A generalized data dashboard to visualize user interface logging in IIR experiments

Sam van Berkel
Delft University of Technology
Delft, The Netherlands

ABSTRACT

Performing user interface studies and logging user interactions is commonplace in the Interactive Information Retrieval field (IIR). As a result, the LogUI framework was developed to make logging such interactions an easier task. However, this framework does not come with any visualization tool, requiring the researcher to analyze their data separately. We present a generalized data dashboard that allows the researcher to perform basic exploratory analysis on their data, without needing to parse their logging data themselves. We gathered a set of important logging metrics from existing user interface experiments and based our dashboard design on these metrics. Finally, we verified the usability and effectiveness of the dashboard by interviewing a set of IIR researchers.

KEYWORDS

Interaction Logging, Data Visualization, User Study

1 INTRODUCTION

The field of Interactive Information Retrieval (IIR) covers research related to studying the diverse end-users of information access and retrieval systems. Within this field, Web-based user interface studies are customary, aiming to investigate the behaviour of end-users and examining the usability of a certain interface [8, 18, 19]. Logging and analysing user interactions is a vital component of performing such studies, however, capturing these interactions on web applications can be a complex task [4]. With the aim of simplifying this process, the LogUI [14] framework was developed. This framework abstracts away most of the complexity and allows the researcher to define which events should be logged with a single configuration object.

However, to analyze the data that LogUI gathers, the researcher can sign into the LogUI server back-end and download a text file containing a list of timestamped events. The issue is that such a file needs to be parsed and analyzed separately. This poses two main problems; first of all, the researcher needs to be experienced with either data analysis software or a data science programming language such as Python or R. Secondly, even if researchers have adequate programming and/or statistics experience, writing parsing and analysis software still requires auxiliary work, taking time which could otherwise be used to perform research.

This is my report for the CSE3000 course.

CSE3000, June 17th, 2021, TU Delft
2021. ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/N/A

To solve these problems, we present in this paper a generalized GUI-based tool that can be used to perform basic exploratory analysis on captured logging data. It is generalized in the sense that we aimed to make it usable regardless of how an experiment was run or from which application the logging data was sourced, as long as it was gathered using LogUI. Our solution is a data dashboard that displays key information without requiring the researcher to have ample experience with data analysis. Specifically, we aim to answer the following research questions; **RQ1:** Which behavioural metrics do IIR researchers generally look for when performing user interface experiments? **RQ2:** How can we present a summary of logging data that allows the user to perform an exploratory analysis on the data?

2 RELATED WORK

2.1 Interaction Logging

There have been several web-based interaction logging frameworks since the mid-2000s. Earlier versions mostly came in two categories; requiring either extra client software or a proxy server, which captures user interactions as they were sent to the webserver. Examples that required a client application are Wrapper [16] and PooDLE [9]. Solutions in the second category are UsaProxy(proxy) [6]. Similar to LogUI, none implemented any tools to visualize the data, this had to be parsed separately.

As browser capability increased, JavaScript-only implementations arose, meaning they could easily be integrated into the web application for which the logging data needed to be gathered. While these reduced the amount of required software, they still require a server that gathers all logging data. Examples are ALF [7], UXJs [25] and the subject of our research, LogUI [14]. While these improved the ease of use in regards to logging interactions, they are still lacking when regarding their visualization tools. Of the aforementioned solutions, only UXJs comes with a data visualization implementation.

The UXJs visualization tools are very limited, they provide what they call a "user summary" containing general metrics such as how many clicks a user executed on average or the amount of mouse movement a user performed. Additionally, they provide an "action summary" that has more detailed information regarding the interactions received by different elements on the page. This allows a researcher to analyze more complex behaviour than the previous page does, but they are still only averages on how often an action occurs. If you would want to find the time between certain actions or how often a certain action occurs before another, UXJs would not suffice.

Finally, there are commercial products such as Google Analytics and Hotjar. While these have expansive data visualizations, they are a lot less configurable than their non-commercial counterparts. They excel at displaying general metrics about demographics and page visits but are lacking in the targeted analysis of detailed user interactions, making them less viable to use in research.

2.2 LogUI

LogUI consists of two main components, a client JavaScript library that is placed within the application for which logging is required. And secondly, a server that stores all logging data and houses a simple web page on which some configuration can be done. Also, there are some technologies that the LogUI client is based on that will need further explanation.

