Chapter XXI
Using Action-Object Pairs as a Conceptual Framework for Transaction Log Analysis

Mimi Zhang
Pennsylvania State University, USA

Bernard J. Jansen
Pennsylvania State University, USA

ABSTRACT

In this chapter, we present the action-object pair approach as a conceptual framework for conducting transaction log analysis. We argue that there are two basic components in the interaction between the user and the system recorded in a transaction log, which are action and object. An action is a specific expression of the user. An object is a self-contained information object, the recipient of the action. These two components form one interaction set or an action-object pair. A series of action-object pairs represents the interaction session. The action-object pair approach provides a conceptual framework for the collection, analysis, and understanding of data from transaction logs. We believe that this approach can benefit system design by providing the organizing principle for implicit feedback and other interactions concerning the user and delivering, for example, personalized service to the user based on this feedback. Action-object pairs also provide a worthwhile approach to advance our theoretical and conceptual understanding of transaction log analysis as a research method.

MOTIVATION

The ultimate purpose of search engine designers is to devise Web search engines that provide the most relevant information to each individual user. Since the user decides whether information is relevant or if the system is suitable, it is critical to understand the user's system evaluation. Sun Tzu (n.d./1971), an ancient Chinese military strategist, said "know the enemy, know yourself;

Copyright © 2009, IGI Global, distributing in print or electronic forms without written permission of IGI Global is prohibited.

your victory will never be endangered" (p.129). This advice can be applied on the battlefield, but it can also apply to building information technology systems.

In a broad sense, one can understand Sun's maxim as if you can know your own capability, and the characteristics and capabilities of people you deal with, it will be easier to devise processes appropriate to the situation. Therefore, in order to fulfill users' information needs and serve them better, we should know the users, understand their goals, and recognize their information search tactics. If we can recognize users' needs and their ways of approaching information, we can provide users with more suitable searching systems.

There are multiple ways to identify the individual user and provide tailored information systems. Search engines can learn about the users both explicitly and implicitly (Keenoy & Levene, 2005). In an explicit fashion, the users provide the necessary information to the system. The basis of this approach is that users would like to answer the questions, fill in a series of forms, or set up the profiles themselves. However, according to Keenoy and Levene (2005, p. 205), explicit feedback has low implementation rates due to the high cost of time and energy, unpredictable and unobvious benefits, and privacy concerns. This is in accordance with Zipf's Law – an individual will only perform actions that cost "the least effort" (Case, 2002, p. 140). Zipf's Law is a grounded and fundamental theoretical construct for information seeking studies. Zipf's Law is used to guide user studies and understanding of human behaviors, as well as the development of information systems.

Rather than relying on explicit feedback by users, implicit feedback based on the analysis of interactions between the user and the system may be a better approach (Keenoy & Levene, 2005; Khopkar, Spink, Giles, Shah, & Debnath, 2003). Although it certainly depends on the design goals, the implicit approach is in many ways superior since the user does not need to perform more actions such as answering questions or setting up profiles. It is an unobtrusive method; therefore, the approach has less chance of altering users' behavior.

The implicit approach is also highly dynamic. Since it analyzes and models current user interactions, it adapts well even if the users' information needs change over time. White, Ruthven, and Jose (2001) compared the effectiveness of explicit and implicit feedback techniques and claimed no statistical difference between the two approaches. In addition, according to Zipf's Law (1949), to users, the implicit feedback approach seems to be superior to the explicit feedback approach considering it costs them nothing but has the same effectiveness as the explicit feedback.

A search engine transaction log is "an electronic record of interactions that have occurred during a searching episode between a Web search engine and users searching for information on that Web search engine" (Jansen, 2006, p. 408). One can use the record of these interactions as a source of the implicit feedback. Dumais (2002) believes this is the only method for obtaining considerable amounts of data about users in a complex environment like the Web. Therefore, transaction log analysis seems a practical and convenient way to know the interactions of users with information systems. One can develop the user model by analyzing the data in transaction logs. Using this data, the system can make backward inferences to model the user and then make forward inferences to assist them with their information need.

However, there is a lack of theoretical frameworks for collecting, analyzing, and understanding data from transaction logs. Do we really need to analyze users' every communication with the computer? If not, what kinds of user-system interactions do the transaction logs need to contain? Log files are usually huge and messy. How can we effectively and efficiently organize and analyze them? How can we get the data to make sense and understand users via the log file? A modeling framework is needed to address these problems. In this chapter, we propose the action-object pair approach as a conceptual method to collect, analyze and understand transaction log data.

In the following section, we present the relevant concepts and the theoretical foundations of the action-object pair approach. We will provide a detailed description of the approach in the method section and its potential applications in the application section. We then describe a series of studies on applying the action-object pair approach to show its practical and theoretical values in the case study section. In the conclusion section, we sum up the issues, the underpinnings, and the advantages of the action-object pair approach.

SCIENTIFIC FOUNDATIONS

The foundational concepts of the action-object pair approach include user modeling in information searching, interaction, implicit feedback, and adaptive hypermedia system. User modeling in information searching allows us to conceptualize the interaction between the user and the system. However, there are various forms of interactions. For the system design purposes, we are interested in modeling interactions as actions performing upon information objects presented via the system. Implicit feedback explores the way to comprehend users in an unobtrusive fashion. Actions and objects together can inform us about the user and provide ways to capitalize on the implicit feedback. The implicit feedback can be used in system design, especially for personalization. Adaptive hypermedia system design is a promising way to utilize the implicit feedback and fits well with the action-object pairs approach. All of these concepts lead to the idea of using the action-object pair approach to conceptualize the analysis of transaction log data.

Modeling in Information Searching

Information scientists have contributed to theorizing the interaction process and modeling searchers in information retrieval and seeking. Wilson (1999) defines a model in the following way: "A model may be described as a framework for thinking about a problem and may evolve into a statement of the relationships among theoretical propositions. Most models in the general field of information behavior are of the former variety: they are statements, often in the form of diagrams, that attempt to describe an information-seeking activity, the causes and consequences of that activity, or the relationships among stages in information-seeking behaviour." (p. 250)

