events dynamically and detecting the widget that provokes an awkward interaction. We propose predicting the interaction effort that a widget demands from a user with a unified score that is based on its usage. To reach such score, we propose capturing micro-measures while users interact with each type of widget. The micro-measures were found relevant by UX experts during a preliminary experiment. Having this unified score could allow comparing the effort demanded by different types of widgets for the same input. This is particularly relevant when different variations of a design are proposed, for example in the context of A/B testing (Speicher et al., 2014; Gardey and Garrido, 2020) or in the context of UX refactoring (Gardey et al., 2020). Similar to the concept of code refactoring, UX refactoring applies small transformations to the user interface (UI) that preserve functionality, but in this case the purpose is to improve UX rather than internal code quality (Gardey et al., 2020; Garrido et al., 2011).

In a previous work we proposed the interaction effort to rate widgets for text inputs and selects (i.e., drop down lists) (Grigera et al., 2019) where expert raters manually rated the interaction effort for different instances of both kinds of widgets in several web forms. Positive results were obtained in predicting the score from feeding the automatically captured micro-measures into a decision tree. This article aims at extending that work in several aspects: first, the studied widgets have been extended to a broader set that includes links, radio buttons, date selects and date pickers. Secondly, micro-measures have been extended so as to include interaction data from the surroundings of a widget, to account for some information about the context of the widget in the page. Thirdly, tools were created to collect the data: a web extension for event capture and micro-measure recording, and a web application for manual rating which shows

video capture of each user test session along with a rating form. Moreover, a larger data collection process has been carried out which enabled wider variation in the data to train the models. The trained models are decision tree regressors which show that it is possible to predict the interaction effort from the captured micro-measures. Since decision trees are white-box models (i.e., it is possible to interpret what the model learns), an analysis was also conducted in order to understand the weight that each micro-measure bears in the decision process.

Summarizing, the main contribution of this article is the definition and the analysis of the micro-measures that serve to predict the interaction effort of individual widgets using decision trees. For this purpose, we conducted a large data collection process to capture user interactions and UX experts' manual interaction ratings. Having a prediction model for individual widgets could be a useful addition to the UX practitioner's toolkit. By incorporating the effort prediction model into the web application, which only requires user interaction events to work, website owners could get effort ratings for all interactive widgets. This could serve different purposes like pinpointing a roadblock in a checkout process, or comparing different alternatives in an A/B testing context. Thus, our approach does not pretend to replace UX evaluation methods; instead, we believe that the interaction effort metric offers a factor within the measure of UX of a design, which can be computed automatically and used to compare designs.

The rest of the paper is organized as follows: Section 2 provides some background and related work on interactivity, cognitive load and automatic UX evaluation methods. Section 3 presents the motivation behind the focus on individual widgets and provides a description of how the interaction effort could be applied. Section 4 describes the rationale behind micro-measures and their use for different types of web form elements. Section 5 describes the process of data collection and tools built for that purpose. Section 6 presents the models created for learning and predicting the interaction effort and analysis of results. Section 7 discusses the limitations of this approach and finally, Section 8 provides conclusions and future work.

2. Background & Related Work

2.1. Interactivity and Cognitive Load

Interaction in action – "interactivity" after Janlert and Stolterman (2017) – can fluctuate from a moment to another during use and its measurement is of interest to improve the UX of a software system. One way to measure interactivity is in terms of the time spent interacting (Janlert and Stolterman, 2017); thus, the time spent in the interaction with a user interface (UI) component includes the user actions explicitly directed to the component, as well as others that are not completely directed to it and that can be placed under the umbrella of "implicit interaction". This umbrella would include the set of minimal interactions that occur unconsciously and automatically in the context of using a UI to achieve a goal (e.g., movements of the cursor around a widget, scrolling a screen, entering and leaving a widget, etc.) (Atterer et al., 2006).

Those patterns of implicit interaction can be collected by monitoring the actual usage of a system in real world environments and evidence shows that said behavior can reflect to some extent the mental effort and cognitive load (Gütl et al., 2005; Chen et al., 2012). In fact, Human-Computer Interaction (HCI) research is intertwined with studies about the Cognitive Load (CL) due to a limited working memory capacity and a vast long-term memory capacity (Baddeley, 1976).

Cognitive Load theories distinguish three types of loads in working memory: intrinsic (defined by the complexity of information that is to be learned); extraneous (caused by an inappropriate presentation of the material or by requiring to perform activities deemed irrelevant, such as having to integrate information from spatially separate sources of information); and germane (results from active schema construction processes) (DeLeeuw and Mayer, 2008). The CL induced by using a software tool can be modeled as a specific component of extraneous CL (Hollender et al., 2010). The amount of extraneous load due to software use is influenced by the complexity of the software, that is, a suboptimal software design according to traditional usability goals. User actions during the interaction constitute observable behaviors that can be considered CL indicators, with evidence for speech, interactive gesture, digital pen input and mouse interactivity (Chen et al., 2012, 2016).

Taken together, the physical and mental efforts users make in order to achieve their goals through interaction with a UI can be linked to some "cost of interaction" with that UI (Budiu, 2013). In our proposal, we call it *interaction effort* (Grigera et al., 2019), and it is inspired by the works on interactivity and extraneous CL, which provide an interpretation of what we want to measure, although we target individual UI widgets.

2.2. Automated UX Evaluation

Incorporating some degree of automation in UX evaluation and repair is deemed crucial to make these practices mainstream in software development, especially for agile teams working in short development cycles (Da Silva et al., 2018; Firmenich et al., 2019). Bakaev et al. (2017) classify automatic evaluation methods in three classes: *metric-based* methods, which help UX experts during inspection methods on static web pages, *interaction-based* methods, which log real user interaction during user tests, and *model-based* methods, in which users and their interaction are simulated in order to create and train models. The method in this paper can be considered a combination between an interaction-based –for it involves logging user interaction events– and a model-based, since we use the interaction data to build models able to predict user behavior.

Metric-based evaluation methods are mainly concerned with helping experts check guidelines (Bouzit et al., 2016) and assessing UX factors that can be analyzed from a static perspective of the website, such as aesthetic appearance and perceived visual complexity. Examples of this method are the works of Dingli & Mifsud (2011) and Oulasvirta et al. (2018). Dingli & Mifsud proposed USEFul, a web usability evaluation framework that focuses on helping UX experts with inspection methods, especially checking guidelines in html source code. Oulasvirta et al. have developed AIM, an online service that may evaluate an existing UI using several metrics on factors like symmetry, colorfulness and visual clutter.

Regarding interaction-based evaluation methods, there are several tools that capture and analyze event logs during remote user testing and may help identifying UX issues through practical visualizations, for instance by using timelines (Burzacca and Paternò, 2013; Paternò et al., 2017) or usage graphs (de Santana and Baranauskas, 2015). In our previous work, we developed a tool called "USF" which, by analyzing interaction events, automatically detects specific UX problems (Grigera et al., 2017a).

There are several studies specifically analyzing mouse events that relate mouse movement with visual attention (Arroyo et al., 2006), eye tracking and task conditions (Navalpakkam and Churchill, 2012), and survey response difficulty (Horwitz et al., 2017). However, approaches in related work, while providing insights on general page design, fail to provide them on localized interaction problems, which is our objective. As described in Section 4, our approach uses a combination of measures on several interaction events, with the intention of comparing user interaction effort of specific widgets.

In the category of model-based evaluation methods, different models have emerged to predict user interaction. Well-known models such as KLM (Card et al., 1980) and Fitts's Law (Fitts, 1954) have been widely used in user interaction improvement. However, they are focused on capturing one dimension of user interaction in isolation, making prediction difficult in realistic interaction tasks (Li et al., 2018). The introduction of artificial intelligence in HCI has enabled major progress in the automation of UX evaluation. Machine learning models are able to find complex patterns in the data, which would not be easily detectable by means of analytical methods. These models have recently been applied to predict the perceived visual complexity of a webpage by learning from features obtained through the static analysis of the target website (e.g., the amount of UI elements of a specific type or a screenshot of the rendered UI) (Oulasvirta et al., 2018; Dou et al., 2019; Michailidou et al., 2021).

Regarding the user interaction —our main focus in this work– there are proposals that model the user performance in menu selections (Bailly et al., 2014; Li et al., 2018). While Bailly et al. proposed a mathematical model for selection time in linear menus, Li et al. developed a deep recurrent neural network to predict the performance in a sequence of tasks. Both approaches share the goal of predicting user interaction in a target interface without testing it with real users. Moreover, they are limited to a specific interaction task, i.e. the option selection in a vertical menu. However, there are aspects of the user performance depending on the context of use which cannot be included by these models. Another work closer to our proposal is that of Speicher et al. (2014), which also relies on user interaction events in order to make predictions; however, the metrics that they capture are highly coupled to the target page structure, so it becomes necessary to develop a different model for each websites group that share a common layout. In our work, the analysis at the level of UI widgets provides independence from a particular web application family, because these widgets are native components used across different websites.