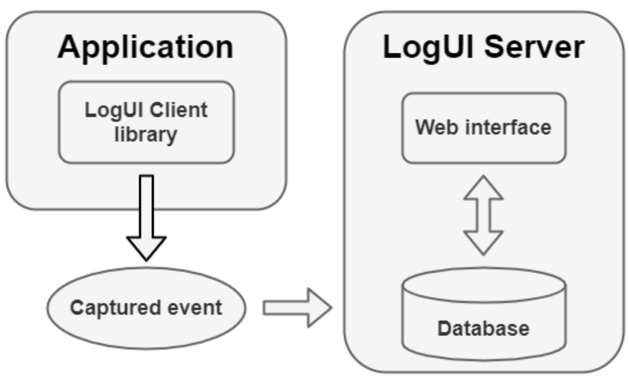

Figure 1: LogUI architecture

2.2.1 Applications and flights. LogUI allows you to log interactions for multiple applications within the same server. This means after setting up LogUI once, you can run experiments over multiple applications on the same server instance without the logging data of different applications interfering with another.

Additionally, experiments often involve a comparison between multiple scenarios within the same application. To accommodate these different experimental setups and configurations, LogUI provides a construct called flights. In short, a flight is simply an identifier that is added to each event so we know to which experimental setup an event belongs when we're analyzing the logging data.

LogUI differentiates between different applications and flights using an authorization token that is generated by the server for each application/flight combination. The researcher places the right authorization token in each client, which LogUI then adds to each event before it sends it to the server.

2.2.2 The DOM and web events. The Document Object Model (DOM) [1] is the data representation of the structure of the content on a web page. It represents a web document as a tree and provides programmatic access, allowing a developer to view and change a document's structure, style or content. Each node in this tree represents an element on the page, e.g. a button or a text paragraph. We can access these elements by providing a selector, which can be a unique id, class or item type or a combination thereof. Giving an element an id can be useful whenever this element occurs on the page only once and we want to target that element specifically. Classes or element types are used to select multiple elements together, such as titles or every button on a web page. We can combine these selectors to form a more specific selector, such as finding all elements that both have class X and Y and are of type Z.

To handle any changes on the web page that may affect code execution, such as user interactions or the resizing of a window, JavaScript uses a concept called events [2]. These events fire automatically whenever such a change occurs and we can subscribe to these events to handle their execution. To handle events we can target an element using the aforementioned selectors and provide a handler function that fires whenever the given event is fired on the given element.

2.2.3 Client library. The LogUI client is a single-file JavaScript library that provides all required functionality to track events associated with a specific element on a web page. All the required configuration is defined within a single JavaScript object. It allows the researcher to define which events should be logged on which element by providing the event name together with a selector and a name that can later be used to identify occurrences of this tracked event.

2.2.4 Server. To aggregate all gathered logging data in a single place, each LogUI client sends their data to a central server. LogUI provides a containerized environment so researchers can easily set up their LogUI server. The server provides a web interface on which the researcher can configure applications and flights mentioned in section 2.2.1, and download the logging data separately for each application.

2.3 Data Dashboards

Data dashboards have their origin in executive information systems [26]. They can comprehensibly display quantitative and qualitative information needed to accomplish an objective or set of objectives. [24] This has made them popular in business intelligence, but it can be a useful research tool as well, as information is at the core of any research. Many guidelines on their design and implementation have been written, such as [5, 15, 24]. These provide information of varying use, the exact implementation can vary vastly depending on the data at hand and the goals of the end-user. As written in [15], "Dashboards can be designed in a variety of ways. There is no one right or wrong way — it depends on the requirements the dashboard has to fulfil."

Research data dashboard implementations are numerous, some modern examples are [20, 27], but research regarding logging data dashboards is sparse. As explained earlier, of the mentioned interaction logging frameworks, only UXJs comes with a data dashboard, which is not very extensive.

3 INTERACTION METRICS IN IIR

To answer RQ1, we gathered a general set of metrics that we could display in the dashboard. LogUI was created as a research tool and is primarily designed for performing user interface experiments [14]. As such we decided to analyze papers that performed such experiments and gather the metrics that they were looking for in their experiment. To make our research more focused, we chose the IIR field, as LogUI was created with this context in mind and this is a field where plenty of user interface experiments are performed.

3.1 Interactive Information Retrieval

The field of IIR covers research related to studying the diverse end-users of information access and retrieval systems [23]. It covers research on information seeking and search behaviour but also involves the development of new methods of interacting with electronic resources. As it often involves analyzing a user's behaviour when interacting with a search engine or another type of digital information retrieval system, there is plenty of research that involves user interaction logging.

3.2 Candidate paper requirements

We considered papers in the IIR field, or parts of their experiment if they met the following requirements:

- They were executed on a web application.
- The behaviour they were looking for could be logged using conventional DOM-events, or a combination thereof.
- They contained experiments that involved logging user interactions, or eye-tracking users.

We will explain these requirements in order.

3.2.1 Web application. Since LogUI is web-based, a requirement to use it is that you're running a web application. We would like to include applications that could reasonably be replicated as a web application, as the technology behind an interface is not important as long as the views and interactions are equal. But deciding to which degree an application can or cannot be written as a web application is a complex subject and beyond the scope of this paper.