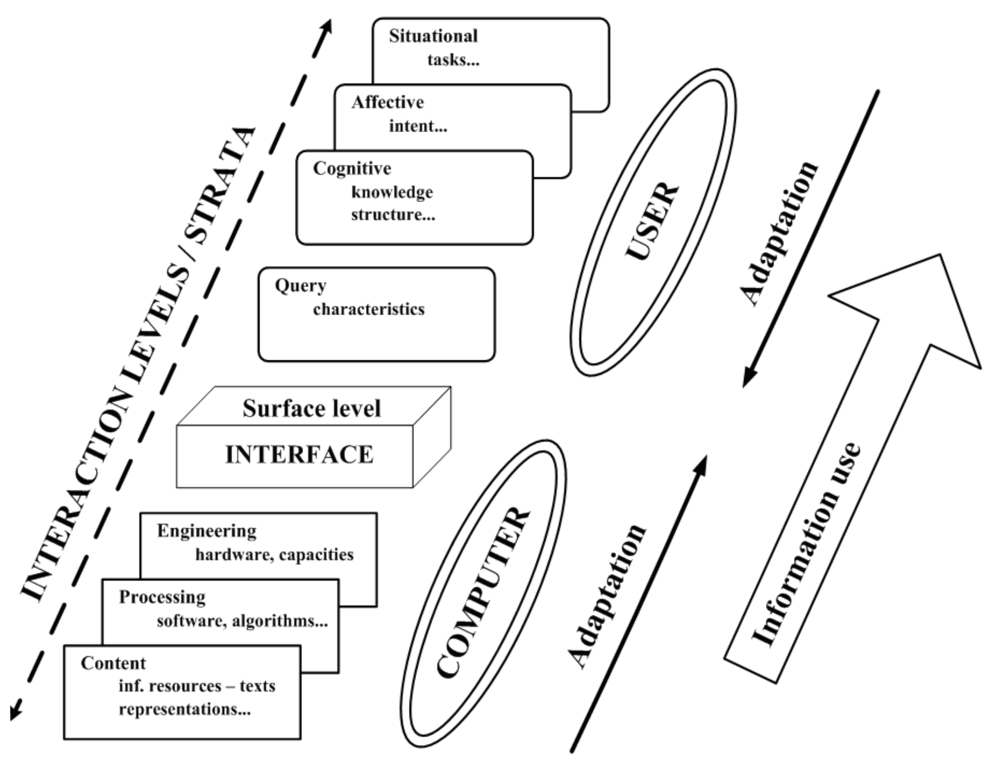
The action-object pair approach is similar to Wilson's (1999, p. 250) concept of a model. It provides a framework for thinking about transaction log analysis, which can uncover the interactive relationship between the user and the system. It attempts to describe an information-seeking activity and depicts the relationships between different sessions of information seeking. The action-object approach is theoretically based on Saracevic's stratified model (Saracevic & Kantor, 1997a, 1997b; Saracevic, Kantor, Chamis, & Trivison, 1988; Saracevic, Mokros, Su, & Spink, 1991; Spink & Saracevic, 1997).

Saracevic and his colleagues (Saracevic & Kantor, 1997a, 1997b; Saracevic et al., 1988; Saracevic et al., 1991; Spink & Saracevic, 1997) developed the stratified model of information retrieval interaction from a series of studies (refer to Figure 1). It describes the interactions between the user and the computer or system during retrieval at a surface level. Saracevic (1997) defined the interaction as "a dialogue between the participants - user and computer - through an interface, with the main purpose to affect the cognitive state of the user for effective use of information in connection with an application at hand" (p.316). It shows that information retrieval (IR) interaction is not a batch process but a deliberate exchange procedure. The exchange occurs on the surface level (i.e. interface). It includes two participants: the user and the computer.

Saracevic (1996, 1997) argued that the user and the computer have different levels or strata. The user side has at least three levels including cognitive, affective, and situational. The cognitive level refers to users' cognitive structures. Users interact with computers and process information cognitively including query development, query modification, relevance judgment, and such. The affective level refers to users' intentions and intentionality including beliefs, motivations, feelings, desires, urgency and so on. It mediates the interaction process. The situational level refers to the context the user is situated in. The context produces the users' information need and influences the way they approach information.

To Saracevic (1996, 1997), the computer side includes at least three strata, which are engineering, processing, and content. The engineering level includes the hardware and its attributes. The analysis will focus on the influence of the attributes on the interaction process. The processing level includes the software and algorithm. The analysis focuses on their effectiveness and evaluation. The content level refers to the information resources and meta-information. The potential analysis could include the adequacy or nature of information, its representation, and so on. The interaction takes place while different levels interact with each other. The adaptations happen to both participants and meet on the interface. The manner that information is used is determined by levels ranging from content toward situation.

Figure 1. Elements in the stratified model of Information Retrieval Interaction (Saracevic, 1997, p. 316)



The strength of the stratified model is its high relevance with the information searching systems. It has a detailed description of the interaction processes and decompositions of both participants into strata, which makes it more relevant to system design compared to most IR models. It focuses exclusively on the query. Saracevic (1997, p. 317) believes "query is the most important aspect of user modeling". One can easily acquire the query via transaction logs. However, the stratified model fails to address the interactive and dynamic nature of the information seeking process beyond labeling it as a communication process. Therefore, we incorporate the action-object pairs into the stratified model and develop the action-object pair approach. We argue that the information seeking process is an interactive and dynamic process as described above. However, what do we mean by "interactive" and "dynamic"? Why is this important for a model?

Interaction

In the area of information searching (i.e., people using online information systems to locate data or information), researchers many times focus on the interactions between people and information searching systems. They picture interactions from different perspectives. Efthimiadis and Robertson (1989) categorized interactions at various stages in the information retrieval process. Bates (1990) presented four levels of interaction (move, tactic, stratagem, and strategy). Belkin and fellow researchers (1995) extensively explored user interaction within an information session. Lalmas and Ruthven (1999) presented interaction as that which occurs across sessions and that which occurs within a session. Jansen and Spink (2006) considered an interaction as any specific exchange between the searcher and the system. The searcher may be multitasking (Spink, 2004) within a searching episode, or the episode may be an instance of the searcher engaged in successive searching (Lin, 2002; Spink, Wilson, Ellis, & Ford, 1998).

While these definitions of interaction are descriptive at a high level, a more practical definition of interaction from the transaction log analysis perspective can benefit both the theoretical understanding and the system design. We propose defining interaction by using an action-object pair. It describes interactions between people and information searching systems as a set of action-object pairs. The interaction process is composed of a series of searchers' actions enabled by the information search systems over some information objects. Action and object set is the basic component of interaction. Our definition can be viewed as a combination of multidisciplinary views of interaction. Action is relevant to research in human-computer interaction and computer science. Object is related to studies in information science. Together they provide a conceptual view of interactions from the user's perspective.

From the discussions above, we have a conceptual understanding of the action-object pair approach. It also has some practical value for system design and development. It can provide implicit feedback to the system in an organized way.

Implicit Feedback

Transaction logs are a method of recording interactions between users and system, and for deriving implicit feedbacks. Implicit feedback is an unobtrusive way to get inputs from users. Researchers have explored various aspects of interactions as measurements of implicit feedback. Goecks and Shavlik (2000) used hyperlinks clicked, scrolling performed and processor cycles consumed. Seo and Zhang (2000) studied reading time, scrolling, link selection and bookmarking as potential implicit feedbacks, and found that bookmarking had the strongest relationship with interesting documents but scrolling had no relationship. Claypool and colleagues (2001) measured mouse clicks, mouse movement, scrolling and elapsed time as the implicit feedback metrics. Kelly and Belkin (2001) studied reading time, scrolling, and interaction. Kelly and Belkin (2004) also examined the display time as the implicit feedback and found no direct relationship between the display time and the usefulness of documents. Shen, Tan, and Zhai (2005) employed previous queries and click through information as the implicit feedback measures.