3. Widget-centered Analysis

In this section we first motivate the relevance of focusing on individual widgets and later describe our approach. In our previous work, as mentioned in the Introduction, we have studied user interaction within small portions of web pages, and proposed transformations to these small portions to improve external qualities of a web application while

preserving functionality. That is, we have proposed the application of the refactoring concept, traditionally focused on internal code qualities (Fowler et al., 1999), to the improvement of usability (Garrido et al., 2011), accessibility (Garrido et al., 2014) and UX (Gardey et al., 2020). UX refactorings, in line with the definition of UX previously introduced, include usability and accessibility refactorings (transformations to improve instrumental aspects) and also account for improvement of hedonic aspects (Gardey et al., 2020). As we mentioned earlier, UX refactorings should preserve functionality, allowing users to perform the same operations on the application before and after they are applied.

Having small transformations focused on a single or a few widgets allows for more manageable and safer changes, that may be incrementally introduced. The additional benefit of having concrete transformations cataloged as UX refactorings is that each one represents a solution to a particular UX issue, which, in the refactoring jargon, is called a "smell" (Grigera et al., 2017a). An example of a UX smell is *Unformatted Input*, which signals that a free text input is used where the user is expected to fill data in a specific format. Figure 1 shows an example of this smell on the birthdate field in a form. As it is, this field can be error-prone since the required date format is unknown to users.

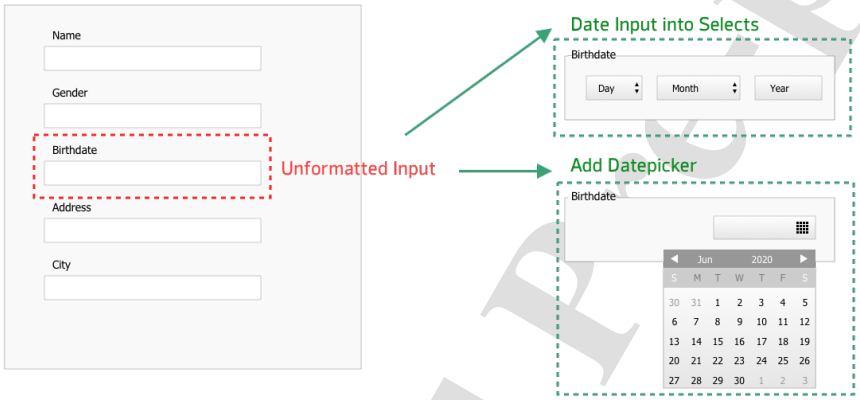

Figure 1: The figure shows an example of a web form with the UX smell Unformatted Input on the "birthdate" field. This smell can be solved by two alternative refactorings: "Date Input into Selects" or "Add Datepicker".

Figure 1 also shows that there are two possible UX refactorings that may solve the aforementioned smell: *Add Datepicker*, which augments the text input with a calendar to select a date, and *Date Input into Selects*, that replaces the input with three select boxes to choose day, month and year. As we can see from this example, the changes applied by a UX refactoring go from augmenting the interaction of an existing widget to replacing it with another widget that serves the same purpose. In this case, users can enter a date on both interfaces, the only difference is the type of widget used for the input.

We have built tools that allow the automatic application of UX refactorings on the client side of running applications (Grigera et al., 2017b; Gardey et al., 2020). These refactorings are called Client-Side Web Refactorings (CSWR) and perform small transformations to the Document Object Model (DOM) of web pages through scripts. Moreover, since there is usually more than one CSWR to solve a given smell, the tools allow to create different versions of the web application, each one with a different CSWR, for the purpose of running user tests on each version (Grigera et al., 2018; Gardey et al., 2020).

For those smells that have alternative refactorings, it becomes necessary to evaluate them and find the best solution for each particular situation. For instance, in the case of *Unformatted Input*, the three select boxes may work better than a datepicker for entering a birthdate, while a datepicker may be more helpful to enter a date in a booking system. It could also be the case that a refactoring does not cause an improvement on a specific UI, and that the original UI works better for the users. As a consequence, we defined the interaction effort score to have a criteria for comparing the performance of different UX refactorings for a given widget.

Since we aim at obtaining the effort score automatically, in this work we propose models to predict the score assigned by UX experts based on their subjective criteria, using interaction data captured from users on the target application. Figure 2 shows the overall approach based on the alternative refactorings for *Unformatted Input*. Using this prediction approach, UX experts could deploy the alternative refactorings and automatically collect interactions

from different users in order to select the refactoring whose widget results in a lower interaction effort. Nevertheless, it is important to mention that this idea is not limited to UX refactorings. It is possible to evaluate the interaction effort of a widget regardless of how it was created.

Figure 2: The overall prediction approach. Models predict the effort of a single user interaction based on its micro-measures collected.

The first step for predicting the interaction effort was to define the set of features that will be used as predictor variables (the micro-measures in this case). The following section describes the method for creating the micro-measures and the selected ones for each widget type. Once we defined the micro-measures to capture, we started with the data collection process to get real user interactions rated by different UX experts from the industry. Finally, we evaluated different prediction models and we report the outcome of decision trees, including an analysis of the micro-measures importance.

4. Micro-measures

We analyzed a set of six types of interactive widgets typically found in web forms (see Figure 3). Since each widget type has its own pattern of interaction, it was necessary to find a different set of micro-measures for each one. In our previous work, we defined micro-measures for text inputs and selects (Grigera et al., 2019); yet, in the present work some of those micro-measures have been replaced and the set of widget types have been extended to include link anchors, radio buttons, date selects and datepickers.

With the goal of modeling the behavior around each type of widget under analysis, we first consolidated a list of possible micro-measures from existing literature and then performed a preliminary experiment with UX experts to refine the list. Since the micro-measures had to be automatically captured, features that required human interpretation such as "frustration" or "intent" were ruled out from the start. Then, a preliminary set of user interactions observed by UX experts was recorded to refine the list of micro-measures, leaving only the most relevant ones in terms of helping decide how much effort users make. Some examples of these features are pause times, mouse speed or time spent around a widget.

This section first describes the sources that inspired the selected micro-measures and then, it defines the resulting ones for each type of widget.

4.1. Selection Process

We followed a multi-step process for creating and refining the list of micro-measures. We started with a list of candidate micro-measures extracted from the existing literature, based on different kinds of interaction like mouse movement, keyboard activity or idle time analysis.

Mouse movement is a relevant source for behavior analysis and prediction, and we found many features reported in the literature for this aspect of interaction. For instance, the action of repeatedly revisiting a widget has been reported as an uncertainty predictor (Dias et al., 2019), so we design specific micro-measures such as "Interactions", to capture the number of times a user gains focus on a widget, or "Hover & Exit" and "Exit & Back", which try to capture the mouse movement towards the widget or away from it. The study of dwell times and pauses has long been considered for behavioral study, even for user profiling (Hurst et al., 2007; Attig et al., 2019). In our model, pauses

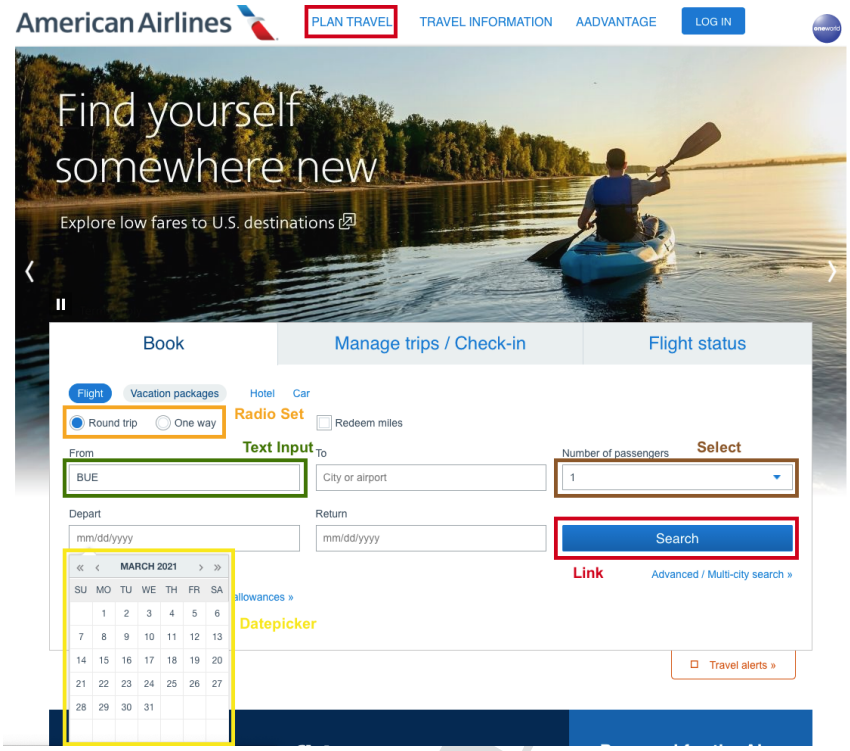

Figure 3: Example of a website including most of the types of the widgets considered in this work