3.2.2 Reproducible with DOM-events. LogUI can only log DOM-events, so behaviour patterns that the researchers were looking for need to be identifiable by these events, or a combination thereof. A simple example of a valid user action that can be identified by DOM-events is checking if a user has clicked a button. This can be logged by registering the "click" [2] event.

3.2.3 Logging interactions or eye tracking. We consider any experiments that performed user interaction logging, regardless of which logging framework the researchers used since we could replicate the experiment as long as the two aforementioned requirements are met. We also consider eye-tracking experiments, since there is a strong correlation between the cursor position and gaze position on a computer screen during web browsing [12]. As such we can use the "mouseenter" and "mouseleave" events [2] to track with reasonable accuracy if a user is looking at a certain element on a screen. This means we can replicate some eye-tracking experiments with LogUI as well.

3.3 Generalizing metrics

Since each experiment's user interface is different, the exact metrics that researchers look for will differ as well. However, we can still generalize some of the metrics that people gather from logging interactions. As an example, the metrics "How many search results are opened on average?" and "How many times has a user-selected an article on average?" can both be generalized to "On average, how many times does event X per session?". This general question is what we can gather from each paper and then display in our data summary. The X is then left as a variable where the user of our dashboard can select the required event himself. So, for each paper we analyzed, we converted the metrics in the experiment to a more generalized form that could be shared between different experiments. These generalized metrics are displayed in Table 1.

3.4 Popular metrics

From the analyzed papers, it became clear that researchers are generally interested in averages over a set of users and not the behaviour of specific individual users. Furthermore, each experiment involved a comparison between different experimental setups, so our dashboard needs to support a clear separation between these setups using the aforementioned flights. The five most occurring general metrics are listed below.

- Average count of event X per session.
  Displays how often a specific event occurs per session on average.
- Average time between event X and event Y per session.
  Displays the average time it takes for event Y to occur after event X occurs per session. Whenever event X occurs multiple time before event Y occurs, we only take the last of the occurrences of event X.
- Average time between occurrences of event X.
- Average count of event X before event Y per session.
  Displays the average of how often event X occurs before event Y per session.
- Average count of event X occurs after event Y per session.
  Displays the average of how often event X occurs after event Y per session. If event Y occurs multiple times before event X occurs in between, we only consider the last of those occurrences of event Y

These are the metrics that we chose to implement in our data dashboard.

4 LOGUI DASHBOARD

We developed a data dashboard as our proposed answer to RQ2, this section will provide insight into the design and workings of our developed solution. We will go over the requirements, the user interface, the key features and finally outline some of the implementation details.

4.1 Requirements

Our requirements for the dashboard are as follows:

- Able to visualize the most occurring metrics in IIR user interface research.

Paper	Experiment metrics	Generalized metrics
Azzopardi et al. [8]	• Time between queries	• Average time between event X and Y per session
Wu et al. [28]	• Average amount event X occurs per session • average time between event X and Y per session	• Average amount event X occurs per session • average time between event X and Y per session
Liu and Shah [19]	• Average query length • Average number of clicks • Average number of pages visited • Average dwell time on a page	• Average amount event X occurs per session • average time between event X and Y per session
Choi et al. [13]	• Number of queries • Number of SERP clicks • Average time to first click after query submission • Average time between subsequent query submissions	• Average amount event X occurs per session • average time between event X and Y per session
Liu et al. [18]	• Query length • Number of clicks • Number of pages visited • Mean dwell time on each SERP • Number of saves	• Average amount event X occurs per session • average time between event X and Y per session
Bogaard et al. [10]	• Average length of a session • Average query length • Average number of clicks	• Average amount event X occurs per session • Average session length • Average amount event X occurs before event Y per session
Ong et al. [21]	• Average number of queries • Average typing speed • Average key press errors • Number of documents clicked • Time spent reading a document	• Average amount event X occurs per session • Average time between event X and Y per session • Average time between occurrences of event X per session
Bota et al. [11]	• Search abandonment	• Average amount event x occurred without event y occurring sometime after
Zhang et al. [29]	• Average time a user immediately re-queries after a search query	• Average amount event x occurs without event y occurring in between
Alanazi et al. [3]	• Average time before the first search result click • Average number of fixations on ads • Average number of fixations on organic results • Average scroll rate • Average click rate on ads	• Average amount event X is occurs per session • Average time between event X and Y per session
Rath et al. [22]	• Average query length per session • Average number of queries per session • Average time spent on a specific page per session • Average pages saved per session	• Average amount event X is occurs per session • Average time between event X and Y per session

Table 1: Brief overview of all analyzed papers and their corresponding LogUI events