Oard and Kim (2001) considered all users' behaviors as a form of implicit feedback and proposed a framework for observed behaviors to improve system performance (refer to Table 1). The framework has two axes: behavior category and minimal scope. Behavior category includes four types of observable behavior: examine, retain, reference, and annotate. Examine refers to searchers' behaviors of checking the information content. It can be view, listen, and select. Retain is about the behaviors of preserving the information content for future usage. It can be print, bookmark, save, delete, purchase, and subscribe. Reference is to create linkage between information contents. It can be copy-paste, quote, forward, reply, link, and cite. Annotate refers to intended behaviors to add personal values to the information content. It can be mark up, rate, publish, and organize. Most follow-on implicit feedback classifications have adhered to this conceptual presentation.

Minimal scope is "the smallest unit normally associated with the behavior" (Oard & Kim, 2001, p. 484). It has three levels, which are segment, object, and class. A segment is a portion of an information object. An object is a self-contained information entity. A class is a set of objects. For example, a Webpage can be an object. A sentence or a paragraph on the Webpage is a segment. A Website including several Webpages is a class. (Oard & Kim, 2001)

Kelly and Teevan (2003) further developed this framework (refer to Table 2) by adding a fifth behavior category: create, which refers to the generation of the information content. It can be type, edit, and author. They also added scroll, find, and query as actions of examining the information segment; browse as action of examining the information class; and email as action of retaining the information object.

Table 1. Potentially observable behaviors (Oard & Kim, 2001, p. 484)

		MINIMAL SCOPE		
		Segment	Object	Class
BEHAVIOR CATEGORY	Examine	View	Select	
		Listen		
	Retain	Print	Bookmark	Subscribe
			Save	
			Delete	
			Purchase	
	Reference	Copy-paste	Forward	
		Quote	Reply	
			Link	
			Cite	
	Annotate	Mark up	Rate	Organize
			Publish	

Jansen and McNeese (2005) further refined this framework and applied it specifically in the Web searching domain (refer to Table 3). They extended the minimal scope axis by adding interface as the minimal scope of the system. The original components are mainly about the Internet content in the information searching area.

In addition, they dropped annotate and create on the behavior category axis because they are more related to the manipulation of information content and less related to information searching. They added two other behavior categories: execute and navigate. These are common behaviors during Web search. Jansen and McNeese's (2005)framework is exclusively tailored for information searching. The actions in each cell also have been altered accordingly. This modified version of the framework is a version of the action-object approach per se. The minimal scope axis is the object. The behavior category axis is the action in a broad term. In each cell, there are actions on the ground level. Using the action-object approach, one could acquire the implicit feedback from searchers.

With the implicit feedback available, what can this information do for the system design? How can we effectively utilize the implicit feedback acquired by using the action-object approach? Adaptive hypermedia system design techniques address these questions, which we can leverage for transaction log analysis and the design of Web searching systems.

Adaptive Hypermedia System

With the implicit feedback, we could personalize a system by utilizing the adaptive hypermedia system design techniques. The adaptive hyper-

Table 2. Modified potentially observable behaviors by Kelly and Teevan (2003, p. 19)

		MINIMAL SCOPE		
		Segment	Object	Class
BEHAVIOR CATEGORY	Examine	View	Select	*Browse*
		Listen		
		Scroll		
		Find		
		Query		
	Retain	Print	Bookmark	Subscribe
			Save	
			Delete	
			Purchase	
			Email	
	Reference	Copy-paste	Forward	
		Quote	Reply	
			Link	
			Cite	
	Annotate	Mark up	Rate	Organize
			Publish	
	Create	*Type*	*Author*	
		Edit		

Table 3. Classification of implicit feedback on system and content during information searching process (Jansen & McNeese, 2005, p. 1482)

		MINIMAL SCOPE			
		SYSTEM	CONTENT		
		Interface	Segment	Object	Class
BEHAVIOR CATEGORY	Execute	Query	Click	Select	
		Open	Scroll		
		Close			
		Resize			
	Examine		View	Open	Browse
			Find		
	Navigate	Back		GoTo	
		Forward		Previous	
				Next	
	Retain	Create	Print	Bookmark	
		Name		Save	
				Purchase	
				E-mail	
	Reference		Copy-Paste		

media systems are defined as "all hypertext and hypermedia systems which reflect some features of the user in the user model and apply this model to adapt various visible aspects of the system to the user" (Brusilovsky, 1996, p. 88). They combine system design with user modeling to fulfill the heterogeneous information needs of each individual user (Bailey, Hall, Millard, & Weal, 2007; Brusilovsky, 1996; Cannataro, Cuzzocrea, & Pugliese, 2001). The hypermedia system is designed to adapt to users' goals, knowledge, background, hyperspace experience and preferences (Brusilovsky, 1996, p. 93-96).

Brusilovsky (1996, pp. 96-100) states that system adaptation can be on two levels: content-level (adaptive presentation) and link-level (adaptive navigation). The adaptive presentation can include technologies such as adaptive multimedia presentation and adaptive text presentation. The adaptive navigation support can contain technologies such as direct guidance, adaptive sorting of links, adaptive hiding of links, adaptive annotation of links, and map adaptation.

Cannataro, Cuzzocrea, and Pugliese (2001) proposed that the adaptive hypermedia system has three basic components: "the Application Domain Model, the User Model, and the techniques to adapt presentations with respect to the user's behavior and to the content provider's goals" (p. 411). The Application Domain Model refers to the descriptions of the hypermedia contents and their organization architecture. Datacentric is the most promising modeling approach. The user modeling is used to uncover "the user's characteristics and preferences and his/her expectations in the browsing of hypermedia" (Cannataro et al., 2001, p. 411).

Cannataro, Cuzzocrea, and Pugliese (2001, p. 411) claimed that this approach to profile users was different from the overlay model and stereotype model. The former approach typically utilizes a series of attribute-value pairs to present the user's characteristics. The latter approach usually classifies users into different groups. The adaptive presentation tailors the presentation of the Application Domain according to the User Model. It is "a manipulation of information fragments, adaptive navigation support" (Cannataro et al., 2001, p. 411) and "a manipulation of the links presented to the user" (Cannataro et al., 2001, p. 411). Ceri and his peers (Ceri, Daniel, Matera, & Facca, 2007) described that the adaptive actions can be adaptive page contents, adaptive navigation, adaptive site view, and adaptive presentation style.

De Bra and Calvi (1998) proposed the concept-value pair method to model users. The adaptive system learns users based on their actions or the answers to the system's questions and employs these actions to predict their needs and desires. Concept-value pairs are used to build up models of the user. In a (*c*, *v*) pair, *c* is a *concept* and *v* is a *value*. The pair represents the amount of knowledge that the user has about a certain concept. The term concept is used in a broad way here, which can also refer to the user's preference. Values can be described in different fashions including numbers, descriptions, and Booleans. For example, the concept is "something", and the value can be a percentage (for instance, 99%), "no knowledge, somewhat knows about, familiar", or "true or false". De Bra and Calvi (1998) believed the representation system with many values cannot be simulated in a practical sense. It would be impossible to simulate "something" with an infinite number of percentages as values. Therefore, the concept should be defined in a fine-grained way for the purpose of simulation. This is a simple but practical user modeling approach. It simulates users in a programmable way. We also draw the action-object pair approach from it.