are being considered in micro-measures "Dwell Time" and "Typing Latency". Mouse speed is also usually mentioned, sometimes as a confusion indicator. Speicher et al. (Speicher et al., 2014) use different mouse-related features such as cursor speed, trail length and hover times in order to predict quantitative metrics of usability, many of which were also captured in micro-measures like "Mouse trace length around widget" or "Dwell+Hover time". They also use some keyboard-related ones, another widely used source for interaction behavior analysis. One of the earliest works in this area (Atterer et al., 2006) analyzes user actions in tasks such as filling out web forms. It is worth to mention that in the related literature, the micro-measures are usually analyzed in the context of full user sessions, whereas in this work we limit the capture to specific widgets and their surrounding areas.

After selecting the first candidate micro-measures, the preliminary experiment proceeded as follows. First, three UX experts were asked to observe and rate individual widget interactions from screen captures (these practitioners make user testings on a daily basis as Senior UX Researchers in the local industry). The ratings were classified in 4 different scores, from 1 (least demanding) to 4 (most demanding). After the rating process, we re-watched the screen captures along with the experts and asked them about the user behaviors they had considered in order to select the rating, which helped refine the initial list of micro-measures by adding new ones or removing those failing to have an apparent impact on the scores. The list of micro-measures was also trimmed for technical reasons that prevented capturing some events with JavaScript. For instance, all mouse activity taking place within an open select box is intercepted by the browsers; therefore, we could only work with micro-measures surrounding the element, or time-related analyses. We also considered avoiding metrics that would have caused noticeable performance issues, although we never had this problem.

4.2. Analyzed Widgets

Since our objective is to be able to compare the performance of various types of widgets for the same input (like the example shown in Figure 1), the models use the set of micro-measures for each type of widget to predict a single score that we call *interaction effort*. This unique score makes it possible to compare different widgets.

The following list details the micro-measures selected for each type of widget under analysis. The first part of

the list includes the common micro-measures shared by all widgets, while the rest are specific for each type. The order of micro-measures in this list is random although, in the process of predicting the interaction effort score, some micro-measures were shown to weigh more than others. This analysis will be presented in a later section.

Shared Micro-measures

- **Dwell+Hover time:** time the user is inactive while the mouse cursor is over the widget area, or with the focus on the widget and the mouse tilted (dwell time), plus time the user is moving the mouse over or around the widget area (hover time). The widget area is a bounding box that contains the widget plus a padding, which allows to capture mouse activity on the surrounding elements such as the widget label. Mouse activity on these elements is considered part of the target widget.
- **Mouse Trace Length around Widget:** the complete length of the mouse trace captured within the widget area. In combined widgets, such as radio button sets or date selects, it is a single area that encompasses all the widgets in the group.
- **Hover & Exit:** a direct movement of the mouse cursor towards the widget area, followed by counter movement away from it, with a similar but opposite trajectory.
- **Exit & Back:** a mouse movement away from the widget area, followed by a counter movement back to it, with a similar though opposite trajectory.
- **Interactions:** number of times the user gained focus on the widget. For text input and datepicker, the focus event is considered, while for the remaining widget types, a focus means that the user enters with the mouse on the widget area and stays within it for at least 2 seconds. This threshold was determined by analyzing the preliminary interaction recordings. Lower values starting from 400 milliseconds were evaluated but they were prone to detect unintentional widget interactions such as a user quickly passing through a widget area when going from one page section to another. Those widget logs with "0" as value in this micro-measure were not included in the datasets, for they are not valid widget interactions.

Text Input

- **Focus Time:** total time the widget has the focus. Similarly to *Dwell+Hover time*, this micro-measure serves to calculate the time that a user spends on the widget when the keyboard is used instead of the mouse, to enter and exit from the widget.
- **Typing Latency:** time elapsed from focus gain to first keystroke. This time is also included in *Focus Time*.
- **Typing Speed:** total typing time in proportion to number of characters typed. It is the time elapsed from the first to the last keystroke divided by the keystrokes amount.
- **Typing Pace SD:** intra-keypress time standard deviation. It estimates the typing pace of the user. The closer the intra-keypress times are to each other (regardless their magnitude), the lower the standard deviation is. This means that interruptions during the typing process result in a higher value.
- **Corrections:** number of deleted characters (with backspace or delete). It gives a measure of the errors made on this widget.
- **Input Switches:** changes from keyboard to mouse (or viceversa). Many users were found to navigate a form with the tab key while the form elements are visible, and switch to the mouse to scroll to the next section while interrupting the interaction with the current widget.

Select

- **Options Display Time:** total time the options list is open. It calculates how much time the user needs to find and select the desired option.
- **Options Selected:** number of times the selection is changed. Very much as *Corrections* for Text Input, it accounts for the (intentional or unintentional) errors made.

Link Anchor

- **Misclicks:** number of missed clicks on the widget area. A missed click is the one that is supposed to activate a specific link but no response is generated on the UI. This behavior may be caused by widget style, for instance, when the widget area is not clearly delimited or the widget does not provide an appropriate feedback to the user when it can be clicked.

Radio Button Set

- **Hover to First Selection:** time elapsed from hover to first selection. Although this time is included in *Dwell+Hover time* for this widget, it was decided to distinguish this time into a separate micro-measure since we observed in the recordings that many times users hesitate before the first selection.
- **Selections:** number of selections made. Similar to *Selections* for Select.
- **Misclicks:** number of clicks within the element not causing an option selection. It is the case, for instance, when an option label is clicked and it is not linked to the corresponding radio element.

Datepicker

- **Selections:** number of date selections made.
- **Clicks:** number of clicks on the controls of a datepicker excluding date selections. It estimates how much a user has to navigate in order to reach the desired date.

Date Selects

- **Combined metrics of each individual select:** that is, the combination of metrics for day, month and year.

5. Data Collection

We conducted a data collection process over a period of six months during 2020. This process consisted in capturing a substantial amount of real interactions for each of the six widgets under analysis, then manually rating them with an effort score, and using this data to train the model and verify its performance.

Different user tests were carried out in order to collect the data needed for training and testing the models. Each user test session was composed by a set of recorded interactions with single widgets, each of these containing its respective micro-measures. Besides this data, a complete screen recording was stored for each session, which was later replayed by the experts in order to manually rate the interactions.

5.1. Participants

Fifty-two end users (23 females, 29 males) were recruited to run tests and provide interaction behaviors. In order to diversify the sample, participants from different age groups, backgrounds, and familiarity with web forms were selected. The mean age was 38 years (Max = 79; Min = 16; SD = 13). Backgrounds were as diverse as high school students, professionals in Architecture, Accounting, Computer Sciences, Medicine, Design, Agronomy, as well as Early Childhood Teachers and retirees. A total 223 user test sessions were obtained with 2278 single widget interactions.

The data collection was conducted during the COVID-19 pandemic, so all tests had to be run remotely and the data acquisition process was completely online. Participants were asked to install a web extension in their browsers that recorded their interaction, and then complete some tasks in different websites. For this, they received a link by e-mail with the two-step instructions page[1]. The first step guided subjects through the installation process for the data capture tool (described in Section 5.4.1), and the second step included the tests to be completed. For each test, subjects were provided with the description of the task they had to perform as well as some data to fill in (product to search, car model to insure, etc.). Participants were allowed to perform small alterations on their personal information, such as changing a digit in their ID or phone number if they felt uncomfortable by submitting real values.

In order to rate the interaction samples, a second group of four UX experts was summoned to act as raters, each one with more than 10 years of experience in the industry. At the time of the experiment, all of them were serving as UX researchers and, as practitioners, they had performed numerous user tests and heuristic inspections of web applications.

[1] http://selfrefactoring.s3.amazonaws.com/ijhcs_experiment/instructions.html

5.2. Websites and tasks for end-users

We used six websites selected considering the presence of widgets under analysis and different application domains. Systems for e-commerce, medical care and public information were included. The chosen websites had to meet two requirements: (1) they had to include as many of the target widget types as possible; (2) they had to allow for recognizable tasks for the participants.