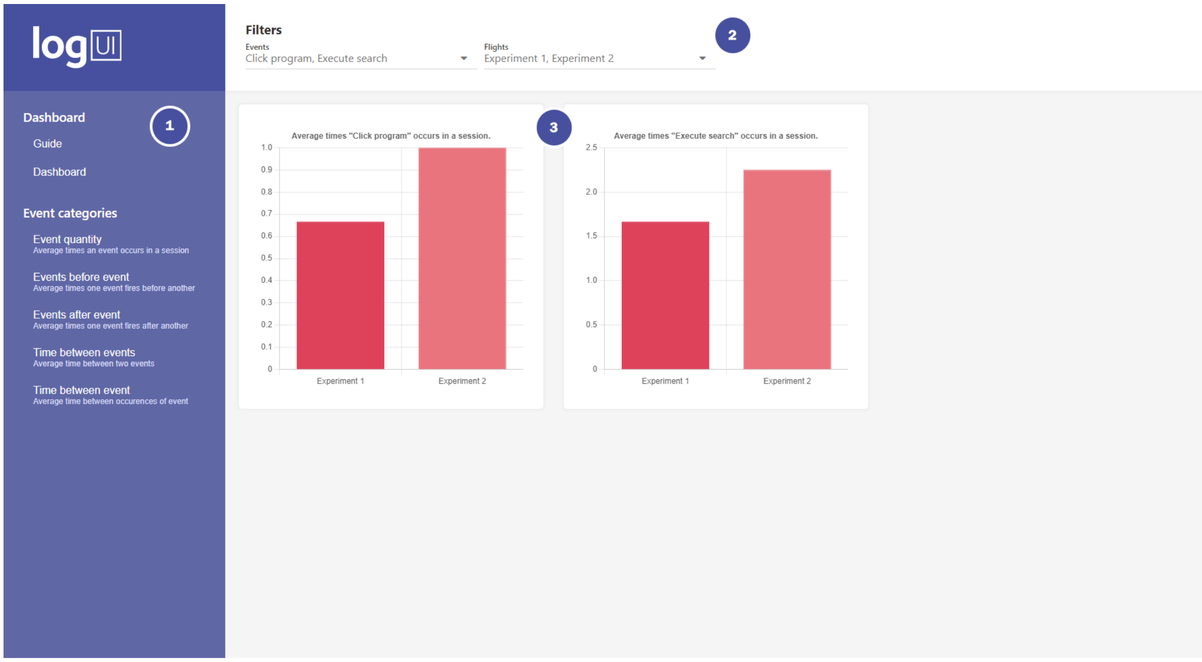

Figure 2: Data dashboard interface

- Able to filter the shown events for each metric
- Able to split any metric by experimental setup.
- Provide a filtered download to aid in further analysis.

The main requirement for this dashboard is that we can visualize the five most occurring metrics that we gathered in section 3.4. This will provide the most requested information while preventing it from being overburdened with tons of different metrics. For each of these metrics, there should be a filter to determine which events are shown. Otherwise, there would too unnecessary information if a lot of events are logged that are not of relevance for a particular metric. Additionally, from section 3 it became clear that researchers generally want to compare metrics between multiple experimental setups. Our dashboard therefore should provide some functionality to split data by experimental setup. Finally, since a dashboard obviously can never present every single piece of information that could be gathered from a given data set, it would be very useful to provide a download for the data at hand. This allows the researcher to still perform extra analysis on the data should the dashboard not provide adequate information.

4.2 User interface

The user interface of our proposed dashboard can be seen in Figure 2. It consists of a sidebar (1), header (2), and content panel (3). In the sidebar, we have the guide, the main dashboard page, and a set of metric pages. Each of the metric pages displays the data for a metric from section 3.4. The sidebar functions as navigation between these different pages. The header houses the event and flight filters that dictate which information is displayed on the dashboard, together with a download button. Finally, the content panel displays the actual logging data in a set of graphs.

4.3 Dashboard features

In this section, we outline the features of the dashboard that aid the user in analysing their data.

4.3.1 Visualizing metrics. On each metric page, the user can select the events and flights they want to see for that metric. The dropdowns are automatically filled from the data, we simply gather every unique event and flight. For each chosen event in the header, there is a separate bar graph that displays the current metric for that event. This allows the user to quickly gather the desired data for each metric. Additionally, every selected event for every metric is displayed on the main dashboard page. This allows the user to easily combine data from different metrics, without constantly having to switch between pages to get an overview. Because every chosen metric is an average, and thus a single figure, we present all metrics in bar graphs, these allow for easy comparison between single figures.

4.3.2 Flight separation. As stated in 2.2.1 LogUI uses flights to differentiate between different experimental setups. Each flight has a unique identifier that is added to each event that was fired from that flight, so we can easily separate all logged events. To accommodate the desire for comparison between experimental setups, shown in section 3, our dashboard displays all metrics separated by flight.