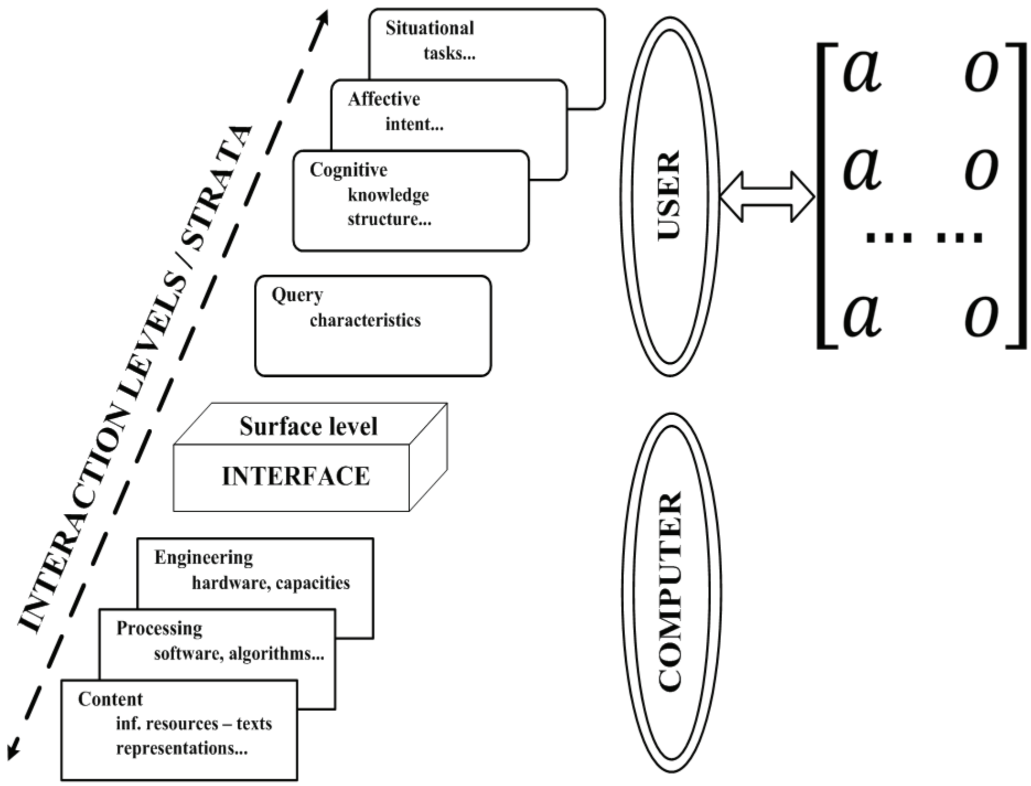
ACTION–OBJECT PAIR APPROACH DESCRIPTION

Successful log analysis is determined by "conducting the analysis with an organized approach" (Jansen, 2006, p. 420). The question is how to define "organized". Most of the previous search logs are organized and analyzed to address some research questions. The typical research questions are at the aggregate level, including the length of query, number of queries per search session, query reformulation pattern, and such (Park, Bae, & Lee, 2005; Silverstein, Henzinger, Marais, & Moricz, 1999; Wang, Berry, & Yang, 2003). These analyses are research question oriented and organize user data according to the research questions addressed. It is an organized approach but has little direct value to the design of personalized systems. Another "organized" approach is individual user oriented. The log analysis is conducted according to each user. This approach is more suitable for the personalized system design. Therefore, we propose the action-object pair approach.

The action-object pair approach is a conceptual framework for transaction log analysis. It is developed based on extending Saracevic's stratified model and modifying the concept-value approach (refer to Figure 2 and 3). The stratified model allows us to describe the interaction process between the user and the system. Its user modeling part does not fit our purpose of developing a conceptual framework for transaction log analysis. Therefore, we replace the user modeling portion with the action-object pairs, which are developed from the concept-value approach.

The conceptual component in the action-object pair approach is (a, o) pair. In a (*a*, *o*) pair, *a* stands for action and *o* stands for object. An action is a specific expression of the user. An object is a self-contained information object, the receipt of the action. One (*a*, *o*) pair represents one interaction between the user and the system. Action can be *submit*, *copy*, *paste*, *print*, *save*, *submit*, *scroll*, *modify*, *click*, *resize*, and such. An action

Figure 2. Extension of stratified model by using action-object pair



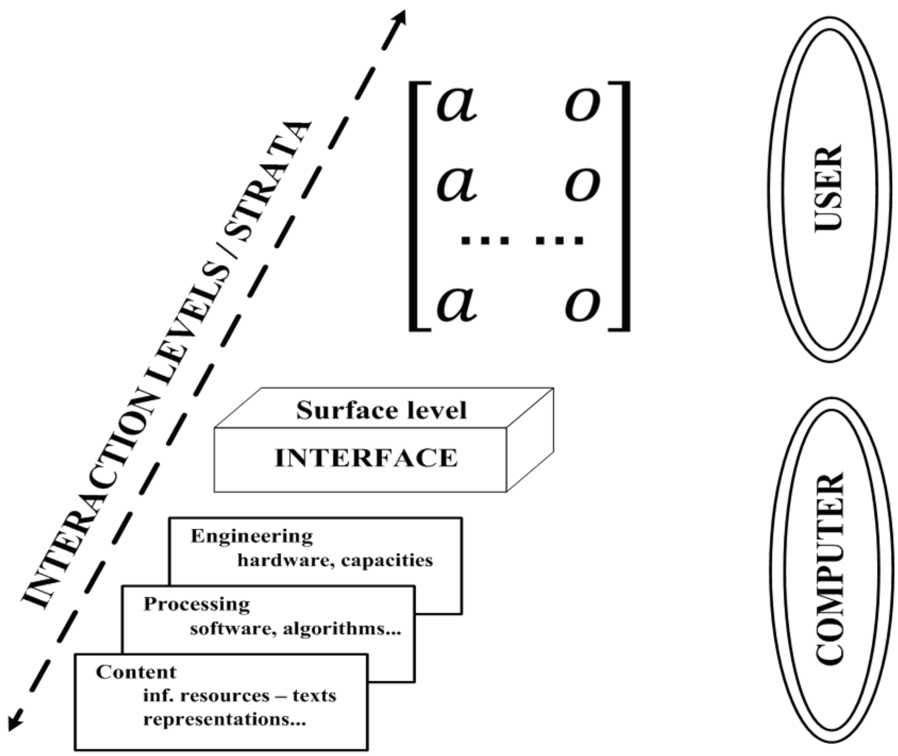
can be derived from analyzing different strata of users and computers. A detailed list of potential actions is in Table 3.

