A User-Centered Experiment and Logging Framework for Interactive Information Retrieval [*] [†]

Ralf Bierig
SC&I Rutgers University
4 Huntington St.,
New Brunswick
NJ 08901, USA
bierig@rci.rutgers.edu

Jacek Gwizdka
SC&I Rutgers University
4 Huntington St.,
New Brunswick
NJ 08901, USA
jgwizdka@scils.rutgers.edu

Michael Cole
SC&I Rutgers University
4 Huntington St.,
New Brunswick
NJ 08901, USA
mcole@scils.rutgers.edu

ABSTRACT

This paper describes an experiment system framework that enables researchers to design and conduct task-based experiments for Interactive Information Retrieval (IIR). The primary focus is on multidimensional logging to obtain rich behavioral data from participants. We summarize initial experiences and highlight the benefits of multidimensional data logging within the system framework.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous

Keywords

User logging, Interactive Information Retrieval, Evaluation

1. INTRODUCTION

Over the last two decades, Interactive Information Retrieval (IIR) has established a new direction within the tradition of IR. Evaluation in traditional IR is often performed in laboratory settings where controlled collections and queries are evaluated against static information needs. IIR introduces the user at the center of a more naturalistic search environment. Belkin and colleagues [3, 2] suggested the concept of an information seeking episode composed of a sequence of a person's interactions with information objects, determined by a specific goal, conditioned by an initial task, the general context and the more specific situation in which the episode takes place, and the application of a particular information seeking strategy.

SIGIR'09, July 19-23, 2009,Boston, USA.

[†]This work is supported, in part, by the Institute of Museum and Library Services (IMLS grant LG-06-07-0105-07)

This poses new challenges for the evaluation of information retrieval systems. An enriched set of possible user behaviors needs to be addressed and included as part of the evaluation process. Systems need to address information about the entire interactive process with which users' accomplish a task. This problem has so far only been initially explored [4].

This paper describes an experiment system framework that enables researchers to design and conduct task-based IIR experiments. The paper is focused on the logging features of the system designed to obtain rich behavioral data from participants. The following section describes the overall architecture of the system. Section 3 provides more details about its specific logging features. Section 4 summarizes initial experiences with multidimensional data logging within the system framework based on initial data analysis from three user studies. Future work is proposed in section 5.

2. THE POODLE IIR EXPERIMENT SYSTEM FRAMEWORK

The PooDLE IIR Experiment System Framework is part of an the ongoing research project. The goal of PooDLE[1] to investigate ways to improve information seeking in digital libraries; the analysis concentrates on an array of interacting factors involved in such online search activities. The overall aim of the framework is to reduce the complexity of designing and conducting IIR experiments using multidimensional logging of users' interactive search behavior. Such experiments usually require a complex arrangement of system components (e.g. GUI, user management and persistent data storage) including logging facilities that monitor implicit user behavior. Our framework enables researchers to focus on the design of the experiment including questionnaire and task design and the selection of appropriate logging tools. This can help to reduce the overall time and effort that is needed to design and conduct experiments that support the needs for IIR. As shown in figure 1, the experiment system framework consists of two sides – a server that operates in an Apache webserver environment and a client that resides on the machine where the experiment is conducted. We distinguish the following components:

- *Login and Authentication* manages participants, allows them to authenticate with the system, and enables the system to direct individuals to particular experiment

[1]http://www.scils.rutgers.edu/imls/poodle/index.html

.