4.3.3 Filtered data download. While our dashboard is suited for an initial exploratory analysis of the data at hand, it could be the case that certain nuances in the data cannot be easily discovered using just the dashboard. A simple example is that there could be a single extreme outlier that changes the average, which would go unnoticed in the current bar graphs of the dashboard. To assist the researcher in the further analysis we provide a filtered download on each metric. This download is similar to the one already provided by LogUI but supplemented with the metric and flight filters that are present in our dashboard.

4.4 Implementation

We have chosen to implement the front-end of the dashboard as a web application, meaning it is written in HTML, CSS and JavaScript, for two primary reasons. First of all, it means the user is not required to install any extra software to use the data dashboard, which also made the user study easier to deploy. Secondly, the LogUI server application introduced in section 2.2.4 already has a web interface to configure the server, developing the dashboard as a web application means it will be easy to fully integrate the dashboard into the LogUI framework. As JavaScript does not come with a standard chart library, we used the ChartJs library to render all charts.

The logging data was stored in MongoDB, which is a popular document database. LogUI already uses MongoDB to store logging data, adapting the same database software for our dashboard again means that integrating the dashboard into LogUI would be straightforward. We retrieve the data from the database using a Node.js and Express based REST API, using Mongoose to talk to the MongoDB instance.

5 METHODOLOGY

To verify the usability and effectiveness of our dashboard we ran a small user study with a group of IIR researchers. They were given a prototype of our dashboard to verify if our proposed solution provides enough information to be used in research.

5.1 User study flow

Our user study was conducted over Zoom, a commercial video conferencing tool. At the start of each session, the participant was first told in short what the interview would encompass. They were then asked for their consent, after which they got a short introduction, which was a condensed version of the background section of this paper. The full user study consisted of the following steps:

(1) Shortly introduce the dashboard.
(2) Monitor the participant through Zoom screen sharing while they independently use the dashboard to run simple experiments.
(3) Let the participant fill the User Experience Questionnaire (UEQ) to obtain quantitative data about the user experience of our dashboard.
(4) Have a semi-structured interview to discuss their use of the dashboard.

5.2 Interaction data preparation

To use, discuss and review our dashboard in an interview it first needs to be populated with data gathered from logging user interactions with an application. We chose Google as an example application because first of all, it's a search engine, which fits our research area. But also because every participant would already be familiar with the application. This removed any trouble the participants could have with our dashboard that simply originated with them not being familiar with the application from which the logging data was sourced. We developed a simple clone ourselves as we obviously don't have the ability to add LogUI to the actual Google website. The events we logged on this application were:

- Click the search button.
- Type in the search bar.
- Click a search result.

We created two separate flights and ran ten different sessions in each flight. We made sure that we had a clear separation in data between the flights for each metric. This ensured participants could come to a clear conclusion for each of the given research questions. Beyond that we did not pay much attention to what we did exactly in each session, as the goal is to create a general dashboard design, meaning it should be usable regardless of how an experiment was run.

5.3 Individual experiments

To figure out the effectiveness and usability of the dashboard we gave the participants a set of research questions that could reasonably be formed around the data. They then had to form conclusions exclusively from the data that our dashboard could display. We observed our participants during this process through a Zoom call and recorded their screen so we could further analyze their use of the dashboard at a later time. The questions we posed were:

- Which experimental setup formed the longest queries?
- Which experimental setup took the longest to click a result after executing a query?
- Which experimental setup clicked the most search results after executing a query?
- Which experimental setup executed the most queries before clicking a search result?

We chose these questions based on the papers we analyzed when gathering the more frequently occurring metrics. Each of these four questions is taken from one of the papers that can be found in Table 1.

5.4 UEQ

To obtain some quantitative data about the quality of the user experience of our dashboard we let the participants fill the User Experience Questionnaire [17]. The UEQ consists of 26 opposing qualities where the participant is asked to rank on a scale of 1 to 7 how much the application matches that quality. For example, if the qualities are (pleasing - unpleasing), a participant would fill in 1 if they find the application very pleasing and 7 if it is unpleasing. The UEQ combines the scores of the 26 qualities into 6 categories:

- Attractiveness: *Overall impression of the product. Do users like or dislike it?*
- Perspicuity: *Is it easy to get familiar with the product and to learn how to use it?*
- Efficiency: *Can users solve their tasks without unnecessary effort? Does it react fast?*
- Dependability: *Does the user feel in control of the interaction? Is it secure and predictable?*

