Web User Interaction Speed Study

Leonardo Germán Loza Bonora
LIFIA, Fac. de Informática, Univ. Nac. de La Plata
La Plata, Argentina
lloza@lifia.info.unlp.edu.ar

Julián Grigera
LIFIA, Fac. de Informática, Univ. Nac. de La Plata
CONICET / CICPBA
La Plata, Argentina
juliang@lifia.info.unlp.edu.ar

Juan Cruz Gardey
LIFIA, Fac. de Informática, Univ. Nac. de La Plata
CONICET
La Plata, Argentina
jcgardey@lifia.info.unlp.edu.ar

Alejandra Garrido
LIFIA, Fac. de Informática, Univ. Nac. de La Plata
CONICET
La Plata, Argentina
garrido@lifia.info.unlp.edu.ar

***Abstract*—Web applications often suffer from interaction problems, causing users to struggle to fill out forms or find information. To diagnose these problems without incurring large costs, a popular solution is to analyze interaction events automatically to find problematic behavior patterns. However, these analyses often do not take into account the user's interaction style, such as speed. In this paper we present a study on the web users' speed by using recordings of real sessions to find how early this characteristic, measured in terms of mouse, scroll and keyboard speed, can be estimated. Preliminary results suggest that it takes approximately 50 seconds or 500 events to obtain a close estimate (>0.85) of the benchmark speed, calculated from the total number of events in the session.**

***Index Terms*—interaction, web interfaces, log analysis**

I. Introducción

La evaluación de interacción de usuario en la web se utiliza para muchos fines, como descubrir pr oblemas de usabilidad o accesibilidad. Con solo observar cómo utilizan los usuarios diferentes dispositivos estándar, como mouse y teclado, se pueden obtener diagnósticos que ayuden a mejorar la interacción.

Este tipo de evaluaciones, sin embargo, suelen requerir un número considerable de recursos: por ejemplo, una prueba de usuario (la técnica más utilizada para evaluar interacción) requiere de voluntarios y profesionales especializados para analizar su comportamiento, además del tiempo. Para reducir estos tiempos y costos, han surgido varias soluciones que proponen automatizar el análisis de interacción mediante la captura de registros (logs) de interacción, para diferentes áreas como la usabilidad [1] o accesibilidad [2].

Muchos de los trabajos que proponen capturar y analizar interacción de manera automatizada no tienen en cuenta las características de los usuarios que generan los logs. Esto lleva a interpretar de manera uniforme el comportamiento de usuarios que tienen diferentes características, como velocidades de mouse o teclado, o tiempos de espera. Además, en muchos casos, los logs se capturan de forma anónima, con lo cual se suma el desafío de no conocer al usuario previamente [3], [4].

Nuestra hipótesis sostiene que conocer el perfil de interacción de usuario ayuda a mejorar la calidad del análisis automatizado de logs. Por ejemplo, si una persona es relativamente lenta al mover el cursor del mouse, un repentino movimiento veloz podría indicar un problema [5]. Sin embargo, para un usuario que siempre mueve el cursor velozmente, el mismo tipo de movimiento rápido no sería de especial interés.

De las distintas maneras y métricas para caracterizar perfiles de interacción de usuario, la velocidad, tanto de mouse como de teclado es un factor presente en la mayoría de los trabajos relacionados, por lo cual en el presente trabajo buscamos determinar cuán tempranamente se puede estimar la velocidad de interacción del usuario web. Para esto, exploramos dos criterios: tiempo de uso (en cuánto tiempo podemos estimar la velocidad), y número de eventos (cuántos eventos necesitamos para estimarla). En un primer estudio, se dispuso una base de datos con grabaciones de sesiones de interacción de usuarios reales siguiendo tareas predefinidas. Utilizando como referencia la velocidad de interacción calculada con el total de los eventos, analizamos cuán tempranamente puede estimarse según dos criterios: tiempo transcurrido y número de eventos.