Each participant completed one user test on each of the four selected websites. Out of the six available websites, two were given to all the participants, and the remaining four were proportionally assigned so each website would get approximately the same amount of participants. The tasks to be completed were the following:

- Search and add products to the shopping cart on a clothing e-commerce system (https://www.lacoste.com/). Participants had to look for two products (a jacket with discount and a shirt) and add them to the shopping cart without completing the checkout.
- Consult the cost of a car insurance for a specific model (https://selfrefactoring.s3.amazonaws.com/testsites/lacaja.html). In order to get the cost of the insurance, participants had to fill in a form with their email and phone number, and with information of the car including its license plate, brand, model and version.
- Request an appointment at a medical clinic (https://selfrefactoring.s3.amazonaws.com/testsites/mayoclinic.html). Participants filled in a form with their personal information, the city where they wanted to have the appointment, the reason for the medical consultation, and the date interval in which they were available.
- Make a reservation in a parking lot of at an international airport (https://selfrefactoring.s3.amazonaws.com/testsites/aerolineas.html). This task consisted in filling in a reservation form with start and end dates, vehicle license plate and vehicle type (motorbike, car, bus).
- Register a given vehicle with the electronic toll system (https://selfrefactoring.s3.amazonaws.com/testsites/telepase.html). The task required to submit a form with contact information and address of the participant, vehicle model and license, and finally credit card information to process the payment.
- Obtain a personal worker ID on a government website (https://www.anses.gob.ar). In this task, it was necessary to find the option to get the worker ID on the website, and then to fill in a form with participant's personal ID, name, gender and date of birth.

The first two tasks (e-commerce and car insurance) were completed by all the participants. The reasons behind the decision being that the e-commerce task was the only one focused on navigation and search to capture interactions with different link widgets, whereas all the other tasks were form-based; meanwhile, the car insurance task involved interactions with radio-sets which are not very common in the remaining tests.

The first and last task were done on the real websites. For the other tasks it was necessary to create a dummy version of the website so as to avoid sending sensitive information to the server. Pages were replicated including all form fields with their constraints and validations for participants to interact with the websites and submit the forms as if they were the real ones.

In order to capture all interactions with their micro-measures and the screen recordings for UX experts to rate, we devised a set of tools and procedures that we detail in Section 5.4.

5.3. Rating of interaction behavior

Once the interaction samples were finally collected, captured interactions were split into four subsets; furthermore, instead of giving each subset to a single rater, a pair of raters was assigned to each. Therefore, four pairs of raters were needed, one for each subset of interactions. Since we had four raters, each one was paired with two others in order to get four pairs. That is, if we labeled the raters as A, B, C and D, then the four pairs were conformed as A/B, A/D, B/C, C/D. This design of overlapping pairs contributed to reduce potential biases, as described at the end of this subsection.

By observing screencasts of the captured sessions, the raters assigned an effort value to each single widget interaction with the help of a rating tool. The criteria for establishing an effort rating was not stipulated or discussed beforehand: each rater only knew that there were four levels of effort to choose from (that were defined as incremental and equidistant), and they assigned the rating according to their own perception without being aware of the micro-measures calculated under the hood. Experts had no other source of information –such user gestures or audio recordings– because the intention was to capture user interactions in an environment as close to the real context of

use as possible, in which users are not aware that they are being evaluated. In the cases where two raters assigned different values to the same interaction, they were asked to agree on a single value during a second round of rating. This consolidation round was also assisted by the tool, which features a special "consolidation" mode.

Considering this procedure, the overlapping pairs design had two benefits: on one hand, each single widget interaction was rated by two independent experts and, on the other, all experts shared a consolidation session with two others instead of one, leading to more uniform ratings overall.

5.4. Tool Support

Data acquisition of both the end-user interactions with the micro-measures and the rating of these interactions given by the UX experts required a systematic method and tool support. Without them, data acquisition would have been a very exhausting process, almost actually impossible to execute during the quarantine restrictions due to COVID-19. Besides making the process feasible and helping to maximize the number of samples, having a strict protocol and tool support also improved internal validity, providing more uniform conditions for both the raters and end-users without the need for a moderator.

The toolset we used to acquire data consists of two modules: a **capturer** used by the participants acting as end-users, that records all user interaction with the website, and a **rater** used by the UX experts, that presents the screencast along with the widget interactions (without revealing the micro-measures). Expert used this tool to assign an effort value to each interaction and thus generating the set of labeled samples. Figure 4 shows a diagram of the acquisition process followed using these tools, from the capture of interactions at end-user sessions to the rating process carried out by UX experts.

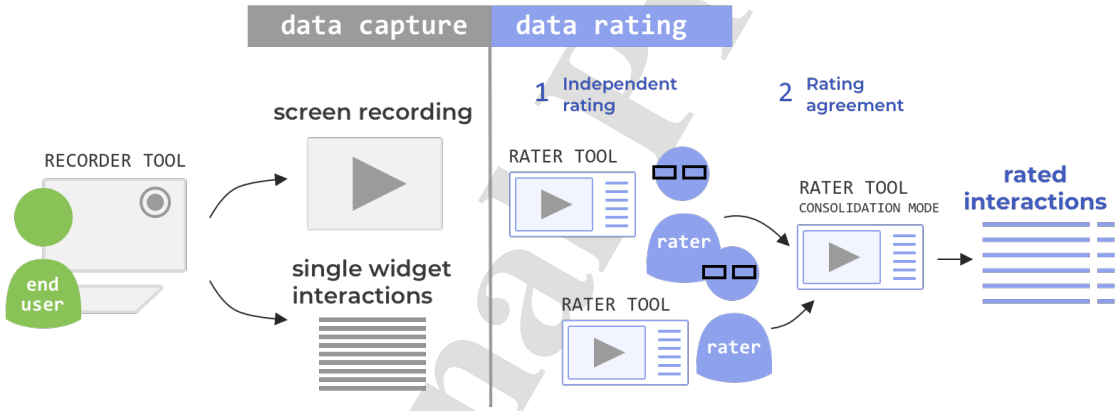

Figure 4: Data acquisition process supported by the toolset developed.

5.4.1. Capturer

Participants recorded their interaction using our capturer tool. This tool is a browser extension that records user sessions, which consist of a screen recording and a collection of interaction logs that contain the corresponding micro-measures for each widget that the user interacted with. Participants were asked to install this extension in their browsers before carrying out the required tasks.

This browser extension adds a single button next to the address bar to start and stop the recording, as shown in Figure 5. When the user activates it, 2 subcomponents start recording: the *screen recorder* captures the current interaction as seen by the user (visual data), and the *micro-measures logger* analyzes each individual widget interaction to compute the micro-measures (analytical data), depending on the type of widget, as listed in Section 4.

Regarding the implementation, the *screen recorder* uses a third party library called *Rrweb*[2] which generates a screencast using as input the target page DOM, and then tracks all the modifications and user interaction events (mouse movement, clicks, etc.) that occurred on site.

[2]https://www.rrweb.io

Figure 5: Google Chrome toolbar with the extension record button.

5.4.2. Rater

The rater module is a web frontend that UX experts used to replay the user sessions and rate the interaction on each single widget. When an expert accesses a session, they can see a video player with the screen recording, together with a form to rate the interaction on each widget. Each interaction has the widget label and its type (A=link anchor; R=radio; S= select, etc.). This layout can be seen in the screenshot shown in Figure 6: screen recording player on the left, and interactions on the right.