An object can be *query*, *URL*, *result*, *Webpage*, *scrollbar*, *window*, and such. Objects can be acquired by studying different strata of computers, especially the content level. For example, a user is interested in purchasing a Canon G9, a digital camera and looking for some reviews before placing the order. The user submits the query "canon g9 review". We can organize this interaction by using action-object pair. Action is *submit* and object is *a query*. The (a, o) pair is (*submit*, *canon g9 review*).

One (a, o) pair is one interaction between the user and the system. A series of (a, o) pairs or an a-o matrix can represent the interaction session, which is defined as a series of interactions between the user and the system to fulfill the user's certain information need. According to the stratified model, the interaction session is the product of situational, affective, and cognitive strata interacting with a search engine via queries on the surface level (Saracevic, 1996, 1997). Therefore, an a-o matrix can also be viewed as such a product. We can use backward inference from the product to acquire insight about the user's three strata. Thus, Saracevic's stratified model can be modified by using a-o matrix to inform on the strata of the user (refer to Figure 2 and 3). The a-o matrix development rules of thumb are: the more (a, o) pairs, the more complicated the model will be (Jansen & Pooch, 2001, p. 22); the more (a, o) pairs, the more accurate the model will be.

This approach provides a novel and efficient way to link interactions between the user and the system together. It can be applied to collecting, analyzing and understanding transaction logs. It

Figure 3. Modified version of stratified model



provides guidance to understand the records of interactions. We can analyze the log by creating an *a-o* matrix. We can analyze the frequency of action-object pairs. Once the frequent co-occurrences of action-object pairs are identified, they can be recommended if they do not appear together. The frequent co-occurrences of action-object pair orders can be recommended when the *a-o* pairs are not presented in those orders. Some low-performance actions can be improved by triggering the recommendation mechanisms. The analysis result can be understood by using a modified version of the stratified model.

The action-object pair approach can be used in designing adaptive search engines. It is an approach developed for the information searching domain from the concept-value pair method, which originally was used to model users and design adaptive hypermedia systems (De Bra & Calvi, 1998; Jansen & Pooch, 2001). Thus, we can use the action-object pair approach to develop adaptive search engines. Adaptive search engine help fulfill the information searching needs of the individual user. They can provide adaptive presentation and adaptive navigation. For example, the search engine can provide different link summaries on the search engine result page to different users. These are potential ways to improve users' information relevance judgment.

We believe the action-object pair approach is an extremely workable method since the user does not need to perform more actions such as answering questions or setting up profiles. It is an unobtrusive method. This approach is also highly dynamic. It can model the user in a timely manner, considering that the users' information needs change all the time. Their actions and the objects they act on are the products of the cognition, affection and situation altogether. It can benefit system design since users' actions are recorded in a way convertible into code to modify the system. In the next section, we will present the potential applications of the action-object pair approach to show its practical value.

APPLICATION

There are three major applications for the action-object pair approach. It can be applied to transaction log collection, transaction log analysis, and understanding users. The action-object pair approach addresses the question on what types of interactions need to be recorded in the transaction log. It accounts for how the transaction log should be analyzed. It uncovers a new way to understand users via data collected in the transaction log.

Transaction Log Collection

The action-object pair approach can be used to guide the transaction log creation. Early in the history of system design, the transaction log was created primarily for system maintenance purposes. It was not utilized for other purposes concerning the user. Jansen, Spink, and Saracevic (2000) published one of the first journal papers using a Web log to understand various aspects of user interactions during Web searching. There are two types of search logs: server-side logs and client-side logs. Server-side logs are generated by Web server applications and record the interactions between the user and search engines via browsers on the computer. Typical server-side logs usually include user identification, date, time, and query. Client-side logs are generated by applications on a user's computer and include the full range of interactions between user-system compared with the server-side logs. W3C (Hallam-Baker & Behlendorf, n.d.) defines the standard format of transaction logs. However, there is a lack of framework to guide the user-system interactions, which the serve-side logs and the client-side logs should capture, especially for system design and user understanding purposes.

The action-object pair approach can address this data collection issue. Transaction logs do not include every user-system interaction. Detailed records of the user-system interactions indeed can bring us a comprehensive and more accurate interaction model. However, it is costly in terms of systems resources. In addition, different people have different standards about degree of comprehensiveness and accuracy. One will agree that the most suitable record is the one that is comprehensive and accurate, while the least costly. Therefore, the right amount of data really depends on the purposes of the transaction log. If you want to use the data to study the collaborative information behavior of a distributed group, you may want to record the communicative actions among group members. These communications have great influence over how the group conducts search. If you are only interested in the individual search behavior, you can ignore these communicative actions. Thus, the action-object pair you are interested in will decide the interactions that the log files should capture. Potential recordable interaction actions are shown in Table 3.

Transaction Log Analysis

Transaction logs are typically messy and large files. How to organize the data in order to conduct an efficient analysis is the starting point of log analysis. It is fundamental and critical in the log analysis process. We propose the using action-object pair approach to organize the transaction log. Every interaction can be transformed to be an action-object pair. A set of action-object pairs can be placed in the modified version of the stratified model (refer to Figure 3). Action-object pairs help frame the analysis and benefits system design. Based on the action-object pairs model, one can consider what kinds of design should be made to support the action-object pairs on the engineering, processing, and content strata.

The action-object pairs can be created in a codable fashion. This characteristic will benefit system design. Action-object pairs can be generated automatically by the software packages. Therefore, the system can conduct transaction log analysis in real time. Based on the analysis results, the system can provide some potential live suggestions or some adaptive personalization to the user. For example, certain action-object pairs can trigger certain system actions. A (*submit*, *query*) can initiate the spelling check function. The spelling suggestions can remind the user to check on some possible mistakes. This feedback and engagement will make sure the user employs the right word to describe his/her information needs and frame the query.

User Modeling

The action-object pair approach can be applied to knowing users and understanding users. In the stratified model, Saracevic (1996, 1997) argued the user's actions were the products of the situation, affection, and cognition combination from the user side. Profilers picture people based on their actions. One can then compose the user file based on the action-object pair approach via the transaction log (refer to Figure 2).

As we have pointed out above, each item in the transaction log can be converted to an action-object pair. Each pair informs us of the user. Based on categorizing the information objects, you can know the domains in which he/she will have an interest. Based on the user's actions and previous studies on the implicit feedback, you can infer if the user finds the relevant information. Based on the user's frequent actions on processing relevant information, one can predict what the user will do the next time in such a situation.

In order to further explain the practical value of the action-object pair, we will present applicable research in the following case study section.

CASE STUDY

The action-object approach has been extensively applied in a series of studies by Jansen (Jansen, 2003, 2005, 2007; Jansen & McNeese, 2005; Jansen & Pooch, 2004). The researchers (Jansen, 2003, 2005; Jansen & Pooch, 2004) employed the action-object approach to design a software agent as plug-in to monitor and support users' interactions. The agent monitored five actions: *bookmark*, *copy*, *print*, *save*, *and submit*; and identified three objects: *documents*, *passages from documents*, and *queries*. The agent monitors the log file. When a certain action-object pair appears, the assistance will be triggered. For example, the action is *submit* and the object is *query*. The action-object pair is (*submit*, *query*). The assistance triggered can be spell-checking and providing the spelling modification suggestions.

