Making Memories: Applying User Input Logs to Interface Design and Evaluation

Tamara Babaian
Computer Information Systems
Department
Bentley College
Waltham, MA 02452-4705
tbabaian@bentley.edu

Wendy Lucas
Computer Information Systems
Department
Bentley College
Waltham, MA 02452-4705
wlucas@bentley.edu

Heikki Topi
Computer Information Systems
Department
Bentley College
Waltham, MA 02452-4705
htopi@bentley.edu

CHI 2006, April 22–27, 2006, Montréal, Québec, Canada.
ACM 1-59593-298-4/06/0004.

Abstract
In this paper, we describe our approach to designing interface components that automate the logging of user input. These recorded logs of user-system interactions can serve as a basis for usability assessment, and we present here the usability measures that can be automatically derived from this logged data. Making user logs an integral component of the system data model extends their usefulness beyond providing information on user behavior. In our prototype, logs are used for creating a more collaborative interface by increasing the system's contextual awareness of user interactions.

Keywords
Human computer interaction, usability, collaborative interfaces.

ACM Classification Keywords
H5.2. Information interfaces and presentation.

Introduction
The availability of web site usage logs has given rise to many exciting developments in the area of interface personalization. The data provided by these records is used to dynamically adapt a web page's content or presentation style to individual users. Such adaptations

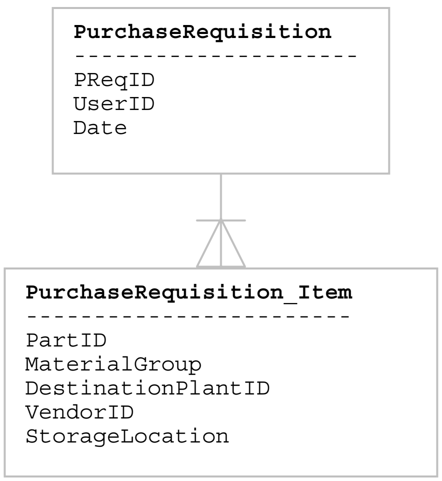

figure 1. Structure of the **PurchaseRequisition** and **PurchaseRequisition _Item** records demonstrates the various data items that must be entered while creating a purchase requisition.

can be based solely on the user's own history of interaction with the site or on that history in conjunction with the collective experience of all users (for examples, see [5,7]). Typically, Machine Learning or Data Mining techniques are used for detecting users' behavioral patterns, characteristics, and preferences.

Similar techniques have had a great impact on widely-used non-web-based software (e.g. desktop editors, administrative applications, etc.), which now commonly include personalizable features. Some systems also adapt their behavior to better suit the goals and experiences of a particular user ([4]). However, as pointed out by Webb et al. in [5], the broader application of Machine Learning to user modeling and user adapted interaction requires large sets of data on system usage, which are typically not available.

We recognize the great value that the logs of user interactions with the system can offer beyond their use in user modeling, as they provide a basis for usability evaluation and for making general improvements to the interfaces. They can also provide valuable information for user training and serve as a foundation for enhanced system capabilities. In this paper, we explain how user input logs are used in our experimental prototype, which is described in the next section. All user inputs to this prototype are automatically recorded by including the necessary functionality in every control with which a user can interact.

In our design, the input logs are an integral part of the data model. As a result, they are accessible to all components of the user interface, thus enabling it to take into account previous as well as on-going interactions with the user. By augmenting the system with this kind of "memory," we contend that it will have a greater capacity for tailoring its behavior to the individual users and their tasks. In addition, this memory will provide a means for continuous usability evaluation of the system's interfaces.

Application

We have implemented user logging in an experimental interface designed to prototype a collaborative approach to user-system interaction with an enterprise information system ([1]). Such a system integrates the business functions of a company (including Human Resources, Accounting, Inventory, etc.) within a single software package. Multiple functional modules are combined to automate management of a company's resources across the entire enterprise.

Our prototype implements an experimental interface for the typical organizational task of creating a purchase requisition, which consists of composing a specification of items that must be purchased (see figure 1 for an illustration). As is characteristic for enterprise-wide information systems tasks, a purchase requisition integrates data from a variety of enterprise system components. Thus, the values of most purchase requisition parameters can be entered using a search and selection mechanism embedded within the system. However, as we have discovered in our earlier studies [6], the amount and complexity of the underlying enterprise data coupled with the complexity of the search interfaces often overwhelms users and interferes with their effective use of the system. For example, the task of entering a set of values into the input fields of a purchase requisition does not sound as if it would be particularly difficult. The system-imposed constraints on the consistency of data across multiple modules, however, complicate matters, because the data must be entered in the exact form and sequence prescribed by the system.