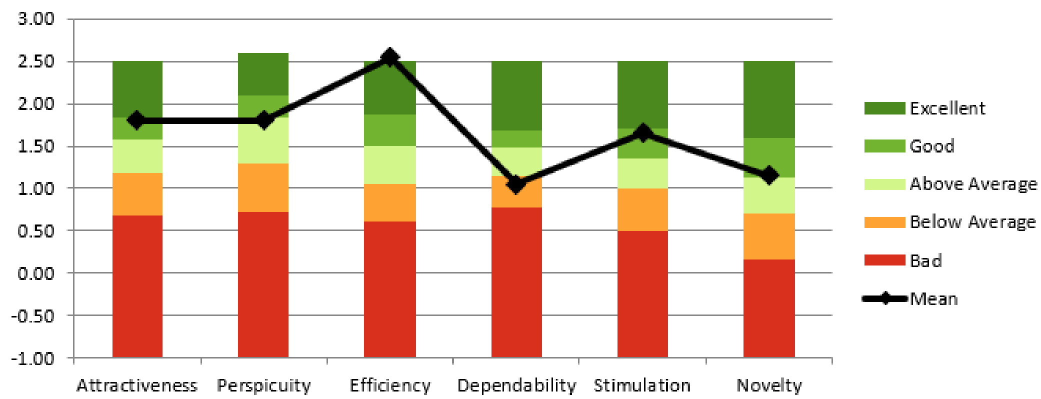

Figure 3: Results from the UEQ over five participants, displayed by category.

- Stimulation: *Is it exciting and motivating to use the product? Is it fun to use?*
- Novelty: *Is the design of the product creative? Does it catch the interest of users?*

This questionnaire comes with a data set containing data from 21175 persons from 468 studies concerning different products, which we can use as a benchmark for our application.

5.5 Interview

While the UEQ provides us with some feedback on the usability of our dashboard, it does not give much insight into the reasons behind a certain rating. To find what should be improved and how we held a semi-structured interview with our participants. In this interview we asked them the following questions:

(1) Do you find it easier to analyze your data using this dashboard compared to parsing analyzing the data yourself.
(2) Do you think the dashboard provides adequate information to perform an exploratory analysis on logging data?
(3) Is there anything you would change to make the dashboard better?

These questions were used as a guideline in the interview, we were not afraid to deviate from them whenever the participants had other points to talk about that they found important.

5.6 Participants

The goal of the experiment is to figure out if our data dashboard provides enough information and is easier to use compared to separately analyzing the data. This means our participants need to have at least some experience with running experiments on a user interface, preferably using LogUI. We recruited participants by consulting the developers of LogUI, through which we contacted students that have used LogUI in their research. We gathered five interview participants, which is less than we would like, which we will discuss further in section 7. All of the participants were well-versed in conducting user interface studies, two of them had experience with LogUI.

6 USER STUDY RESULTS

This section outlines the results we got from running our user study. We discuss how participants performed in the experiment. We show our score in each of the UEQ categories and compare this to a benchmark dataset and finally summarize the interviews we held.

6.1 UEQ Results

The results from the 26 questions in the UEQ were condensed to six core qualities as discussed in section 5.4. These results are shown in Figure 3, where the mean from the rankings given by our participants are shown among the benchmark bars. The scales for the benchmark are shown as follows: bad (among 25% of worst results), below average (between 25% and 50% of worst results), above average (between 25% and 50% of best results), good (between the top 10% and 25% of best results), excellent (among the top 10% of results).

Our application scored in the good category for most categories. The outliers are efficiency, where we score excellent, and dependability, where we score just below average. We hypothesize that we score high in efficiency because, for one, the dashboard responds very fast. On an AMD Ryzen 3600 desktop with 16GB of ram, it takes 1 millisecond to switch between two metric pages and about 300 milliseconds for the values of the metrics to be calculated whenever one of the filters changes. Additionally, we scored high on (cluttered - organized). This corresponded with what we heard in the interviews, as the participants mentioned the UI looked organized and they could easily find what they were looking for.

We scored low on dependability, which contains the following items; (unpredictable - predictable), (obstructive - supportive), (not secure - secure) and finally (does not meet expectations - meets expectations). From these items, the lowest scoring was (unpredictable - predictable), so this is where we find we need to focus our attention. The interview gave a slight insight into why some participants found our application unpredictable.

6.2 Experiment performance and interview

In section 5.3 we clarified how we let participants use our dashboard to answer a set of basic research questions, we will now discuss the results of this experiment. The average time it took for the participants to read and answer all four research questions was 5 minutes and 19 seconds. Of the five participants, only one was unable to find a clear answer to one of the research questions, which was the question "Which setup formed the longest search queries?". The participant later explained that he was unsure if the "type in search" event corresponded with the user typing a single character in the search box. We believe that this is not a problem with the dashboard itself, as the naming of these events is in the hands of the researcher. We find that we can reasonably conclude that a researcher would know to which event a certain event name corresponds if he named them himself.

The interview gave some assurances that the dashboard was satisfactory, but also gave a good set of features that we could improve upon. All of the participants answered "yes" to both question 1 and 2 from section 5.5. Which gave us some assurance that researchers see this dashboard as an improvement over parsing the data themselves. They found that the dashboard in its current state provides enough information to do an exploratory analysis, but also noted some improvements that they thought would be useful. We will further discuss some improvements that were suggested that we could implement at a later stage.