The agent provided assistance on five major issues: structuring queries, spelling, query refinement, managing results, and relevance feedback (Jansen, 2003, pp. 746-747, 2005, p. 914; Jansen & Pooch, 2004, pp. 22-24).

Structuring Queries

Users find it hard to properly structure queries, especially applying the rules of a particular system. In particular, they do not know how to and when to use Boolean operators (e.g., AND, OR, NOT) (Proctor, 2002) and term modifier symbols (e.g. '+', '!') (Spink, Jansen, Wolfram, & Saracevic, 2002) in an appropriate way on certain systems.

Agent Assistance

If the user submits a query, the agent recognizes this as a (*submit*, *query*) pair. It checks the query's structure based on the system's syntactic rules and corrects any mistake to make sure the query is properly structured before submitting it to the search engine.

Spelling

Users often make spelling mistakes in queries (Jansen et al., 2000; Yee, 1991), which can potentially reduce the number of results returned. However, it is usually difficult to detect these spelling errors because people can make the same mistakes while creating the documents, and especially in large document collections like the Internet, the probability of making the same spelling mistakes is extremely high. Therefore, these misspelled queries frequently retrieve results. The user may even not notice the query includes a spelling mistake.

Agent Assistance

A (*submit*, *query*) pair alerts the agent to check for spelling. The agent parses the query into terms, examining each term using an online dictionary. If the agent fails to locate the term in the dictionary, it will provide spelling suggestions or remind the users to check the spelling themselves. The agent's current online dictionary is *ispell* (Gorin, 1971). It can employ any online dictionary by using the proper application program interface (API).

Query Refinement

Searchers do not modify their query, although there may be other terms that relate directly to or better describe their information needs (Bruza, McArthur, & Dennis, 2000). Studies by Jansen and his colleagues (1998) disclose that searchers rarely refine their queries, or do so incrementally. They usually modify their queries only one or two times.

Agent Assistance

By identifying a (*submit*, *query*) pair, the agent analyzes each query term and looks into a thesaurus to suggest synonyms and the contextual definitions of the query terms. The agent uses WordNet (Miller, 1998) but can utilize any online thesaurus with proper modifications.

Managing Results

Users have difficulty managing the number of results (Gauch & Smith, 1993). They have difficulty in increasing the number of results when there are not enough results and decreasing the number of results when there are too many results (Yee, 1991). Roughly speaking, user queries are very broad. Broad queries usually result in an unmanageable number of results. However, Silverstein and his colleagues (1999) claimed that few searchers view more than the first ten or twenty documents from the result list.

Agent Assistance

Recognizing the (*submit*, *query*) pair and the number of results, the agent provides suggestions to refine the query. When the number of results is larger than twenty, the agent provides guidance to refine the query to reduce the length of the result list. When the number of results is less than twenty, the agent provides suggestions to expand the query to increase the results returned by the system.

Relevance Feedback

Harman (1992) pointed out that relevance feedback provided effective search assistance. However, searchers seldom use it even if it is offered. Koenemann and Belkin (1996) proposed to automate this process. Mitra and his peers (1998) suggested automating this process by using term relevance feedback.

Agent Assistance

Upon recognizing a (*bookmark*, *document*), (*print*, *document*), (*save*, *document*) or (*copy*, *passage*) pair, the agent executes a version of relevance feedback by using terms from the document or passage object. For example, when the user goes over a document from the results list and conducts bookmarking, printing, or saving, the agent recommends terms from the document that the user can have potential interest in adding to the query.

According to Jansen and his peers (2003, 2004, 2005), the agent can be enabled or disabled by users. The empirical test shows this agent could significantly improve the system's performance in terms of precision. All the participants employed the agent feedback at least once. Users were willing to accept automated assistance especially after locating the relevant information. Query refinement was the most frequently used assistance and relevance feedback was least frequently used. The average workload was measured by using the SWAT method (Boff & Lincoln, 1988) and the result was 5.37 out of 9. The potential source of workload was the inappropriate feedback supply manner.

Jansen and McNeese (2005) further developed this agent. They refined the term used to describe action and object. They replaced *copy* as *copy-paste*, *submit* as *execute*, *passages from documents* as segment. It records more actions including *send to*, *view*, *scroll*, *next*, *goto*, and *previous*, although the agent does not utilize them to make any inference. The reason was due to a lack of consensus on the implicit feedback from these actions. They add a module called tracking module to formulate the action-object pair and then send it to a certain module of the agent. They dropped the query structuring module and added a new module called similar queries. Search engines such as AltaVista (Anick, 2003) recommends similar queries from previous users for current users to reformulate the queries. The agent recognizes the (*submit*, *query*) pair and searches for queries containing all or some of this query submitted by previous users. It displays the top three unique queries as recommended modifications.

According to Jansen and McNeese (2005), the modules in the agent have been improved. Spelling assistance is taken care of by the query term module. The agent triggered by the (*submit*, *query*) pair not only parses query into terms but also removes query operators (such as MUST APPEAR, MUST NOT APPEAR, and PHASE). The dictionary has been switched from *ispell* (Gorin, 1971) to Microsoft Office Dictionary. The query refinement module has also been changed. If there are more than 30 results returned by the system, the agent initiated by the (*submit*, *query*) pair will try to locate AND, MUST APPEAR, and PHRASE operators. If there is no such operator, the agent will reformulate queries with existing terms and appropriate usage of AND, MUST APPEAR, and PHRASE operators. If the query has AND or MUST APPEAR operators, the agent will reformulate the query with PHRASE operator. If the query has PHRASE operator, there is no action from the agent. If the number of results returned by the search engine is less than 20, a similar process as above will happen by replacing AND with OR. The managing results module was renamed as a query reformulation module. Instead of using 20 results returned by the search engine as a boundary of too many or too few results, they used 10 and 30 (i.e. if the number of the results is less than 10, then there are too few results; if the number of results is larger than 30, then there are too many results). The agent will make certain recommendations accordingly. In addition, the agent now preprocesses each term in the query with the assistance of the Microsoft Office thesaurus before sending it to the thesaurus API.

Jansen and McNeese (2005) empirically evaluated the second version of the agent. They found that the system performance had increased about 20% measured by the number of user-selected relevant documents. The users interacted with the system in a predictable way. The most common three-state pattern is Execute Query - View Results: With Scrolling - View Assistance. The implementation rate of the agent was 71%. However, there was no obvious correlation between the use of assistance and previous searching performance. Jansen (2007) conducted another user study by using the second version of the agent to test the effectiveness of automated searching assistance based on the implicit feedback. The conclusion was that the searching performance indeed improved and increased about 30% but the result depended on the evaluation metric used.

In the case study above, the action-object pair approach has shown its values in terms of providing a theoretical framework for collecting, analyzing, and understanding the search log file. It also facilitated the system design and improved the system performance. The participants did not reject of using the agent. All of these indicate the action-object pair approach is a promising conceptual framework for data collection, transaction log analysis, and user modeling.