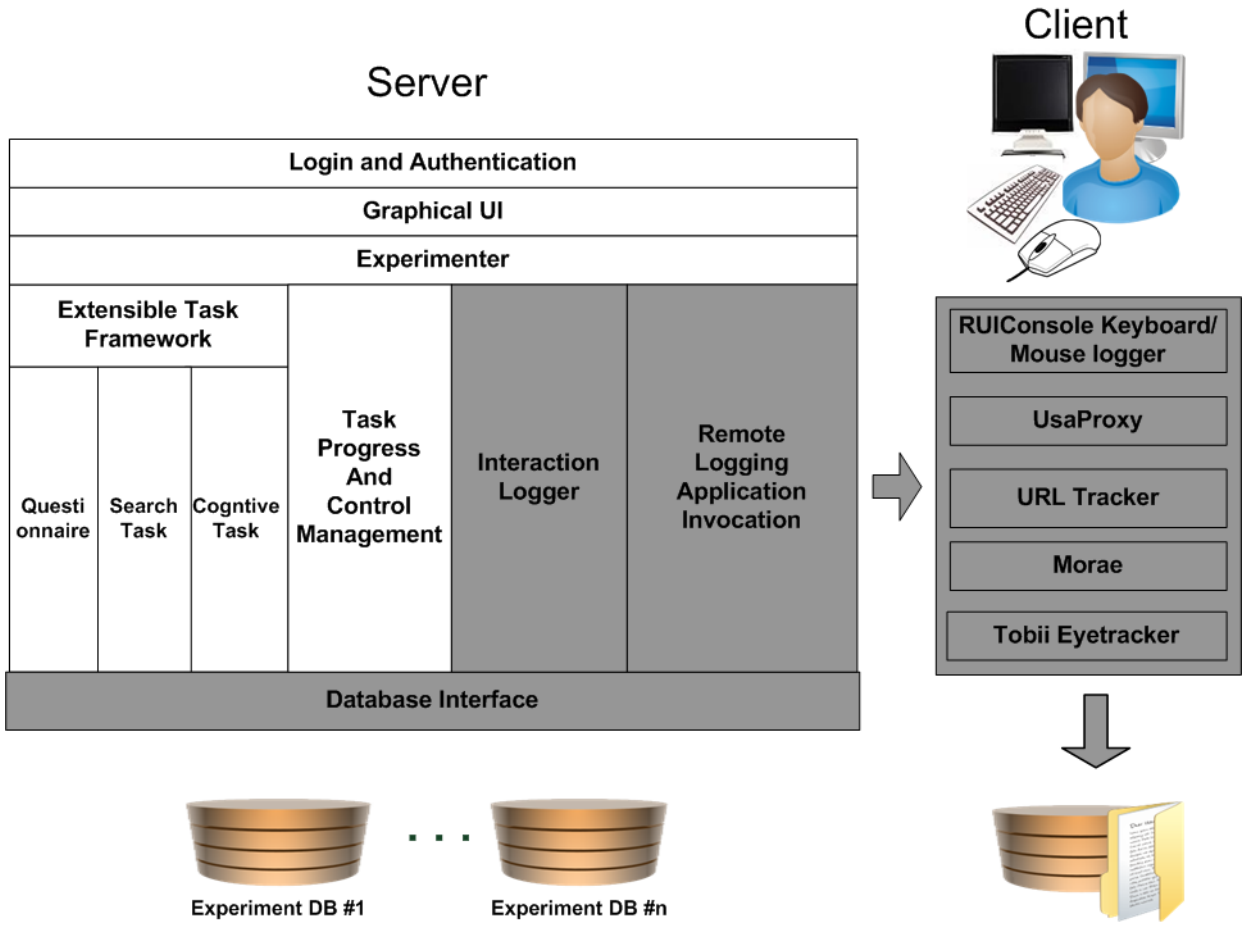

Figure 1: System components of the PooDLE IIR Experiment System Framework. Logging features highlighted in grey.

setups; multiple experiments may exist and users can be registered for multiple or multi-part experiments at any time.

- The *Graphical UI* allows participants to authenticate with the framework and activate their experiment. Each experiment consists of a number of rotated tasks that are provided with a generic menu that presents the predefined task order to the user. After every completed task, the UI guides the participant back to the menu that now highlights the completed tasks. This allows participants to navigate between tasks and gain feedback that helps them to track their progress. In addition, the interface presents participants with additional information, instructions and warnings when progressing through the tasks of an experiment.
- The *Experimenter* controls and coordinates the core components of the system – these are:
  - An *Extensible Task Framework* that provides a range of standard tasks for IIR experiments that are part of the framework (e.g. questionnaires for acquiring background information and general feedback from participants, search tasks with a bookmarking feature and an evaluation procedure, and cognitive tasks to obtain information about individual differences between participants). Tasks are easily added to this basic collection and can be reused as part of the framework in different experiments.
  - The *Task Progress and Control Management* provides participants with (rotated) task sequences, monitors their state within the experiment, and allows them to continue interrupted experiments at a later point in time.
  - The *Interaction Logger* allows tasks to register and trigger logging messages at strategic points within the task. The system automatically logs the beginning and end of each task at task boundaries.
  - *Remote Logging Application Invocation* calls logging applications that reside on the client. This allows for rich client-sided logging of low level user behavior obtained from specific hardware (e.g. mouse movements or eye-tracking information).
- The Database interface manages all access to one or more databases that store users' interaction logs as

well as the basic experiment design for other system components (e.g. participants, tasks and experiment blocks in the form of task rotations for individual users).