6.2.1 Unpredictable behaviour. There were some aspects of the UI of the dashboard that made the participants feel it was unpredictable. First of all, the metric that measures how often an event occurs in a session on average was displayed as "How many times an event occurs in a session". This metric was similarly worded to the metric "Time between an event in a session on average", while the first metric involves a quantity and the second metric reports a time span. This problem was reported by the first two participants. To solve this we reworded the first metric to "Average amount an event occurs in a session" and the second metric to "Average time between occurrences of an event". We made this change and this change only before we continued the user study with the next three participants. All of these three subsequent participants reported they found the application predictable. While the sample size is very low, this still gives us an indication that the change assisted in the predictability of the UI.

Secondly, whenever a metric required two events to be selected, we combined them in a single drop-down. This meant that for the metric "Average time between event X and event Y "we would show "Event X - Event Y" as an entry in the drop-down. Participants reported that they found this confusing, as it was not clear which event would go in which place in the metric. To improve this it was suggested that we simply create two drop-downs, one for each event. The header of the drop-down can then indicate in which of the variables in the metric the event would fit.

6.2.2 Probability distribution. Four of the five participants recommended that whenever we display a mean, we should visualize the distribution of the data. An insight into the distribution can for example show if there is an extreme outlier that skews the mean. A solution would be to display a box plot instead of a simple bar graph. This would still allow for easy comparison between flights but would also visualize the distribution without the UI becoming much more cluttered.

6.2.3 Individual users. Some participants noted that it would be useful to be able to see the data of individual users in table form. This again could be especially useful if there are some extreme outliers in the data. Being able to see those individually could give a greater insight into how those extremes came to be.

6.2.4 Improved filtering. It was noted that, while we already provide some filtering, it could be improved. LogUI allows researchers to define custom application-specific properties, for example, if you want to further differentiate users within a flight for whatever reason. Adding filters for these custom properties would be very useful in that case.

7 CONCLUSIONS AND FUTURE WORK

We have described our proposed data dashboard, which has been shown to be able to visualize logging data for exploratory analysis. The metrics we gathered from existing papers have allowed us to show which behavioural metrics IIR researchers generally look for when performing user interface experiments. These metrics ensure that we display the data that is most commonly required in the IIR community. Our user study has given us the impression that the dashboard is suited to perform research. The participants were able to answer the given research questions in a little more than 5 minutes, which we find is rather quick. In the future, it would be useful for us to run an additional user study where we ask half of the participants to answer the same research questions while having to manually parse the same data, so we can make a more accurate comparison. The UEQ has given us the idea that the dashboard has a satisfactory user interface. Together, we find this shows that our solution to RQ2 is compelling, however, there is still some future work to be done.

We find one of the main problems with our research is that we conducted the user study with only five participants, due to time and availability constraints. This means We got a broad idea about what researchers think of our solution, but not enough to perform actual statistical analysis on the data or form more concrete conclusions. In the future, we would like to perform the same user study with more participants to get more accurate results. Not only could we then take researchers from the IIR field, but also expand our research to other fields where user interface logging is prevalent as well. This means we would have to expand our research into the most common metrics into these other fields as well.

As mentioned in section 6.2, there are several suggested features that we could implement into the dashboard. We also would like to integrate the dashboard into the existing LogUI framework, but this would need to be discussed with the LogUI developers. All in all, we find that we have provided a useful data dashboard on which we aim to continue development to provide more advanced features in the future.

REFERENCES

[1] [n.d.]. Document Object Model (DOM) - Web APIs: MDN. https://developer.mozilla.org/en-US/docs/Web/API/Document_Object_Model

[2] [n.d.]. Webtechnologie voor ontwikkelaars. https://developer.mozilla.org/nl/docs/Web/Events

[3] Afrah Olayan Alanazi, Mark Sanderson, Zhifeng Bao, and Jaewon Kim. 2020. The impact of ad quality and position on Mobile SERPs. In *Proceedings of the 2020 Conference on Human Information Interaction and Retrieval*. 318–322.

[4] Jason Alexander, Andy Cockburn, and Richard Lobb. 2008. AppMonitor: A tool for recording user actions in unmodified Windows applications. *Behavior Research Methods* 40, 2 (2008), 413–421.

[5] Michael Alexander and John Walkenbach. 2013. Excel dashboards and reports. (2013).

[6] Richard Atterer. 2006. Logging usage of AJAX applications with the" UsaProxy" HTTP proxy. In *Proceedings of the WWW 2006 Workshop on Logging Traces of Web Activity: The Mechanics of Data Collection*.