II. Trabajo relacionado

Diversos autores han trabajado en la caracterización de usuarios a partir de registros de interacción, especialmente en el área de aplicaciones de escritorio. Sin embargo, se considera la habilidad del usuario en programas que están orientados a realizar tareas específicas y no una interacción general. Entre los trabajos sobre el tema, Hurst et al. [6] proponen un método para detectar la experiencia de usuario de forma dinámica y poder clasificar si este es hábil en una herramienta particular, en este caso, de edición de imágenes. En un trabajo orientado a interfaces adaptativas en aplicaciones de escritorio, Ghazarian et al. [7] presentaron métodos y

978-1-6654-5674-6/22/$31.00 ©2022 IEEE

experimentos para construir clasificadores de habilidad automáticos para adaptar las interfaces de usuarios. Los mismos autores realizaron posteriormente un estudio sobre el vínculo entre las pausas en las interacciones de los usuarios con las aplicaciones y sus niveles de habilidad, utilizando algoritmos de aprendizaje automático para crear clasificadores con los atributos capturados [8]. Combinando movimientos de mouse y el análisis de pausas, en un trabajo más reciente de Attig et al. [9] se probó detectar la competencia y la performance de los usuarios durante el uso de pruebas en el software de estadística SPSS. Grossman et al. [10] exploraron la viabilidad de inferir automáticamente niveles de experiencia de usuario según sus patrones de uso in situ sobre AutoCAD, hallando una correlación entre el rendimiento en comandos individuales y los tiempos de finalización de estos. Si bien estos trabajos se enfocan en aplicaciones particulares, y en el contexto de escritorio, brindan un buen punto de partida para la selección de eventos a los cuales prestar atención y en cómo capturarlos. También indican que las métricas pueden cambiar según el contexto estudiado.

Por otra parte, se hallaron estudios que tienen por objetivo analizar la interacción en ambientes web. Por ejemplo, Huang et al. [11] examinan el comportamiento del cursor del mouse en páginas de resultados de motores de búsqueda (SERPs), observando clicks, movimiento y hover. Lagun et al. [12] lograron capturar automáticamente características claves del comportamiento de movimiento de mouse mediante la búsqueda de subsecuencias. Kirsh [13] llevó a cabo un estudio que se centró en las direcciones del movimiento del mouse y velocidades, y qué indican estos más que la posición del mouse. Speicher et al. [3] crearon un pipeline para seguir interacciones de cursor en el cliente, analizarlas y aprender con base en modelos de relevancia. Diriye et al. [14] estudiaron el motivo de abandono de páginas de búsqueda mediante predicción automática, utilizando la interacción con los resultados (movimiento de cursor, desplazamiento y clicks) y la sesión de búsqueda completa. Aunque estos trabajos se enfoquen en la web, están centrados estudiar la importancia de los eventos en sí, o en analizar el comportamiento del usuario para predecir que va a hacer. Sin embargo, esto nos ayuda a discernir qué eventos escuchar, y de qué forma, para lograr su predicción en el menor tiempo o cantidad posibles.

III. Metodología

Con el fin de hallar cuán tempranamente se puede obtener una estimación de la velocidad de interacción de usuarios web, fue realizado un análisis de muestras reales. Para esto, fue utilizada una base de datos existente con capturas de interacción de usuarios completando tareas específicas sobre aplicaciones web. Sobre la misma se realizaron tareas de preparación de datos, y fue estudiada la velocidad de interacción respecto del número de eventos y del tiempo transcurrido.

A. Elección de métricas

Decidimos centrarnos en obtener los promedios de las velocidades más utilizadas e influyentes en los estudios citados: velocidad de mouse, scroll e inter-tecla. Para las dos primeras utilizamos como unidad *píxeles sobre milisegundos*, mientras que para la última utilizamos *caracteres por segundo*.

Otro factor por el cual las métricas fueron seleccionadas es por su facilidad y velocidad de captura. Al ser calculadas a partir de eventos simples de bajo nivel, no conlleva un procesamiento complejo conseguirlos para hacer estimaciones en tiempo real de comportamiento.

Fue definida una métrica llamada Velocidad Compuesta de Interacción (VCI), que consiste en sumar las 3 velocidades estudiadas: mouse, scroll, e inter-tecla. Dado que las unidades de cada velocidad son diferentes, para poder sumarlas utilizamos su relación con la velocidad *final*, es decir, la velocidad calculada a partir de la totalidad de los eventos de la sesión. Por ejemplo, si en una sesión de 10' se recorren 5000 pixels con el *scroll*, la velocidad *final* será de 5000/600 *pixels / segundo*. Si tomáramos la velocidad a los 5 minutos, y observáramos que se recorrieron 1000 pixels, podríamos decir que la velocidad en la mitad de la sesión sería de 1000/300 *pixels / segundo*. Para obtener una unidad compatible con las otras, la velocidad del corte fue divida por la velocidad final, obteniéndose la relación de cercanía con ésta. En el ejemplo citado, sería 0.4 (1000/300 / 5000/600). Este mismo proceso se realiza con las otras dos velocidades, y permite realizar la sumatoria con una unidad de medida uniforme. Finalmente, se divide por 3 (el número de velocidades) para estandarizarlo alrededor del número 1, que representa la velocidad de referencia. Cuando más se acerca al 1, más se acerca a la velocidad de referencia, tanto por por encima (>1) o por debajo (<1).