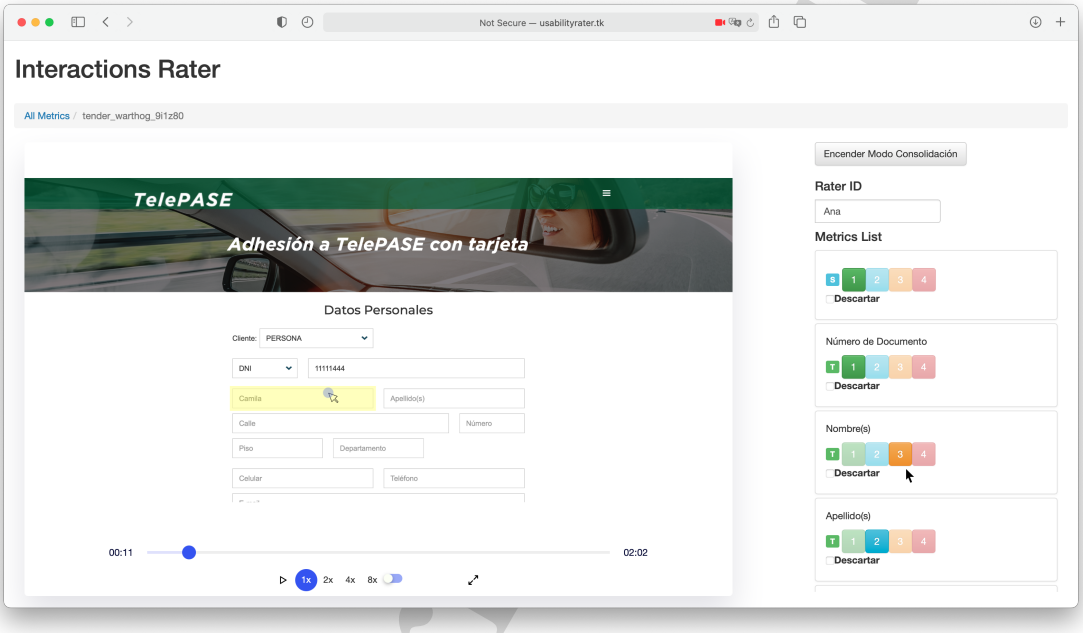

Figure 6: Screenshot of rater tool for assigning effort values to single interactions. On the left is the screencast player and on the right is the form for assigning values. The top right button labeled as "Encender modo consolidación" switches the consolidation mode. The metrics list features 4 buttons for assigning 1-4 rating, and each one displays the widget's label, if available. Notice also that the micrometric where the cursor is (labeled as "Nombre(s)") highlights the associated widget on the screencast.

When rating a set of interactions for a particular session, the tool is able to link what the player is showing with the captured micromeasures, so when the expert puts the mouse over an interaction, the tool highlights the corresponding widget in the video (as shown in the screenshot). This allows the raters to put each effort score right after they observe the interaction. It is important to mention that the raters are not allowed to observe other raters' scores assigned to the interactions (if any).

In order to unify the multiple scores that interactions may have assigned, we added a consolidation mode which, when activated, analyzes the scores and then highlights the interactions with conflicting values, showing the value that each rater has assigned and allowing to select a final score under the label "Consolidated".

6. Prediction Models

As we previously described, the interaction effort is a score assigned by UX experts, who carefully analyze a screencast and decide a value based on what they observe. Given that the score is subjective, UX experts may use different criteria to assign it. Thus, our goal in this work was to investigate if, by using machine learning techniques, it is possible to predict the interaction effort automatically. With that purpose we developed prediction models of the interaction effort based on micro-measures automatically captured for each widget type of interest. Models use as input the calculated micro-measures of a specific widget interaction and return a score that ranges from 1 (effortless) to 4 (considerable effort). A regression approach was chosen to predict continuous values that account for the distances between the predicted and real scores in the error function. For instance, we consider that ratings of 2 and 3 are closer together than 1 and 4.

6.1. Data preparation

As described in Section 5, data was collected by capturing user interactions on real websites. These recordings include a screencast of the user session and the corresponding micro-measures, automatically recorded using a custom script. As previously explained, each training sample was labeled by two different UX experts who analyzed the screencasts and assigned an effort score to the widget interactions. After rating all interactions, the two experts agreed on a consolidated value for the interactions that were rated with different scores.

Once data had been collected, a cleaning process for each dataset was carried out in which we decided to discard widget interactions with missing or outlier values in any of their micro-measures. Table 1 below presents the number of collected interactions for each score of the six widget types selected. Outlier and wrong values were detected through the analysis of the corresponding screencasts. By watching the interactions it was possible for us to identify errors and discrepancies in their calculated micro-measures. For instance, there were interactions for which the Dwell+Hover time observed on the screencast was considerably lower than the one calculated by the micro-measures logger. The mismatches between the screencasts and the micro-measures may have taken place due to errors in the logger or unexpected user behaviors not observable in the recordings. Such interactions were not taken into account to prevent misleading the training process.

Concerning models features, a decision was made to discard both *Exit & Back* and *Hover & Exit*, since they mostly got 0 values, meaning that the interaction patterns that they captured were quite rare.

Table 1
Number of collected interactions for each widget type.

Collected Interactions			
Widget	**Total**	**Discarded**	**Used**
Text Input	755	60	695
Select	394	27	367
Anchor	474	74	400
Radio Button Set	582	27	555
Datepicker	171	2	169
Date Selects	101	9	92

6.2. Model selection

With the aim of identifying the model that best fitted each widget dataset, different regression approaches were evaluated: linear regression, support-vector machines, multi-layer perceptrons and decision tree regressors. The performance of the models was analyzed in terms of the mean absolute error (MAE) and the r-squared coefficient of determination (R^2), which is used to determine how well the model explains the variance of the rating values. For the evaluation, we employed repeated cross-validation to reduce the bias that may arise when datasets are randomly divided into training and testing.

We found that decision trees outperform the other algorithms in all six datasets. Table 2 shows the results. Considering that the predicted value is a continuous one, it must be rounded to the closer integer to obtain a valid score, so a MAE=0.27 was considered to be acceptable. Consequently, the resulting models show that micro-measures can predict the interaction effort.

For all decision trees, a predefined *max depth* and *minimum samples per leaf* was used so as to avoid the overfitting. Regarding these hyperparameters, we found an optimal performance with a minimum of 10 samples per leaf and a max depth of 5. In the specific case of text input, link, and select, where interactions with rating 1 were much more frequent than the others ratings, a random under-sampling was performed before the training phase to minimize model bias.

Table 2
Performance of each widget model.

Widget Model	MAE	R^2
Text Input	0.22	0.72
Select	0.16	0.82
Anchor	0.19	0.82
Radio Button Set	0.27	0.65
Datepicker	0.26	0.57
Date Selects	0.19	0.69

6.3. Micro-measures importance

We analyzed the weights that decision trees assign to each micro-measure according to Gini importance. In particular, we further analyzed the screencasts with the goal of establishing a relationship between what we observed on the recordings, and the importance that models assign to each micro-measure. The insights obtained allow us to refine even more the list of micro-measures, and to identify interaction patterns that may lead to poor user experiences.

From the shared micro-measures, *Dwell+Hover Time* was the one that resulted in a higher importance along the different widget types, which can be due to its capability to separate the different ratings. Figure 7 shows the boxplots of this micro-measure for the collected interactions grouped by rating, where it can be observed that the median is different for each rating in all widgets. This is even more evident for the widgets in which *Dwell+Hover Time* was the most important micro-measure such as link and radio button set. In these cases the values closer to the median (Q2 and Q3) are also clearly separated between the ratings.

The next subsection presents the considerations that are specific of each widget model.

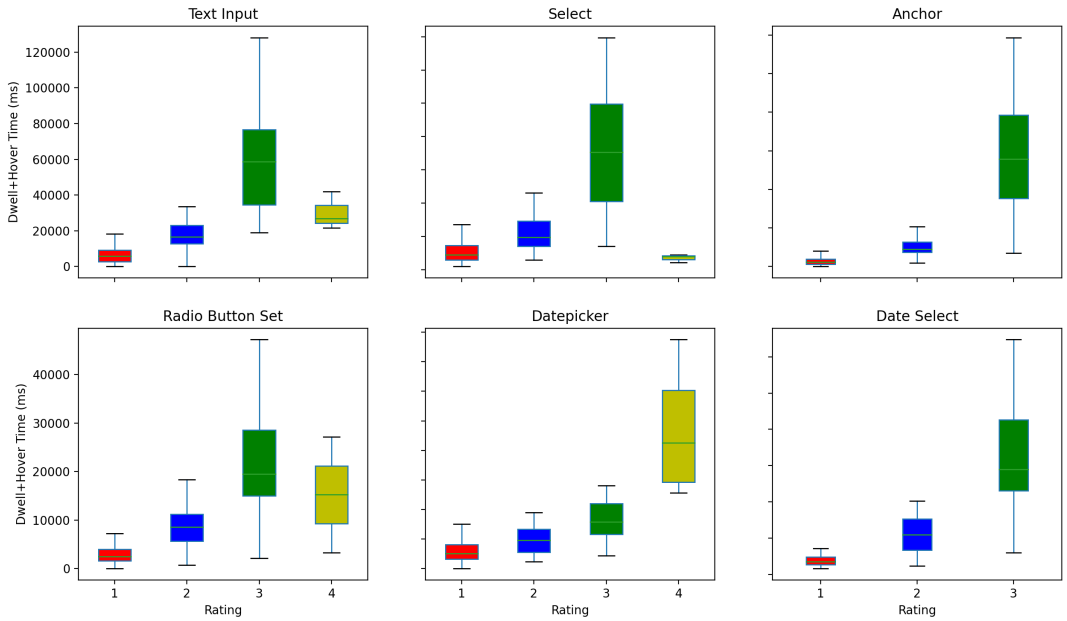

Figure 7: Boxplots of *Dwell+Hover Time* for each widget type.