Creation of a purchase requisition occasionally requires the user to perform *related* tasks, such as adding the specification of a part to a master list or adding a new vendor record. Our studies [6] have revealed that users often struggle with navigating the complex maze of screens and menus from unfamiliar parts of the system that are required for completing related tasks. Their prior experience with the system is often not sufficient for helping them perform new tasks.

The goal behind our experimental prototype is to demonstrate how usability can be enhanced by designing a system that exhibits more collaborative behavior in support of its users ([1]). In our interface, a greater degree of collaboration is achieved by improving the mechanisms by which the system communicates information about its data and processes and provides guidance to the user. The availability of user interaction logs means that the system is better equipped to reason about the overall context of a task and to have greater insight into what data or processes are relevant to a specific user and task.

In the next two sections, we describe the structure of the stored user interactions and show how they are used to (1) assess the usability properties of the interface and (2) provide a greater degree of collaborative support to the user.

User input data model

To record the details of a user's interaction with the system, we have extended interface components, henceforth referred to as widgets (implemented using the Java Swing library), to log all input from the user. The user enters data and navigates the system via these widgets, and all of the interactions are recorded in a **UserEntry** table, the fields of which are shown in figure 2.

```
UserEntry
------------------
UserID
WidgetID
TimeFocusGained
TimeFocusLost
EntryData
ContentOnLoseFocus
```

figure 2. Fields of **UserEntry** record the data entered by the user (EntryData) into a widget with a given WidgetID during the time period when the widget was in focus (between TimeFocusGained and TimeFocusLost). ContentOnLoseFocus stores the content of the input field at the time the focus was lost.

Each UserEntry record contains an EntryData field, which stores all user input directed to the widget during the time when it was in focus. This input can be entered via keystrokes, mouse button events, or by a value transferred into an input field from a selection mechanism, such as a search. The beginning and end of each time interval during which the widget was in focus is recorded, along with the final content of the input field at the time the focus was lost. Time stamping the gain and loss of focus enables a precise reconstruction of the sequence of recorded user actions from the start to the completion of the work performed on the task. We also define the *duration* of each UserEntry as the difference between TimeFocusLost and TimeFocusGained.

Uses of the collected data

The following information can be derived *automatically* from the user input logs:

(1) *An estimate of the time it took to complete a particular part of the task.* By logging all user entries with a reference to a particular widget to which the input was directed, we can measure the lower and upper bounds on the time it took to complete user entries to *any* group of widgets, with data entry possibly spanning multiple screens. The lower bound of the time estimate is computed as the total sum of all UserEntry durations. The upper bound is equal to the difference between the latest and earliest time stamps of UserEntry records across all of the widgets in the group.

(2) *An estimate of a number of changes to the input made before the final input was entered.* This can be measured by counting the number of disjoint sequences of deletion keystrokes. For example, a sequence of keystrokes "23<bk><bk>5", where <bk> refers to the Backspace character, produces a *number of changes measure* of one. Note that this way of measuring the number of different times the value entered into a field is modified is not precise. For example, the same change of value from 23 to 5 could have been achieved through the sequence of keystrokes "23<bk>←<del>5", which would be treated as containing two modifications rather than one.

(3) *Different methods utilized for input in a particular field (direct typing vs. cut and paste vs. search and select).* This is accomplished by using a special vocabulary of symbols, which differentiates between input methods, to specify the content of the EntryData field.

(4) *Frequency of usage of a particular part of a system.* Like measure (1), it can be evaluated at various levels of detail: from using one specific widget to using a complete task interface.

There are several benefits for the design and evaluation of user interfaces that arise from keeping records of user input collected in the course of system usage. Measures (1) through (4), along with other information that can be derived from the logs, can be used for multiple purposes, including making usability assessments and creating additional system capabilities, as described next.

Using input logs for usability assessment

The ISO [3] defines usability as the *effectiveness*, *efficiency*, and *satisfaction* with which specified users achieve specified goals using a particular interface. *Effectiveness* refers to the accuracy and completeness with which the users achieve their goals. *Efficiency* refers to resources expended in relation to the accuracy and completeness of goals achieved, and *satisfaction* denotes the comfort and acceptability of the work system to its users and other people affected by its use.