B. Preparación de los datos

La base de datos utilizada contiene 191 grabaciones de sesiones de uso de aplicaciones web de compras y formularios de otros tipos de sitios (ej. gubernamentales). Los usuarios reclutados que siguieron tareas en dichos sitios que generalmente consistían en realizar compras o completar formularios de registro. Las sesiones tienen duraciones de entre 18 segundos para los formularios más breves hasta 826 segundos los más extensos.

Para calcular las métricas propuestas tuvo que realizarse un trabajo de preparación, ya que la herramienta que utilizada para capturar (rrweb-io[1]) está optimizada para almacenar y reproducir *screencasts*. Por este motivo, no almacena eventos JavaScript nativos, sino que define una forma especial para reproducir los eventos de manera visual y programática, y así mostrar en pantalla de una manera comprensible las interacciones de cada usuario. Esto causa que los tipos guardados y sus valores no correspondan con los eventos del navegador, lo que también produce una pérdida de información.

La base consta de 2 documentos en formato JSON:

- Screencast: posee metadatos de cada muestra capturada. Consta de un identificador, nombre, marca de tiempo de generación y una lista de identificadores de eventos.

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

- ScreencastEvent: posee información sobre cada evento. Consta de un identificador, marca de tiempo de generación, un tipo (representado por un número entero) y un registro con valores (cuya estructura interna depende de su tipo).

Se realizaron consultas utilizando el Aggregation Framework de MongoDB[2] para reconstruir los eventos de la herramienta a una forma similar a los de JavaScript, y con éstos computar las velocidades. Esto permite que cada consulta funcione con futuras bases de datos mientras cada evento respete su formato original.

La estructura del tipo de los eventos de mouse y scroll de la herramienta de captura era similar a la de JavaScript. Solo constaba de un nombre de tipo diferente y una menor cantidad de datos, pero con los necesarios para realizar los cálculos deseados: posiciones en el eje x y en el eje y en la pantalla visible del navegador y el tiempo de generación del evento.

Para calcular la velocidad promedio del mouse por muestra, fue realizada la suma de las distancias euclídeas entre los puntos que representan la posición del mouse en eventos de movimiento de mouse ordenados por tiempo. Este resultado fue dividido por el tiempo transcurrido de la muestra. La velocidad del scroll fue calculada de igual manera, simplificando la operación de cálculo de la distancia por trabajar únicamente con el eje y del navegador.

El cálculo de la velocidad inter-tecla acarreó complicaciones. La librería utilizada para la captura no guarda eventos de pulsaciones de teclado, sino las cadenas de caracteres escritas en una misma etiqueta HTML <label> de tipo texto con su tiempo de creación. Esto causó que haya bloques de texto en los cuales no se diferenciaban de manera clara las palabras escritas entre ellas, ni borrado de caracteres. Como era necesario calcular solo el tiempo de escritura, hubo que delimitar cuáles de estas cadenas pertenecían a una misma palabra, y así conseguir el tiempo de pulsación entre caracteres.

La delimitación se logró calculando la distancia de Levenshtein entre dos cadenas de texto subsiguientes ordenadas por tiempo. Si ambas cadenas tenían una relación del 66%, se consideraba que pertenecían a la misma palabra, agregándose un evento falso "inputEnd" entre cada palabra para delimitar su final. Este número se obtuvo empíricamente, a través de la observación del acierto producido con las muestras.

Para obtener la velocidad inter-tecla promedio por muestra, fue calculada la diferencia del tiempo entre eventos adyacentes que sean ambos del tipo "input", quedando como resultado caracter escrito por segundo. Luego todos los resultados fueron sumados y el promedio buscado fue calculado.

C. Cálculo de estimaciones de velocidad

Para cada muestra fueron realizados múltiples cortes por criterio de cantidad de eventos (cada 100 eventos) o por tiempo transcurrido (cada 5 segundos). En cada corte fue calculada la VCI. Finalmente, fue calculado el promedio de la estimación por corte de muestra, su desviación estándar y la cantidad de muestras que representan estos valores.