CONCLUSION

Understanding users is critical for effective system design. Implicit feedback functions better than explicit feedback in many situations since it is an unobtrusive and burden-free method for users. A transaction log is a direct and convenient way to record implicit feedback from users. Researchers and designers can exploit this resource. The question is "how". How can one get the most value from large and messy log files? How can one know if the data in the log is recorded in an appropriate fashion to provide the implicit feedback? How can one know what is an efficient and effective way to make sense of the log? How can one know if the transaction log is processed in the right way to get the implicit feedback? There is a lack of frameworks for providing guidance for collecting, analyzing, and understanding data from transaction logs. Therefore, we propose the action-object pair approach as a conceptual framework for transaction log analysis.

The action-object pair approach is an extension of Saracevic's stratified model and developed by modifying the concept-value approach. We use the stratified model as a starting point for this modeling approach. The user component of the stratified model is adjusted to fit the purpose of transaction log analysis. We modify the concept-value approach, converting it to the action-object pair approach. This approach is utilized to replace the user component in the stratified model. From this, one gets a modified version of the stratified model (refer to Figure 3). In the action-object approach, an action is defined as a specific expression of the user. An object is a self-contained information object, the receipt of the action. One (a, o) pair is one interaction between the user and the system. A set of (a, o) pairs or a-o matrix can represent the interaction session, in which a particular information need gets satisfied.

The action-object pair approach provides a novel way to guide the transaction log collection, organize the transaction log, and deliver the implicit feedback to the system. The log file must have enough data to build the action-object pairs. The more (a, o) pairs, the more accurately one can model the user. Therefore, the log can be organized as (a, o) pairs. The system can use the (a, o) pairs as the implicit feedback. Based on it, the system can provide adaptive services to the users. The system can model users in a timely fashion, which means the system can potentially provide timely adaptation to the user. This is very important. In a series of queries, a user can refer to 'Amazon' as an online store for one query and as a river in the next query. In addition, the action-object pair approach advances our conceptual understanding of collecting, analyzing, and comprehending transaction logs. It sheds some theoretical light on questions such as what to collect and how to organize, analyze, and understand the log file.

REFERENCES

Anick, P. (2003, July-August). *Using terminological feedback for Web search refinement - A log-based study*. Paper presented at the Twenty-Sixth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Toronto, Canada.

Bailey, C., Hall, W., Millard, D. E., & Weal, M. J. (2007). Adaptive hypermedia through contextualized open hypermedia structures [Electronic Version]. *ACM Transactions on Information Systems*, *25*(4), Article 16. Retrieved October 20, 2007, from http://doi.acm.org/10.1145/1281485.1281487

Bates, M. J. (1990). Where should the person stop and the information search interface start? *Information Processing and Management*, *26*(5), 575-591.

Belkin, N., Cool, C., Stein, A., & Thiel, S. (1995). Cases, scripts, and information-seeking strategies: On the design of interactive information retrieval systems. *Expert Systems with Applications*, *9*(3), 379-395.

Boff, K. R., & Lincoln, J. E. (1988). *Engineering Data Compendium: Human Perception and Performance*. Wright-Patterson A.F.B. Dayton, Ohio: Harry G. Armstrong Aerospace Medical Research Laboratory.

Brusilovsky, P. (1996). Methods and techniques of adaptive hypermedia. *User Modeling and User-Adapted Interaction*, *6*(2-3), 87-129.

Bruza, P., McArthur, R., & Dennis, S. (2000, July). *Interactive Internet search: Keyword, directory and query reformulation mechanisms compared*. Paper presented at the Twenty-third Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Athens, Greece.

Cannataro, M., Cuzzocrea, A., & Pugliese, A. (2001). A probabilistic adaptive hypermedia system. *Proceedings of the International Conference on Information Technology: Coding and Computing (ITCC '01)*, USA, 411-415.

Case, D. (2002). *Looking for information: a survey of research on information seeking, needs, and behavior*. San Diego, CA: Academic Press.

Ceri, S., Daniel, F., Matera, M., & Facca, F. M. (2007). Model-driven development of context-aware Web applications [Electronic Version]. *ACM Transactions on Internet Technology,* 7(1), Article 2. Retrieved October 20, 2007, from http://doi.acm.org/10.1145/1189740.1189742

Claypool, M., Le, P., Wased, M., & Brown, D. (2001, January). *Implicit interest indicators*. Paper presented at the Sixth International Conference on Intelligent User Interfaces, Sanata Fe, NM.

De Bra, P., & Calvi, L. (1998, June). *AHA: A generic adaptive hypermedia system*. Paper presented at the Second Workshop on Adaptive Hypertext and Hypermedia, HYPERTEXT'98, Pittsburgh, PA.

Dumais, S. T. (2002). *Web experiments and test collections*. Retrieved April 20, 2003, from http://www2002.org/presentations/dumais.pdf

Efthimiadis, E. N., & Robertson, S. E. (1989). Feedback and interaction in information retrieval. In C. Oppenheim (Ed.), *Perspectives in information management* (pp. 257-272). London: Butterworths.

Gauch, S., & Smith, J. (1993). An expert system for automatic query reformulation. *Journal of the American Society for Information Science*, *44*(3), 124-136.

Goecks, J., & Shavlik, J. (2000, January). *Learning users' interests by unobtrusively observing their normal behavior.* Paper presented at the Fifth International Conference on Intelligent User Interfaces, New Orleans, LA.

Gorin, R. E. (1971). *Developer of ispell*. Retrieved 12 February, 2000, from http://fmg-www.cs.ucla.edu/geoff/ispell.html

Harman, D. (1992, June). *Relevance Feedback Revisited.* Paper presented at the Fifteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Copenhagen, Denmark.

Jansen, B. J. (2003, October). *Designing automated help using searcher system dialogues.* Paper presented at the 2003 IEEE International Conference on Systems, Man and Cybernetics, Washington, DC.

Jansen, B. J. (2005). Seeking and implementing automated assistance during the search process. *Information Processing and Management*, *41*(4), 909-928.

Jansen, B. J. (2006). Search log analysis: What is it; what's been done; how to do it. *Library and Information Science Research*, *28*(3), 407-432.

Jansen, B. J. (2007). *Evaluating the effectiveness of automated searching assistance*. Manuscript submitted for publication.

Jansen, B. J., & McNeese, M. D. (2005). Evaluating the effectiveness of and patterns of interactions with automated searching assistance. *Journal of the American Society for Information Science and Technology*, *56*(14), 1480-1503.

Jansen, B. J., & Pooch, U. (2001). Web user studies: A review and framework for future work. *Journal of the American Society for Information Science and Technology*, *52*(3), 235-246.

Jansen, B. J., & Pooch, U. (2004). Assisting the Searcher: Utilizing Software Agents for Web Search Systems. *Internet Research - Electronic Networking Applications and Policy*, *14*(1), 19-33.