6.3.1. Text Input

In terms of the Gini importance, the most important micro-measures are *Typing Pace SD*, *Focus Time* and *Dwell+Hover Time* (see Table 3). The last two micro-measures are highly correlated (0.83 Pearson's coefficient), so we ran the model excluding one of them from the selected features and the prediction results were not modified. The correlation between *Dwell+Hover Time* and *Focus Time* is likely to occur because they both serve the purpose of measuring the time that a user spends on a widget, so they overlap when the user employs the mouse to focus on the widget and leaves the mouse around it while typing the input. Although there exist cases in which they do not overlap, for instance, when the keyboard is used to enter and abandon the widget, such cases were not frequently observed in the collected interactions.

The remaining micro-measures bear minimal importance in the decision process. A reason may be found in that just *Typing Pace SD* together with *Dwell+Hover Time* or *Focus Time* allow achieving a clear separation between the different scores.

Table 3
Average Gini importance for each micro-measure in the Text Input model.

Micro-measure	Gini importance
Typing Pace SD	0.78
Focus Time	0.082
Dwell+Hover Time	0.058
Corrections	0.024
Typing Speed	0.008
Interactions	0.006
Mouse Trace Length	0.004
Typing Latency	0.002
Input Switches	0

6.3.2. Select

The most important micro-measures were the ones that measure the time that the user spends on the widget: *Options Display Time* and *Dwell+Hover Time* (Table 4). The first one estimates the total time that the option list is open, while the latter only considers the mouse dwell and hover around the select when the options list is closed, as the browser does not allow to capture interaction events when the options list is being shown. The high importance of *Options Display Time* reflects the most common interaction pattern observed in the screencasts, which is that users spend most of the time on the widget scanning through the options list in order to find the desired choice, so it was the main factor which determined the corresponding interaction score. The low importance of *Options Selected* could be explained by the fact that once users had picked their desired option, it was usually unlikely for them to change it, consequently, most of the collected interactions have only one selection. Even in cases when they decided to change the option, they were more likely to do so only once. The same lack of variability in the values was observed in the *Interactions* micro-measure.

Table 4
Average Gini importance for each micro-measure in the Select model.

Micro-measure	Gini importance
Options Display Time	0.801
Dwell+Hover Time	0.194
Mouse Trace Length	0.003
Interactions	0
Options Selected	0

6.3.3. Link

Dwell+Hover Time is the feature with highest Gini importance for the link interaction effort prediction (Table 5). Analyzing the screencasts, it becomes evident that users hover over a link while deciding whether its purpose matches the goal they want to achieve, so the time it takes to make such decision can be used to determine the interaction effort score. *Mouse Trace Length* is correlated with *Dwell+Hover Time* (0.5 coefficient) as users tend to scan link content

by moving the cursor, especially when it contains text. Regarding the *Misclicks* micro-measure, it was observed that clicking on a link surroundings with the intention of activating the link was an infrequent behavior, so that is why the model found this micro-measure irrelevant. Moreover, it is prone to false positives when a group of links are displayed together, as many of the detected missed clicks on a link were, in fact, targeted to another link that appeared very close.

Table 5
Averaged Gini importance for each micro-measure in the Link model.

Micro-measure	Gini importance
Dwell+Hover Time	0.93
Mouse Trace Length	0.065
Misclicks	0.003
Interactions	0

6.3.4. Radio Button Set

Just as in the previous model, *Dwell+Hover Time* was the most important micro-measure in the decision process (Table 6). In general, we observed that users put the mouse on the target radio button set and then analyze the available options to select the desired one. The positive correlation with *Mouse Trace Length* (0.53) could be due to the behavior of hovering over each of the candidate options before the selection, so the hesitation is also reflected on the mouse movement. *Hover To First Selection* is contained in *Dwell+Hover Time* and their values tend to be very close when the user abandons the widget after the first selection. This may explain the minimal importance of *Hover to First Selection*, as in most of the interactions users made only one selection. As with the missed clicks in link, clicking on a radio button set that did not trigger an option selection was a very rare interaction, consequently, *Misclicks* was not considered by the decision tree.

Table 6
Average Gini importance for each micro-measure in the Radio Button Set model.

Micro-measure	Gini importance
Dwell+Hover Time	0.908
Mouse Trace Length	0.037
Selections	0.037
Hover to First Selection	0.027
Interactions	0.006
MisClicks	0

6.3.5. Datepicker

Table 7 shows their average feature importance. The number of date selections (*Selections*) was the main feature that enabled the separation of interactions with different ratings. Users had to correct the selection very often (30% of the collected interactions have at least 2 selections); the reason is that there were instances of this widget with constraints for the allowed values, though they were not clearly informed, consequently, users had to change their selection after attempting to submit the target form with wrong values. This happened for instance in the parking lot reservation website, where the start date had to be at least two days after the current date, and the total amount of days between the selected dates could not exceed the amount determined by the chosen fare.

The importance of *Interactions* is also due to date corrections, as every new date selection causes an increment in the number of interactions with the target widget. In fact, both features present a very strong correlation (0.9). Regarding the *Dwell+Hover Time* and *Mouse Trace Length*, it is important to mention that mouse movements on the calendar could not be captured due to technical reasons, so such micro-measures on a datepicker only considered the input surroundings as if there was no calendar. This may explain why there was no substantial difference in terms of *Dwell+Hover Time* and *Mouse Trace Length* for the different ratings since the users spent most of the time with the calendar opened. The *Clicks* micro-measure, whose goal is to estimate how much the user has to navigate on the calendar in order to find the desired date, was not so important for the model because in general the target date was zero or one click ahead the starting point (the current date).

Table 7
Average Gini importance for each micro-measure in Datepicker model.

Micro-measure	Gini importance
Selections	0.73
Interactions	0.160
Dwell+Hover Time	0.070
Clicks	0.021
Mouse Trace Length	0.008

6.3.6. Date Select

This model takes as input the same features as the select model with the difference that, in this case, the values of the micro-measures are calculated aggregating the interactions of the three target select inputs, i.e., month, day and year. Observing the feature importance of the model (Table 8), the most important one is the *Options Display Time*, just as with select. Besides that, *Dwell+Hover Time* gained more relevance because users tended to dwell on the widget area after the selections made in order to check the selected date. *Mouse Trace Length* also became more significant since it is common to use the mouse to switch from one input to another. Mouse movement even increased when users needed to correct any of the selected values.

Table 8
Average Gini importance for each micro-measure in Date Select model.

Micro-measure	Gini importance
Options Display Time	0.561
Dwell+Hover Time	0.333
Mouse Trace Length	0.105
Interactions	0
Options Selected	0

7. Discussion and limitations

The proposal in this work allows assessing the user interaction effort on different widgets that are standard UI elements of web forms. Compared to other approaches that analyze user interaction with a broader perspective, i.e., at the page or user session level, we believe that our fine grained interaction analysis makes it easier to determine whether users are struggling with particular UI elements; thus, it becomes possible to suggest solutions (in terms of UX refactorings) to the UX smells that may arise. Moreover, it allows defining a concrete measure to compare small design variations in terms of UX. This, together with the fact that the interaction analysis is not limited to a predefined user task on the target website, makes the interaction effort metric suitable for A/B testing. Thus, the assessment of the user interaction effort can take place in production, without user awareness of the presence of a test and without predefined tasks, which are typical of a user testing session.

Using a white-box model such as decision trees allowed us to interpret and analyze how each feature influences the prediction. From this analysis we observed that in general the most relevant micro-measures were the ones that captured the time that users spend on the target widget. Conversely, those intended to account for errors turned out to be less important in comparison, probably due to the fact that the errors encountered determined to some extent the widget interaction time. Regarding the micro-measures that were found irrelevant, they mostly capture interaction patterns that were infrequent in the collected interactions; such was the case of *Hover & Exit* and *Exit & Back* for links, and *Misclicks* for radio button sets and links.

Our method could have been affected by different biases that, in some cases, we tried to anticipate and mitigate. The most important one being, in terms of external validity, the amount of samples gathered in general, and especially the amount of interactions rated as "considerable effort" (ratings 3 and 4). Even if we mitigated the first one by implementing the recorder tool (having learned from our previous experiment), it was not easy to recruit a large number of volunteers, and those who could have provided more "considerable effort" samples were not prone to install and operate the browser extension on their own. In addition, the number of samples was ultimately limited by the amount

which could be rated, involving more effort, since UX experts had to go through each screencast (in some cases, twice, to get an agreement), and this is a much slower and careful task than simply recording the interactions.