The different aspects of usability can be evaluated using a variety of methods: user tests and observations of system use, user satisfaction questionnaires, heuristic evaluation, etc. User input logs contain data that can serve as a basis for automatic determination of the time a user spent working on each particular part of the task (see measure (1)), thereby aiding in the evaluation of the efficiency of the interface. The number of changes to the input, measured by (2), is also relevant to interface efficiency: a high value of this measurement is evidence of the user's confusion, and thus indicates lower efficiency and effectiveness of the interface. Finally, the accuracy and completeness of the result of the user's work, i.e. the effectiveness with which the system enables the user to achieve his goal, can be judged by comparing the record of the final content of the input fields to the correct values.

Interface personalization. Data describing the interactions of a particular user (or a group of users) with the system that is collected over time can be used for inferring patterns of usage and thus can help in customizing the interface for that user (or group). Simple examples of such personalization include shortcuts to the most frequently accessed parts of the system and default input values. In more complex cases, the system could detect sets of tasks that are frequently performed in a particular sequence, identify a user's intended process, and provide streamlined access to the appropriate task sequence for completing that process.

Another valuable system capability enabled by the logs is providing a level of instruction that is commensurate with the user's level of competency with the particular part of system. The user's familiarity with a particular interface and knowledge of relevant system features can be dynamically and automatically assessed using measure (4). That information can then be used for targeted on-line training and active help, as done, for example, in [2].

Record of the context. A system that keeps track of its interaction with the user has a greater capability for reasoning about the context of each user action. This contextual awareness can be used to enhance the support given to the user in a variety of situations. For example, consider a case where a user must enter a system-provided identifier for a part into a purchase requisition. Such identifiers are usually difficult to remember, so search mechanisms are typically provided to aid in locating them. An enterprise system usually supplies about a dozen different search methods (e.g. by part type, by plant, by vendor, by manufacturer, etc), which complicates the selection process. However, if by consulting the interaction log the system observes that the user has already entered the destination plant for the ordered part, it can make a useful recommendation to select a search that is narrowed down to those parts used in the previously specified plant.

Another example of benefiting from the system's memory of an interaction is in the fairly common situation where two disjoint tasks use the same data. For example, the user enters a new part specification to the master list with the intention of creating a purchase requisition for it. These two tasks are logically separate and are rarely preformed in sequence. Yet, after the user adds the new part, she should not have to remember the system-assigned part identification in order to enter it into the purchase requisition. Instead, the system should be programmed to automatically use the identifier of that part as a default in the appropriate field of the purchase requisition. Without a record of the user's on-going interaction with the system, it would be impossible to detect that the addition of a part is followed immediately by the creation of a purchase requisition. Therefore, the system would not be able to establish a semantic connection between the added part and its subsequent use. This example demonstrates the use of the system's memory of the interaction across multiple system tasks and modules.

Conclusions and future work

User input logs can be used for tailoring interactions with a system to a specific user. The information they provide about the tasks the user typically performs, their context, and related processes can be used for personalizing and adapting the interface to better meet the needs of the user. The design of the system can also be enhanced based on this context-specific data on user interactions. The design of our prototype incorporates user logging into its data model, thereby enabling continuous tracking of user input and easy access to the interaction history by all system modules and interfaces.

We plan to apply the methods of user interface usability evaluation and interface enhancement that we presented here to our experimental prototype. We will compare the effectiveness of the measurements obtained automatically from its input logs to usability assessments based on already established methodologies in order to assess the validity of our proposed measures. This effort will doubtlessly lead to refinements of and possible extensions to the techniques presented in this paper. We will also continue our research on using input log data for improving the design of system-user interactions.

Acknowledgements

This work has been supported by a grant from Bentley College. The prototype has been implemented by Sean Murray and Vivek Sharma.

References:

[1] Babaian, T., Lucas, W.. and Topi, H.. Collaborating to Improve ERP Usability. Proc. of ICEIS 2004.

[2] Brusilovsky, P. and Cooper, D. W. Domain, Task, and User Models for an Adaptive Hypermedia Performance Support System. Proc. of IUI 2002, 23-30.

[3] ISO 9241-11:1998 Ergonomic requirements for office work with visual display terminals (VDTs) -- Part 11: Guidance on usability (1998)

[4] Jameson, A. Adaptive Interfaces and Agents. Human-computer interaction handbook J. A. Jacko & A. Sears (Eds.), (2003), 305-330.

[5] Perkowitz, M. and Etzioni, O. Towards adaptive web sites: Conceptual framework and case study. Artificial Intelligence, 118, 1-2 (2000), 245-275.

[6] Topi, H., Lucas, W., and Babaian, T. Identifying Usability Issues with an ERP Implementation. Proc. of ICEIS 2005.

[7] Webb, G.I., Pazzani, M. J. and Billsus, D. Machine Learning for User Modeling. *User Modeling and User Adapted Interaction*, 11 (2001), 19-29.