[7] Leif Azzopardi, Myles Doolan, and Richard Glassey. 2012. ALF: a client side logger and server for capturing user interactions in web applications. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*. 1003–1003.

[8] Leif Azzopardi, Diane Kelly, and Kathy Brennan. 2013. How query cost affects search behavior. In *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval*. 23–32.

[9] Ralf Bierig, Jacek Gwizdka, and Michael J Cole. 2009. A user-centered experiment and logging framework for interactive information retrieval. In *UIIR@ SIGIR*.

[10] Tessel Bogaard, Laura Hollink, Jan Wielemaker, Lynda Hardman, and Jacco Van Ossenbruggen. 2019. Searching for old news: User interests and behavior within a national collection. In *Proceedings of the 2019 Conference on Human Information Interaction and Retrieval*. 113–121.

[11] Horatiu Bota, Adam Fourney, Susan T Dumais, Tomasz L Religa, and Robert Rounthwaite. 2018. Characterizing search behavior in productivity software. In *Proceedings of the 2018 Conference on Human Information Interaction & Retrieval*. 160–169.

[12] Mon Chu Chen, John R Anderson, and Myeong Ho Sohn. 2001. What can a mouse cursor tell us more? Correlation of eye/mouse movements on web browsing. In *CHI'01 extended abstracts on Human factors in computing systems*. 281–282.

[13] Bogeum Choi, Robert Capra, and Jaime Arguello. 2019. The effects of working memory during search tasks of varying complexity. In *Proceedings of the 2019 Conference on Human Information Interaction and Retrieval*. 261–265.

[14] Claudia Hauff David Maxwell. 2021. LogUI : Contemporary Logging Infrastructure for Web-Base Experiments. (2021).

[15] Andrea Janes, Alberto Sillitti, and Giancarlo Succi. 2013. Effective dashboard design. *Cutter IT Journal* 26, 1 (2013), 17–24.

[16] Bernard J Jansen, Raghavan Ramadoss, Mimi Zhang, and Nan Zang. 2006. Wrapper: An application for evaluating exploratory searching outside of the lab. *EESS 2006* 14 (2006).

[17] Bettina Laugwitz, Theo Held, and Martin Schrepp. 2008. Construction and evaluation of a user experience questionnaire. In *Symposium of the Austrian HCI and usability engineering group*. Springer, 63–76.

[18] Jiqun Liu, Matthew Mitsui, Nicholas J Belkin, and Chirag Shah. 2019. Task, information seeking intentions, and user behavior: Toward a multi-level understanding of Web search. In *Proceedings of the 2019 conference on human information interaction and retrieval*. 123–132.

[19] Jiqun Liu and Chirag Shah. 2019. Investigating the impacts of expectation disconfirmation on web search. In *Proceedings of the 2019 Conference on Human Information Interaction and Retrieval*. 319–323.

[20] Gavin McArdle and Rob Kitchin. 2016. The Dublin Dashboard: Design and development of a real-time analytical urban dashboard. (2016).

[21] Kevin Ong, Kalervo Järvelin, Mark Sanderson, and Falk Scholer. 2018. Qwerty: The effects of typing on web search behavior. In *Proceedings of the 2018 Conference on Human Information Interaction & Retrieval*. 281–284.

[22] Manasa Rath, Souvick Ghosh, and Chirag Shah. 2018. Exploring online and offline search behavior based on the varying task complexity. In *Proceedings of the 2018 Conference on Human Information Interaction & Retrieval*. 285–288.

[23] Ian Ruthven. 2008. Interactive information retrieval. *Annual review of information science and technology* 42, 1 (2008), 43–91.

[24] Veronica S Smith. 2013. Data dashboard as evaluation and research communication tool. *New directions for evaluation* 2013, 140 (2013), 21–45.

[25] Jaime Solís-Martínez, Jordan Pascual Espada, Rubén González Crespo, B Cristina Pelayo G-Bustelo, and Juan Manuel Cueva Lovelle. 2020. UXJs: Tracking and Analyzing Web Usage Information With a Javascript Oriented Approach. *IEEE Access* 8 (2020), 43725–43735.

[26] Robert J Thierauf. 1991. *Executive information systems: a guide for senior management and MIS professionals*. Quorum Books.

[27] Anne Marie JWM Weggelaar-Jansen, Damien SE Broekharst, and Martine De Bruijne. 2018. Developing a hospital-wide quality and safety dashboard: a qualitative research study. *BMJ quality & safety* 27, 12 (2018), 1000–1007.

[28] Wan-Ching Wu, Diane Kelly, and Avneesh Sud. 2014. Using information scent and need for cognition to understand online search behavior. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*. 557–566.

[29] Haotian Zhang, Mustafa Abualsaud, and Mark D Smucker. 2018. A study of immediate requery behavior in search. In *Proceedings of the 2018 Conference on Human Information Interaction & Retrieval*. 181–190.