The collection of interactions with outlier or wrong values could also have introduced bias in the prediction models. In particular, we mentioned that some interactions presented discrepancies between their screencasts and their calculated micro-measures. We decided not to consider such interactions because we could not detect the causes of those mismatches.

Another potential bias was the subjectivity in the ratings, which also affects external validity. Having a small number of raters might have resulted in a particular (non-representative) way of judging effort, and since the training and testing data comes from the same source, this could threaten the validity of the prediction results. We tried to mitigate this by using different combinations of expert couples in this way: we split the interactions in different groups, and assigned different couples of experts to each one by combining the four volunteers in a balanced way. We also asked them to discuss and agree on the values only after they had gone through the first round of rating. The rater tool did not show ratings from other raters except in the consolidation mode.

As mentioned in the Introduction, effort ratings may be influenced by other factors outside visible interactions, so it would be interesting to combine it with other UX evaluations methods. At present, we are exploring the relationship between single widget effort and overall session effort as perceived by UX practitioners. This could also help to study if there exists a correlation between effort ratings and usability aspects like efficiency, or even satisfaction. Furthermore, in future work we plan to research the possible relationship with properties that can be processed by static code analysis, such as layout, contrast, or other hedonic factors. We envision a composable measure that could be customized by the UX expert when defining a controlled experiment.

The tool for the remote capture of interactions was primarily developed with the aim of maximizing the samples, but it also responded to COVID-19 restrictions. Should we have been able to make such captures in person, we believe it would have been possible for us to collect far more user interaction data (such as volunteers gestures or, at least, their voice) in order to help the experts to produce more accurate ratings. However, this would have drastically reduced the amount of captured interactions. We preferred to work with a higher number of samples that the remote capture and rating allowed, even if it forced us to develop a complex tool support. As an additional benefit, the remote protocol was very consistent and improved both reproducibility and internal validity.

8. Conclusions and Future Work

This paper shows our work on predicting the interaction effort of individual widgets, particularly, six types of widgets that may be found in web forms. Our approach is original in that it measures the interaction of web widgets rather than entire pages or sessions. The models show that is feasible to predict the interaction effort using the micro-measures captured, as we consider acceptable a maximum error of 0.27 (MAE) for this prediction task. In turn, this allows us to use the interaction effort metric in a setting similar to that of A/B testing, i.e., as a metric related to UX instead of conversion rate to compare designs. This article also makes a contribution towards understanding the importance of individual micro-measures when studying the behavior in the interaction between end-users with particular form widgets.

Our approach is similar to the philosophy of code refactoring in that it is based on proposing small changes to improve design, which, when automated, is more tractable and affordable. UX refactorings also produce small changes in the design of a web application, and with the work presented in this article, they may also be tested and compared automatically through the interaction effort metric. Moreover, refactorings may be composed so as to create substantial changes and we believe that the interaction effort metric may as well be aggregated from the ones in individual widgets.

Measuring the interaction effort on the widgets of a web page in an automatic and transparent way also serves to complement user testing; for instance, when there is a need to evaluate a small change in a non-expensive and non-intrusive way (since the subjects of a user test may change their behavior when feeling evaluated).

Future work includes several directions. Firstly, we need to validate the created models. We plan to collect a new dataset with different websites and users from the ones used in this work, in order to analyze how well the models generalize the results. Secondly, we would like to extend the interaction effort metric to other types of widgets not present in forms; for example, widgets for different content media. Related to this, we are interested in studying which measures contribute to the effort when users are not filling forms but consuming content. It may become also necessary to compare other kinds of page components such as text and scroll bars. Thirdly, we are studying different alternatives to accumulate interaction effort from individual users and from more than one widget (Gardey and Garrido, 2020).

The latter will be used when applying UX refactorings involving more than one element. Lastly, as discussed before, we intend to study the combination of the effort rating with other types of measures of the whole page, like page layout and aesthetic, and whole sessions.

Thus, we envision that this work will contribute to boost the integration of User Centered Design practices in agile development processes. On the one hand, it provides an automatic measure for user behavior on the real application that may be used in tight schedules without the cost of user testing or, when time allows, in combination with other metrics for an integrated approach to UX. On the other hand, the interaction effort metric may be used in conjunction with an incremental process of UX refactoring that, keeping the users at the center of development, applies small transformations as UX smells are discovered and selects the best solutions based on user feedback. In conclusion, we believe that measuring the interaction effort may serve different purposes for which we would like to foster research.

Acknowledgements

The authors wish to acknowledge the support from the Argentinian National Agency for Scientific and Technical Promotion (ANPCyT), grant number PICT-2019-02485.

References

Akers, D., Jeffries, R., Simpson, M., Winograd, T., 2012. Backtracking events as indicators of usability problems in creation-oriented applications. ACM Transactions on Computer-Human Interaction 19, 1–40. doi:10.1145/2240156.2240164.

Arroyo, E., Selker, T., Wei, W., 2006. Usability tool for analysis of web designs using mouse tracks, in: Conference on Human Factors in Computing Systems - Proceedings, pp. 484–489. doi:10.1145/1125451.1125557.

Atterer, R., Wnuk, M., Schmidt, A., 2006. Knowing the user's every move: User activity tracking for website usability evaluation and implicit interaction, in: Proceedings of the 15th International Conference on World Wide Web, pp. 203–212. doi:10.1145/1135777.1135811.

Attig, C., Then, E., Krems, J.F., 2019. Show me how you click, and i'll tell you what you can: predicting user competence and performance by mouse interaction parameters, in: Advances in Intelligent Systems and Computing, Springer, Cham. pp. 801–806. doi:10.1007/978-3-030-11051-2_122.

Baddeley, A.D., 1976. The Psychology of Memory. Harper & Row, New York, NY, USA.

Bailly, G., Oulasvirta, A., Brumby, D.P., Howes, A., 2014. Model of visual search and selection time in linear menus, in: Conference on Human Factors in Computing Systems - Proceedings, Association for Computing Machinery. pp. 3865–3874. doi:10.1145/2556288.2557093.

Bakaev, M., Mamysheva, T., Gaedke, M., 2017. Current trends in automating usability evaluation of websites: Can you manage what you can't measure?, in: Proceedings - 2016 11th International Forum on Strategic Technology, IFOST 2016, pp. 510–514. doi:10.1109/IFOST.2016.7884307.

Bouzit, S., Gaelle, C., Vanderdonckt, J., 2016. Automated Evaluation of Menu by Guidelines Review, in: RoCHI - International Conference on Human Computer Interaction, MATRIX ROM, Bucharest, Romania. pp. 1–12.

Brhel, M., Meth, H., Maedche, A., Werder, K., 2015. Exploring principles of user-centered agile software development: A literature review. Information and Software Technology 61, 163–181. URL: http://dx.doi.org/10.1016/j.infsof.2015.01.004, doi:10.1016/j.infsof.2015.01.004.

Budiu, R., 2013. Interaction Cost. Nielsen Norman Group URL: https://www.nngroup.com/articles/interaction-cost-definition/.

Burzacca, P., Paternò, F., 2013. Remote Usability Evaluation of Mobile Web Applications, in: Kurosu, M. (Ed.), Human-Computer Interaction. Human-Centred Design Approaches, Methods, Tools, and Environments, Springer Berlin Heidelberg, Berlin, Heidelberg. pp. 241–248.

Card, S.K., Moran, T.P., Newell, A., 1980. The keystroke-level model for user performance time with interactive systems. Communications of the ACM 23, 396–410. doi:10.1145/358886.358895.

Chen, F., Ruiz, N., Choi, E., Epps, J., Khawaja, M.A., Taib, R., Yin, B., Wang, Y., 2012. Multimodal behavior and interaction as indicators of cognitive load. ACM Transactions on Interactive Intelligent Systems 2, 1–36. doi:10.1145/2395123.2395127.

Chen, F., Zhou, J., Wang, Y., Yu, K., Arshad, S.Z., Khawaji, A., Conway, D., 2016. Theoretical Aspects of Multimodal Cognitive Load Measures, in: Robust Multimodal Cognitive Load Measurement, pp. 33–71. doi:10.1007/978-3-319-31700-7_3.

Da Silva, T.S., Silveira, M.S., Maurer, F., Silveira, F.F., 2018. The evolution of agile UXD. Information and Software Technology 102. doi:10.1016/j.infsof.2018.04.008.

DeLeeuw, K.E., Mayer, R.E., 2008. A Comparison of Three Measures of Cognitive Load: Evidence for Separable Measures of Intrinsic, Extraneous, and Germane Load. Journal of Educational Psychology 100, 223–234. doi:10.1037/0022-0663.100.1.223.