3. USER INTERACTION LOGGING

This section focuses on the logging features of the Experiment System Framework as highlighted in grey in figure 1. The logging features and the arrangement of logging tools within the framework have been informed by the following requirements:

- *Hybridity:* All logging functionality is divided between a more general server architecture and a more specific client; this integrates server-based as well as client-based logging features into a hybrid system framework. Whereas the server logs user interactions uniformly across experiments, client logging is targeted to the capabilities of the particular client machine used for the experiment. Researchers can select from a range of logging tools or integrate their own tools to record user behavior. This enables the system to use low level input devices, normally inaccessible by the server, to be controlled by logging tools residing on the client.

- *Flexibility:* Client logging tools can be combined through a loosely coupled XML-based configuration that is provided at task granularity. The system framework uses these task configurations to start logging tools on the client when the participant enters a task and stops them when the participant completes a task. This gives researchers the flexibility to compose logging tools as part of the experiment design and attach them to the configuration of the task. Such configurations can later be reused as design templates which promotes uniformly across experiments and ensures important types of user interaction data are being logged.

- *Scalability:* Experiments can be configured to apply a number of different client machines as part of the data collection. A researcher can, for example, trigger another client computer to record video from a second web camera or simultaneously activate several clients for experiments that involve more than one participant. Redundant instances of the same logging tools can be instantiated to produce multiple data streams to overcome potential measurement errors and instabilities on a data stream due to load or general failure of hard and software.

The client is configured to work with the following selection of open-source and commercial logging tools that record different behavioral aspects of participants:

- *RUIConsole* is an adapted command line version of the RUI tools developed at Pennsylvania State University [5]. RUI logs low level mouse movements, mouse clicks, and keystrokes. Our extension additionally provides full control over its logging features through a command line interface to allow for more efficient automated use within our experiment framework.

- *UsaProxy* is a javascript based HTTP proxy developed at the University of Munich [1] that logs interactive user behavior unobtrusively through injected javascript. It monitors page loads as well as resize and focus events. It identifies mouse hover events over page elements, mouse movements, mouse clicks, keystrokes, and scrolling. Our version of UsaProxy is slightly modified as we don't log mouse movements with this tool. UsaProxy can run directly on the client, but can also be activated on a separate computer to balance load.

- The *URL Tracker* is a command line tool that extracts and logs the users current web location directly from the Internet Explorer (IE) address bar and makes it available to the system framework. This allows any task to determine participants' current position on the web and to monitor their browsing history within a task.

- *Tobii Eyetracker:* We use the Tobii T60 eyetracking hardware which is packaged with Tobii Studio[2], a commercial eyetracking recording and analysis software. The software records eye movements, eye fixations, as well as webpage access, mouse events and keystrokes.

- *Morae* is a commercial software package for usability testing and user experience developed and distributed by TechSmith[3]. It records participants' webcam and computer screen as video, captures audio, and logs screen text, mouse clicks and keystrokes occurring within Internet Explorer.

This extensible list of logging tools are loosely coupled to the *Interaction Logger* and the *Remote Logging Application Framework* components through task configurations for individual tasks. The task configuration describes which logging tools are used during a task and the software framework activates them as soon as participants enter a task and deactivates them as soon as they complete a task.

The researcher can create a selection of relevant tools for each task of a particular IIR experiment from the available logging tools supported by the system framework. First, one should select all user behavior the researcher is interested in. Second, the observable data types that provide evidence for the existence and the structure of these user behaviors is identified. Finally, these data types are linked with relevant logging tools. In the next section we summarize experiences from three distinct experiments that were designed and performed with our experiment system framework. We do not describe these experiments in this paper. Instead, we focus on key points and issues that should be addressed when collecting multidimensional logging data from hybrid logging tools.

4. EXPERIENCES FROM MULTIDIMENSIONAL DATA LOGGING

Data logging with an array of hybrid tools, as described in the previous section, has a number of benefits and challenges. This section summarizes our initial experiences from conducting three IIR user experiments with the system framework and some initial processing and integration of its data logs.