[2]https://www.mongodb.com/developer/products/mongodb/aggregation-framework/

IV. Resultados

Los resultados respecto del número de eventos y tiempo transcurrido pueden observarse en las Figuras 1 y 2 respectivamente. Es importante notar que las muestras tienen diferentes duraciones, ya que cada una es una grabación de un usuario completando una tarea. Por esto la curva "Cantidad de muestras" denota el porcentaje de muestras que contienen como mínimo el número de eventos (o tiempo) indicado en el eje x. Naturalmente, en el margen izquierdo, esta curva comienza en 1, dado que todas las muestras tienen como mínimo 0 eventos y duran 0 segundos. A medida que avanzamos sobre el eje, este porcentaje comienza a bajar. En ambos casos, nuestro foco está en hallar el momento más temprano que cumpla con 3 condiciones: que la velocidad combinada se acerque a la de referencia, que la desviación sea baja, y que el número de muestras de las que surgen estos números sea alto (lo que indica que hay más evidencia soportando la estimación). En las siguientes subsecciones se muestra el momento óptimo según las muestras analizadas.

A. Estimación según cantidad de eventos

En la figura 1 se ve que, si bien cerca de los 300 eventos la VCI se estabiliza cerca del valor de referencia (representado por 1), a partir de los 500 eventos la desviación estándar baja considerablemente. En los cortes siguientes ésta sigue disminuyendo, pero también disminuye la cantidad de muestras que contienen el número de eventos indicado en el eje x, por lo cual el corte de 500 eventos es un buen equilibrio par una estimación temprana, cercana a 1, con baja desviación, y un número considerable de muestras que la respaldan.

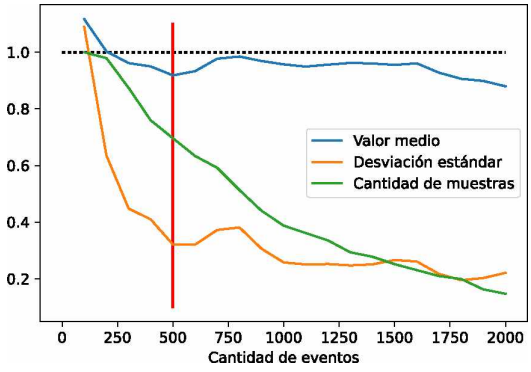

Fig. 1. VCI respecto del nro. de eventos.

B. Estimación según tiempo transcurrido

Los resultados según el tiempo de muestra se ven en la Figura 2. En este caso, es notable el momento alrededor de los 50 segundos. También sucede algo parecido al análisis

previo: la estimación se estabiliza cerca de 1 relativamente pronto (cerca de los 20 segundos), pero la desviación tarda un poco más en bajar, por lo que destacamos el momento cercano a los 50 segundos donde esto sucede. Además, el número de muestras que llegan al menos a este tiempo sigue siendo alto.

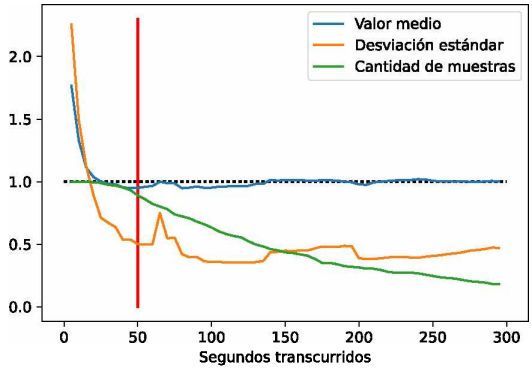

Fig. 2. VCI respecto del tiempo transcurrido.

V. Sesgos y limitaciones

El estudio presentado podría tener algunos sesgos, sobre todo en las muestras utilizadas y su captura. En primer lugar, es importante destacar que las sesiones tienen diferentes duraciones y números de eventos. Esto lleva a que el cálculo de la velocidad tomado como referencia pueda tener diferentes precisiones en diferentes usuarios. Se buscó mitigar este sesgo incluyendo en los gráficos el número de muestras que alcanza la cantidad de eventos o de tiempo indicado en el eje x. Por otra parte por más extensa que sea una muestra, siguen siendo sesiones relativamente breves, con lo cual es posible que la velocidad final calculada no sea un indicador muy fiel de la velocidad del usuario frecuente. Otro posible sesgo es el sesgo de completitud, que indica que un usuario al que se le pide que complete una tarea, se vea presionado a finalizarla. Esto puede llevar a una mayor eficacia que en condiciones normales. Si bien no se ha podido mitigar estos dos últimos sesgos por haber utilizado muestras ya tomadas, se está planificando un nuevo estudio teniendo esto en cuenta, que se describe en la siguiente sección.