Jansen, B. J., & Spink, A. (2006). How are we searching the World Wide Web? A comparison of nine search engine transaction logs. *Information Processing and Management*, *42*(1), 248-263.

Jansen, B. J., Spink, A., & Saracevic, T. (2000). Real life, real users, and real needs: A study and analysis of user queries on the Web. *Information Processing and Management*, *36*(2), 207-227.

Keenoy, K., & Levene, M. (2005). Personalisation of Web search. In B. Mobasher and S. S. Anand (Eds.), *Intelligent Techniques for Web Personalization* (pp. 201-228). Berlin, German: Springer-Verlag.

Kelly, D., & Belkin, N. J. (2001, Sepetmber). *In reading time, scrolling and interaction: Exploring implicit sources of user preferences for relevance feedback.* Paper presented at the Twenty-fourth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, New Orleans, LA.

Kelly, D., & Belkin, N. J. (2004, July). *In display time as implicit feedback: Understanding task effects.* Paper presented at the Twenty-seventh Annual International Conference on Research and Development in Information Retrieval, Sheffield, England.

Kelly, D., & Teevan, J. (2003). Implicit feedback for inferring user preference: A bibliography. *SIGIR Forum*, *37*(2), 18-28.

Khopkar, Y., Spink, A., Giles, C. L., Shah, P., & Debnath, S. (2003). Search engine personalization: An exploratory study [Electronic Version]. *First Monday*, 8. Retrieved October 20, 2007, from http://firstmonday.org/issues/issue8_7/khopkar/index.html

Koenemann, J., & Belkin, N. (1996, April). *A case for interaction: A study of interactive information retrieval behavior and effectiveness.* Paper presented at the ACM SIGCHI '96 Conference on Human Factors in Computing Systems, Vancouver, Canada.

Lalmas, M., & Ruthven, I. (1999, May). *A framework for investigating the interaction in information retrieval.* Paper presented at the Ninth European-Japanese Conferences on Information Modeling and Knowledge Bases, Japan.

Lin, S. J. (2002, August). *Design space of personalized indexing: Enhancing successive web searching for transmuting information problem,* Paper presented at the American Conference on Information Systems, Dallas, TX.

Miller, G. (1998). *WordNet: An electronic lexical database.* Cambridge, MA: MIT Press.

Mitra, M., Singhal, A., & Buckley, C. (1998, August). *Improving automatic query expansion.* Paper presented at the Twenty-first Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Melbourne, Australia.

Oard, D., & Kim, J. (2001). Modeling information content using observable behavior. *Proceedings of the Sixty-fourth American Society for Information Science Annual Meeting*, *38*, 481-488.

Park, S., Bae, H., & Lee, J. (2005). End user searching: A Web log analysis of NAVER, a Korean Web search engine. *Library and Information Science Research*, *27*(2), 203-221.

Proctor, E. (2002). Boolean operators and the naive end-user: Moving to AND. *Online*, *26*(4), 34-37.

Saracevic, T. (1996, October). *Modeling interaction in information retrieval (IR): A review and proposal.* Paper presented at the Fifty-ninth American Society for Information Science, Baltimore, MD.

Saracevic, T. (1997). The stratified model of information retrieval interaction: Extension and applications. *Proceedings of the Sixtieth American Society for Information Science Annual Meeting*, *34*, 313-327.

Saracevic, T., & Kantor, P. (1997a). Studying the value of library and information services: I. Establishing a theoretical framework. *Journal of the American Society for Information Science*, *48*(6), 527-542.

Saracevic, T., & Kantor, P. (1997b). Studying the value of library and information services: II. Methodology and taxonomy. *Journal of the American Society for Information Science*, *48*(7), 543-563.

Saracevic, T., Kantor, P., Chamis, A. Y., & Trivison, D. (1988). Study of information seeking and retrieving: I. Background and methodology. *Journal of the American Society for Information Science*, *39*(3), 161-176.

Saracevic, T., Mokros, H., Su, L., & Spink, A. (1991). Interaction between users and intermediaries during online searching. *Proceedings of the Twelfth Annual National Online Meeting*, *12*, 329-341.

Seo, Y. W., & Zhang, B. T. (2000, June). *Learning users' preferences by analyzing Web-browsing behavior.* Paper presented at the Fourth International Conference on Autonomous Agents, Barcelona, Spain.

Shen, X., Tan, B., & Zhai, C. (2005, August). *Context-sensitive information retrieval using implicit feedback.* Paper presented at the Twenty-eighth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Brazil.

Silverstein, C., Henzinger, M., Marais, H., & Moricz, M. (1999). Analysis of a very large Web search engine query log. *SIGIR Forum, 33*(1), 6-12.

Spink, A. (2004). Multitasking information behavior and information task switching: An exploratory study. *Journal of Documentation*, *60*(3), 336-345.

Spink, A., Jansen, B. J., Wolfram, D., & Saracevic, T. (2002). From E-sex to E-commerce: Web search changes. *IEEE Computer*, *35*(3), 107-111.

Spink, A., & Saracevic, T. (1997). Interaction in information retrieval: Selection and effectiveness of search terms. *Journal of the American Society for Information Science*, *48*(8), 741-761.

Spink, A., Wilson, T., Ellis, D., & Ford, F. (1998). Modeling Users' Successive Searches in Digital Environments study [Electronic Version]. *D-Lib Magazine.* Retrieved October 20, 2007, from http://www.dlib.org/dlib/april98/04spink.html

Sun, T. (1971). *The Art of War* (S. B. Griffith Trans.). New York: Oxford University Press. (Original work published n.d.)

Hallam-Baker, M. P. & Behlendorf, B. (n.d.). *Extended log file format.* Retrieved October 15, 2007, from http://www.w3.org/TR/WD-logfile.html

Wang, P., Berry, M., & Yang, Y. (2003). Mining longitudinal Web queries: Trends and patterns. *Journal of the American Society for Information Science and Technology*, *54*(8), 743-758.

White, R. W., Ruthven, I., & Jose, M. J. (2001, November). *Comparing implicit and explicit feedback techniques for Web retrieval: TREC-10 interactive track report.* Paper presented at the Tenth Text Retrieval Conference (TREC-10), Gaithersburg, MD.

Wilson, T. D. (1999). Models in information behaviour research. *Journal of Documentation*, *55*(3), 249-270.

Yee, M. (1991). System design and cataloging meet the user: User interfaces to online public access catalogs. *Journal of the American Society for Information Science*, *42*(2), 78-98.

Zipf, G. K. (1949). *Human behavior and the principle of least effort.* Cambridge, MA: Addison-Wesley Press.

KEY TERMS

Action: An action is a specific utterance of the user.

Action Object (*a*, *o*) Pair: In (*a*, *o*) pair, *a* stands for action and *o* stands for object.

Action-Object Pair Approach: One (*a*, *o*) pair is one interaction between the user and the system. A series of (*a*, *o*) pairs or *a*-*o* matrix can represent the interaction session, which is defined as a series of interactions between the user and the system to fulfill the user's certain information need.

Object: An object is a self-contained information object, the receipt of the action.