[2]http://www.tobii.com
[3]http://www.techsmith.com

- *Accuracy and Reliability:* Using data streams from multiple logging tools limits the risk of measurement errors to enter data analysis. This is especially relevant to IIR due to its need to conduct experiments in naturalistic settings where people perform tasks in conditions that are not fully controlled and therefore less predictable. Such settings allow participants to solve tasks with great degrees of freedom. As a result of this, user actions in such settings tend to be highly variable. Measurement errors or missing data, for example based on varying system performance and network latencies, have a larger impact because the entire interaction is studied. Multiple data streams from different sources improve the overall accuracy of recorded sessions and increase the reliability of detecting features in individual logs. Furthermore, the use of multiple data logs limits of chances that artifacts created by individual logging tools and their assumptions will affect downstream analysis.

- *Disambiguation:* The use of multiple data logs allows to contextualize each log with the logs produced by other tools and disambiguate uncertainties in the interpretation of logging event sequences. We found that the most common cases are timestamp disambiguation and the synchronization of event accuracies.

  - *Timestamp disambiguation:* The timestamp granularity of recorded events usually varies between logging tools. For example, Tobii Studio records eye tracking data with a constant frequency determined by the eye tracking hardware (e.g. 60 logs per second (17 ms) for the T60 model) whereas UsaProxy records events only every full second and RUIConsole records events dynamically only when they occur. The combination of logging data from different tools helps to better determine the real timing of events by providing different viewpoints for the same sequence of actions a user has performed. Low granularity timestamps might collapse a number of user events to a single point of time and, based on that, change the natural order in which these events are recorded. Alternative secondary logging data can help to detect such event sequences and help disambiguating and correcting them.
  - *Detail of event structure:* Every logging tool imposes a number of assumptions on the data produced by a user – which events to log, which events to differentiate and how to label them. Two logging tools recording the same events can therefore produce different event structures with varying detail. For example, RUIConsole differentiates a mouse click into a press and a release event whereas Tobii Studio considers a mouse click as a single event. Different logging tools recording the same user actions produce events with a structure of different detail that can be used to contextualise conflicting recordings of user actions.

- *Scalability:* Concurrent use of logging tools may create performance issues on the client machine especially with tools that produce large amounts of data. Especially the combined use of Morae and Tobii Studio can be demanding when using high quality web camera and screen capture recording. Limited hardware resources may have a direct effect on the recording accuracy of other logging tools. More importantly, however, a overloaded client may have an effect on participants and their ability to accomplish tasks realistically. This can be avoided by choosing a sufficiently equipped client machine and a fast network. As mentioned in section 4, the software framework supports the distribution of logging tools over several machines, while these tools are activated centrally by the server architecture, which can help to better balance the load.

- *Stability:* Concurrent use of multiple logging applications can destabilize the client computer. Individual applications can affect each other especially when logging from the same resources (e.g. from the same instance of Internet Explorer). Currently, our system framework does not monitor running logging tools and there is no mechanism to recover tools that hang or break during a task. This is a feature we will incorporate into a future version of the system framework.

5. FUTURE WORK

Future work on the experiment system framework will focus on further improvement of logging tool integration and monitoring. We are currently developing a graphical user interface for researchers to more easily design IIR experiments with the system and monitor progress of running experiments and the accuracy of its data logs. An extension to the experiment system framework presented in this paper is a data analysis system that allows us to fully integrate, analyse and develop models from the recorded data. In particular, we are interested in creating higher level constructs from integrated low-level logging data that can be used to personalise interactive search for users. The experiment system framework will be released as open source to the wider research community.

6. REFERENCES

[1] R. Atterer, M. Wnuk, and A. Schmidt:. Knowing the User's Every Move - User Activity Tracking for Website Usability Evaluation and Implicit Interaction. In *15th International World Wide Web Conference (WWW2006)*, Edinburgh, Scotland, 2006.

[2] N. Belkin. Intelligent Information Retrieval: Whose Intelligence? In *Fifth International Symposium for Information Science (ISI)*, pages 25–31, Konstanz, Germany, 1996. Universtaetsverlag Konstanz.

[3] N. Belkin, C. Cool, A. Stein, and U. Thiel. Cases, Scripts, and Information-Seeking Strategies: On the Design of Interactive Information Retrieval Systems. *Expert Systems with Applications*, 9(3):379–395, 1995.

[4] A. Edmonds, K. Hawkey, M. Kellar, and D. Turnbull. Workshop on logging traces of web activity: The mechanics of data collection. In *15th International World Wide Web Conference (WWW 2006)*, Edinburgh Scotland, 2006.

[5] U. Kukreja, W. E. Stevenson, and F. E. Ritter. RUI – Recording User Input from interfaces under Windows and Mac OS X. *Behavior Research Methods*, 38(4):656–659, 2006.