VI. Conclusiones y trabajo futuro

De los dos análisis realizados, los resultados sugieren que el tiempo transcurrido es un mejor indicador que la cantidad de eventos, ya que ambos llegan en algún momento determinado a una estimación cercana a uno, pero solo en los cortes de tiempo va disminuyendo de manera significativa la desviación estándar.

Las muestras utilizadas eran de diversos tipos de páginas web y los tiempos totales de interacción entre ellas diferían por un gran margen. Quizás trabajar con muestras más uniformes o localizadas pueda ayudar conseguir valores más constantes y según tipo de interacción.

Como trabajo futuro planificamos trabajar con otras formas de acumular resultados. Dado que la sumatoria dio resultados interesantes, se podría hacer una nueva, pero teniendo en cuenta los pesos de importancia de cada tipo de evento que compone la métrica.

Para tener una mejor referencia de velocidad, y al mismo tiempo evitar el sesgo de completitud, estamos organizando un estudio complementario con voluntarios que utilizarán la herramienta por tiempos prolongados. Esto, por un lado, nos dará una caracterización más precisa, y por el otro, al no tener consignas concretas, esperamos que los usuarios actúen de manera más natural.

References

[1] T. Carta, F. Paternò, and V. F. d. Santana, "Web usability probe: a tool for supporting remote usability evaluation of web sites," in *IFIP Conference on Human-Computer Interaction*, pp. 349–357, Springer, 2011.

[2] H. Kacorri, S. Mascetti, A. Gerino, D. Ahmetovic, H. Takagi, and C. Asakawa, "Supporting orientation of people with visual impairment: Analysis of large scale usage data," in *Proceedings of the 18th International ACM SIGACCESS Conference on Computers and Accessibility*, pp. 151–159, 2016.

[3] M. Speicher, A. Both, and M. Gaedke, "Tellmyrelevance! predicting the relevance of web search results from cursor interactions," in *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*, pp. 1281–1290, 2013.

[4] J. Grigera, A. Garrido, J. M. Rivero, and G. Rossi, "Automatic detection of usability smells in web applications," *International Journal of Human-Computer Studies*, vol. 97, pp. 129–148, 2017.

[5] J. Hernandez, P. Paredes, A. Roseway, and M. Czerwinski, "Under pressure: sensing stress of computer users," in *Proceedings of the SIGCHI conference on Human factors in computing systems*, pp. 51–60, 2014.

[6] A. Hurst, S. E. Hudson, and J. Mankoff, "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, 2007.

[7] A. Ghazarian and S. M. Noorhosseini, "Automatic detection of users' skill levels using high-frequency user interface events," *User Modeling and User-Adapted Interaction*, vol. 20, no. 2, pp. 109–146, 2010.

[8] A. Ghazarian and A. Ghazarian, "Pauses in man-machine interactions: a clue to users' skill levels and their user interface requirements," *International Journal of Cognitive Performance Support*, vol. 1, no. 1, pp. 82–102, 2013.

[9] C. Attig, E. Then, and J. F. Krems, "Show me how you click, and i'll tell you what you can: Predicting user competence and performance by mouse interaction parameters," in *International Conference on Intelligent Human Systems Integration*, pp. 801–806, Springer, 2019.

[10] T. Grossman and G. Fitzmaurice, "An investigation of metrics for the in situ detection of software expertise," *Human–Computer Interaction*, vol. 30, no. 1, pp. 64–102, 2015.

[11] J. Huang, R. W. White, and S. Dumais, "No clicks, no problem: using cursor movements to understand and improve search," in *Proceedings of the SIGCHI conference on human factors in computing systems*, pp. 1225–1234, 2011.

[12] D. Lagun, M. Ageev, Q. Guo, and E. Agichtein, "Discovering common motifs in cursor movement data for improving web search," in *Proceedings of the 7th ACM international conference on Web search and data mining*, pp. 183–192, 2014.

[13] I. Kirsh, "Directions and speeds of mouse movements on a website and reading patterns: a web usage mining case study," in *Proceedings of the 10th International Conference on Web Intelligence, Mining and Semantics*, pp. 129–138, 2020.

[14] A. Diriye, R. White, G. Buscher, and S. Dumais, "Leaving so soon? understanding and predicting web search abandonment rationales," in *Proceedings of the 21st ACM international conference on Information and knowledge management*, pp. 1025–1034, 2012.