Dias, M.C., Cepeda, C., Rindlisbacher, D., Battegay, E., Cheetham, M., Gamboa, H., 2019. Predicting response uncertainty in online surveys: A proof of concept, in: BIOSIGNALS 2019 - 12th International Conference on Bio-Inspired Systems and Signal Processing, Proceedings; Part of 12th International Joint Conference on Biomedical Engineering Systems and Technologies, BIOSTEC 2019, SciTePress. pp. 155–162. doi:10.5220/0007381801550162.

Dingli, A., Mifsud, J., 2011. USEFul: A Framework to Mainstream Web Site Usability Through Automated Evaluation. International Journal of Human Computer Interaction (IJHCI) .

Dou, Q., Zheng, X.S., Sun, T., Heng, P.A., 2019. Webthetics: Quantifying webpage aesthetics with deep learning. International Journal of Human Computer Studies 124, 56–66. doi:10.1016/j.ijhcs.2018.11.006.

Firmenich, S., Garrido, A., Grigera, J., Rivero, J.M., Rossi, G., 2019. Usability improvement through A/B testing and refactoring. Software Quality Journal 27, 203–240. doi:10.1007/s11219-018-9413-y.

Fitts, P.M., 1954. The information capacity of the human motor system in controlling the amplitude of movement. Journal of Experimental Psychology 47, 381–391.

Fowler, M., Beck, K., Brant, J., Opdyke, W., Roberts, D., 1999. Refactoring: Improving the Design of Existing Code. Addison-Wesley.

Gardey, J.C., Garrido, A., 2020. User experience evaluation through automatic A/B testing. International Conference on Intelligent User Interfaces, Proceedings IUI , 25–26doi:10.1145/3379336.3381514.

Gardey, J.C., Garrido, A., Firmenich, S., Grigera, J., Rossi, G., 2020. UX-Painter: An Approach to Explore Interaction Fixes in the Browser. Proceedings of the ACM on Human-Computer Interaction 4, 1–21. doi:10.1145/3397877.

Garrido, A., Rossi, G., Distante, D., 2011. Refactoring for usability in web applications. IEEE Software 28, 60–67. doi:10.1109/MS.2010.114.

Garrido, A., Rossi, G., Medina, N.M., Grigera, J., Firmenich, S., 2014. Improving accessibility of Web interfaces: refactoring to the rescue. Universal Access in the Inf. Society 13, 387–399.

Grigera, J., Gardey, J.C., Rodriguez, A., Garrido, A., Rossi, G., 2019. One metric for all: Calculating interaction effort of individual widgets. Conference on Human Factors in Computing Systems - Proceedings , 1–6doi:10.1145/3290607.3312902.

Grigera, J., Gardey, J.C., Rossi, G., Garrido, A., 2018. Live versioning of web applications through refactoring, in: ASE 2018 - Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering, pp. 872–875. doi:10.1145/3238147.3240483.

Grigera, J., Garrido, A., Rivero, J.M., Rossi, G., 2017a. Automatic detection of usability smells in web applications. International Journal of Human-Computer Studies 97, 129–148. doi:10.1016/j.ijhcs.2016.09.009.

Grigera, J., Garrido, A., Rossi, G., 2017b. Kobold: Web Usability as a Service, in: Proceedings of the 32nd IEEE/ACM International Conference on Automated Software Engineering, pp. 990–995. doi:https://doi.org/10.1109/ASE.2017.8115717.

Gütl, C., Pivec, M., Trummer, C., García-Barrios, V.M., Mödritscher, F., Pripfl, J., Umgeher, M., 2005. AdeLE (Adaptive e-Learning with Eye-Tracking): Theoretical Background, System Architecture and Application Scenarios. European Journal of Open, Distance and E-Learning (EURODL) 8.

Hassenzahl, M., Burmester, M., Koller, F., 2021. User experience is all there is. i-com 20, 197–213.

Hassenzahl, M., Tractinsky, N., 2006. User experience-a research agenda. Behaviour & information technology 25, 91–97.

Hoda, R., Salleh, N., Grundy, J., 2018. The rise and evolution of agile software development. IEEE Software 35, 58–63. doi:10.1109/MS.2018.290111318.

Hollender, N., Hofmann, C., Deneke, M., Schmitz, B., 2010. Integrating cognitive load theory and concepts of human-computer interaction. Computers in Human Behavior 26, 1278–1288. doi:10.1016/j.chb.2010.05.031.

Horwitz, R., Kreuter, F., Conrad, F., 2017. Using Mouse Movements to Predict Web Survey Response Difficulty. Social Science Computer Review 35, 388–405. doi:10.1177/0894439315626360.

Hurst, A., Hudson, S.E., Mankoff, J., 2007. Dynamic detection of novice vs. skilled use without a task model, in: Proceedings of the SIGCHI conference on Human factors in computing systems, pp. 271–280.

ISO, 2019. ISO 9241-210:2019 - Ergonomics of human-system interaction — Part 210: Human-centred design for interactive systems. ISO/TC 159/SC 4. URL: https://www.iso.org/standard/77520.html.

Janlert, L.E., Stolterman, E., 2017. The Meaning of Interactivity—Some Proposals for Definitions and Measures. Human-Computer Interaction 32, 103–138. doi:10.1080/07370024.2016.1226139.

Law, E.L.C., Roto, V., Hassenzahl, M., Vermeeren, A.P., Kort, J., 2009. Understanding, scoping and defining user experience: a survey approach, in: Proceedings of the SIGCHI conference on human factors in computing systems, pp. 719–728.

Li, Y., Bengio, S., Bailly, G., 2018. Predicting human performance in vertical menu selection using deep learning, in: Conference on Human Factors in Computing Systems - Proceedings, pp. 1–7. doi:10.1145/3173574.3173603.

Michailidou, E., Eraslan, S., Yesilada, Y., Harper, S., 2021. Automated prediction of visual complexity of web pages: Tools and evaluations. International Journal of Human-Computer Studies 145. doi:10.1016/j.ijhcs.2020.102523.

Navalpakkam, V., Churchill, E.F., 2012. Mouse tracking: Measuring and predicting users' experience of web-based content, in: Conference on Human Factors in Computing Systems - Proceedings, pp. 2963–2972. doi:10.1145/2207676.2208705.

Norman, D., 2013. The Design of Everyday Things: Revised & Expanded Edition. Basic books.

Oulasvirta, A., De Pascale, S., Koch, J., Langerak, T., Jokinen, J., Todi, K., Laine, M., Kristhombuge, M., Zhu, Y., Miniukovich, A., Palmas, G., Weinkauf, T., 2018. Aalto Interface Metrics (AIM): A service and codebase for computational GUI evaluation, in: UIST 2018 Adjunct - Adjunct Publication of the 31st Annual ACM Symposium on User Interface Software and Technology, pp. 16–19. doi:10.1145/3266037.3266087.

Paternò, F., Schiavone, A.G., Conti, A., 2017. Customizable Automatic Detection of Bad Usability Smells in Mobile Accessed Web Applications. Association for Computing Machinery, New York, NY, USA. pp. 1–11. doi:10.1145/3098279.3098558.

de Santana, V.F., Baranauskas, M.C.C., 2015. WELFIT: A remote evaluation tool for identifying Web usage patterns through client-side logging. International Journal of Human-Computer Studies 76, 40–49. doi:10.1016/j.ijhcs.2014.12.005.

Sharp, H., Preece, J., Rogers, Y., . Interaction Design: Beyond Human-Computer Interaction. 5th ed.

Speicher, M., Both, A., Gaedke, M., 2014. Ensuring web interface quality through usability-based split testing, in: International conference on web engineering, pp. 93–110. doi:10.1007/978-3-319-08245-5_6.

WPForms, L., 2020. 101 Unbelievable Online Form Statistics & Facts for 2020. URL: https://wpforms.com/online-form-statistics-facts/.

- A metric to compare user interaction effort in alternative web interface widgets for a specific task.
- A set of micro-measures that capture the user's interaction behavior in 6 different types of widget.
- An experiment to collect and rate user interactions with individual widgets in real websites.
- A model that predicts the user interaction effort of a widget based on the corresponding micro-measures and the ratings of UX experts.

Juan Cruz Gardey: Methodology, Formal analysis, Software. Writing - Review & Editing.

Julián Grigera: Methodology, Software, Validation, Writing - Review & Editing.

Andrés Rodríguez: Conceptualization, Investigation, Writing - Review & Editing.

Gustavo Rossi: Supervision.

Alejandra Garrido: Conceptualization, Writing - Review & Editing, Project administration.

Declaration of interests

☒ The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

☐The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: