

A User-Tracing Architecture for Modeling Interaction with the World Wide Web

Peter Pirolli
pirolli@parc.xerox.com

Wai-Tat Fu¹
wfu@gmu.edu

Robert Reeder²
reeder+@cs.cmu.edu

Stuart K. Card
card@parc.xerox.com

PARC
3333 Coyote Hill Road
Palo Alto, California 94304

ABSTRACT

To construct detailed models of the psychology of users interacting with the World Wide Web (WWW) we have developed a methodology for studying and analyzing ecologically valid WWW tasks. The methodology involves creating a user trace: a record of all significant states and events in the user-WWW interaction based on the analysis of eye tracking data, application-level logs, and think-aloud protocols. A user-tracing architecture has been implemented for developing simulation models of user-WWW interaction and for comparing simulation models against user-trace data. The simulation model, called SNIF-ACT (Scent-based Navigation and Information Foraging in the ACT theory) is given the same tasks as observed users, and then the model simulates activity with the WWW to achieve those tasks. The user tracing architecture compares each action of the SNIF-ACT simulation directly against observed user actions. The model and architecture has been used to successfully match detailed user trace data from four users working on two tasks each.

CATEGORIES AND SUBJECT DESCRIPTORS

H.52 [User Interfaces]: Theory and methods

GENERAL TERMS

Measurement, Human Factors, Theory

KEYWORDS

User models, ACT-R, information foraging, World Wide Web, SNIF-ACT, user tracing.

1. INTRODUCTION

The 1980s witnessed a confluence of increased computing power, storage capacity, and networking, along with innovations in information access and hypermedia. These developments lead to the release of the World Wide Web (WWW) in 1991, shortly after its initial proposal by Berners-Lee [1]. Despite a decade of widespread use, there has been limited progress towards a deep scientific understanding of the psychology of human-WWW interaction. Partly this is because analysis of user behavior in terms of detailed cognitive models requires the labor-intensive

analysis of overwhelming amounts of data, and partly it is because unlike most models in human-computer interaction, the analysis of human-WWW interaction requires modeling user interaction with the semantics of Web content. Neither of these has been practical.

The purpose of this paper is to present a *user-tracing architecture* for developing scientific models of user interaction with the WWW. This architecture enables the development of computational models of users that can be tested against datasets that contain *user traces*: transcriptions or recordings of user interaction with interfaces to the WWW. It is used for developing and testing *cognitive-perceptual simulation models* by matching the output of the model against the traces. The paper presents the user-tracing architecture in the context of a particular cognitive-perceptual model, SNIF-ACT, being developed for modeling user-Web interaction. The aim of this model is to characterize user link choices when interacting with Web pages as well as interaction times, errors, and eye gaze patterns based on the words used to describe the links, layout of the page, user working memory and visual attention patterns. We expect such a model to help explain such observed phenomena as lostness and stickiness.

Web interaction is an instance of information-intensive work, interaction for a purpose with large amounts of information. We have proposed an approach to such tasks using information foraging theory [2, 3], analyzing them as an attempt by a user to maximize information gained per unit time. This is a theory at the adaptationist level, where we predict what actions it would be beneficial for the user to take based on the task environment. The SNIF-ACT model is a complementary model at the lower proximal mechanisms level, a detailed mechanistic account of information foraging, taking into account cognitive and perceptual limitations of the user to predict what actions the user actually will take.

Before it is possible to develop a cognitive model at the level of SNIF-ACT, however, some means must be found to analyze detailed traces of user interaction in a way that helps the analyst build the model and a means must be found to compare the model against user traces. The problem is especially difficult because only some of the predicted operations will result in events captured on the trace; other operations will be internal mental events. The model must be evaluated against this partial

¹ Present address: George Mason University, MS 3e5 4000 University Drive, Fairfax, VA 22030.

² Present address: Carnegie Mellon University, 4000 Forbes Avenue, Pittsburgh, PA 15213

trace to evaluated for fit to the data. This paper reports on a method for doing this.

2. USABILITY AND THE WWW

One reason that the WWW poses problems for browser design and WWW design is the enormous growth of WWW content. It is estimated that the average user in 2001 had access to 525 billion WWW documents (See the Hobbes' Internet Timeline at <http://www/isoc/guest/zakon/Internet/History/HIT.html>). In such an information-rich world, the real design problem to be solved is not so much how to provide access to more information, but rather, how to increase the amount of relevant information encountered by a user as a function of the user's interaction time. Studies show the Web has major usability problems. It has been estimated that 65% of virtual shopping trips on the WWW end up in failure [4] and 40% of users do not return because of problems in WWW site design [5].

In reaction to the usability problems of the WWW, there have been many attempts to understand WWW users and to develop WWW usability methods. Empirical studies [6] have reported general patterns of information seeking behavior, but have not provided much in the way of detailed analysis. WWW usability methodologists [7-10] have drawn on a mix of case studies and empirical research to extract best design practices for use during development as well as evaluation methods for identifying usability problems [11]. For instance, principles regarding the ratio of content to navigation structure on WWW pages [9], the use of information scent to improve WWW site navigation [12], reduction of cognitive overhead [8], writing style and graphic design [7], and much more, can be found in the literature. Unfortunately, these principles are not universally agreed upon and not universally applicable. For instance, there is debate about the importance of download time as a usability

factor [12]. There is even debate about how to determine the number of users that are required for testing [13]. Consequently, all usability methodologists advocate the use of user studies and design validations which may include field studies, user interviews, surveys, task analysis, focus groups, and direct testing with users [7]. Such methods can identify requirements and problems with specific designs, and may even lead to some moderately general design practices, but they are not aimed at the sort of deeper scientific understanding that may lead to large improvements in WWW interface design.

We believe that the development of theory in this area can greatly accelerate progress and meet the demands of changes in the way we interact with the WWW [14]. Greater theoretical understanding and the ability to predict the effects of alternative designs could bring greater coherence to the usability literature, and provide more rapid evolution of better designs. In practical terms, a designer armed with such theory could explore and explain the effects of different design decisions on WWW designs before the heavy investment of resources for implementation and testing. The exploration of the design space is also more efficient because the choices among different design alternatives are better informed: Rather than randomly generating and testing design alternatives, the designer is in a position to know which avenues are better to explore and which are better to ignore. Theory and scientific models themselves may not be of direct use to engineers and designers, but they form a solid and fruitful foundation for design models and engineering models [15, 16].

Unfortunately, cognitive engineering models that had been developed to deal with the analysis of expert performance on well-defined tasks involving application programs [17] have had limited applicability to understanding foraging through content-rich hypermedia, and consequently new theories are needed. One attempt [18] at developing a GOMS model of WWW users [15] failed to have any correlation at all with user behavior [19]. One reason is probably that the model's behavior was based purely on the structure of pages and links—none of the behavior was determined by the semantics of page content. A day-in-the-life protocol analysis of real users engaged in their own tasks with the WWW [20] showed that the majority of user time was devoted to processing WWW content. So, a model of WWW users will have to deal with how user behavior depends on content. Chi and others in our group have developed a model of "information scent" that can predict surfing patterns of users looking for information on a site [21]. This model does not, however, deal with the cognitive or perceptual mechanisms involved.

3. A USER TRACING ARCHITECTURE

A possible method of developing the type of computational cognitive-perceptual models we need is to make models with stochastic parameters and run them Monte Carlo style, comparing their aggregated output against aggregated groups of users. This is a good method for exploring the effect of various inventions at the browser level or at the level of design of websites or web pages, but it averages out information related to the cognitive and perceptual mechanism at work needed to develop and validate these models.

Here we develop an alternative method aimed at extracting and validating information and an individual user level, depicted schematically in Figure 1. The components of the user tracing architecture are:

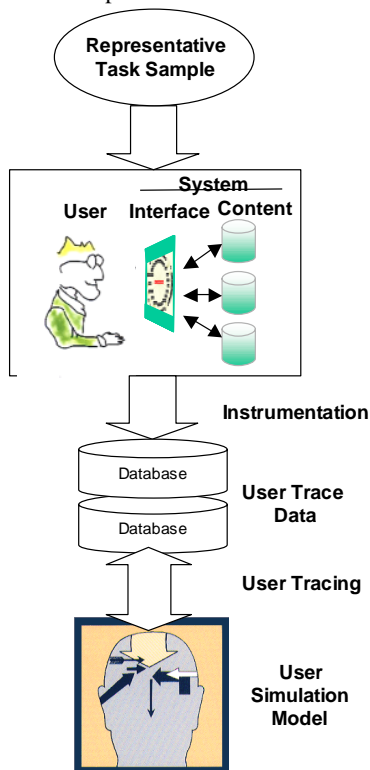


Figure 1. The user tracing architecture.

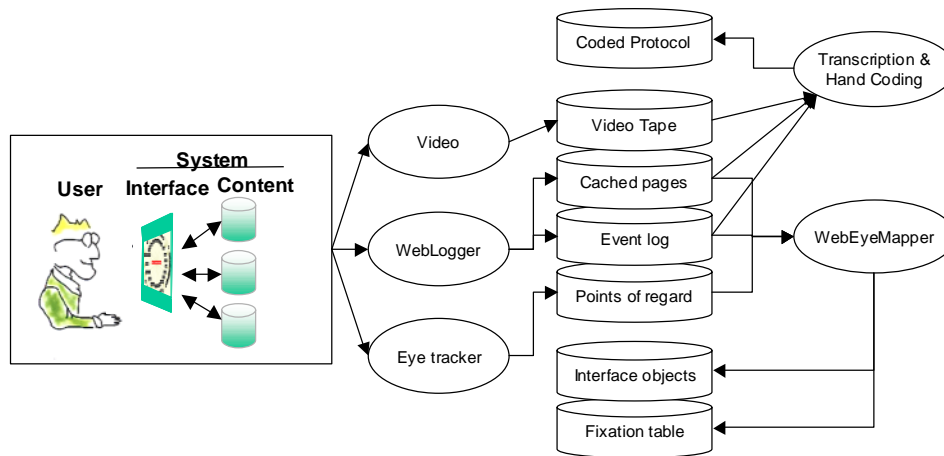


Figure 2. User tracing instrumentation and databases.

1. *Tasks.* Users are given a set of written Web tasks to do. These same written tasks will be given to the model. The user does the tasks using a browser connected to the WWW.
2. *Instrumentation.* The browser is instrumented to produce a trace of behavior and the user talks aloud while performing the task. All the Web pages accessed are also saved away. Human-friendly representations of this trace are produced to aid the theorist in building the models and in coding the spoken transcripts. The result is a set of databases containing user traces and associated data.
3. *Cognitive-perceptual stimulation model.* A user simulation model is constructed (or probably refined).
4. *User Comparator.* The model is run in the user trace architecture. On each cycle, the model makes a prediction, generating another element in the trace, which will involve accessing the saved Web pages. The user trace comparator uses a set of rules to determine whether there is a match with the protocol trace; if not, an error is scored against the model and it is set back on track.

We now consider these components in greater detail.

4. TASKS AND INSTRUMENTATION

Our method begins with the construction of ecologically valid tasks—that is, tasks that resemble “what people do in real, culturally significant situations” [22]. Controlled laboratory experiments are then conducted using tasks derived from this task database. The laboratory experiments collect data using an

eye tracker, logging software that collects all user interactions with a WWW browser, and video recordings of think-aloud verbal protocols [23]. These data are coded by automatic means and by hand into a comprehensive trace of states and events representing the interaction of user with the WWW. Computational models of user cognition and perception are then developed to simulate—as accurately as possible—the observed user-WWW interactions.

4.1. Tasks

Tasks for our study were selected from a database collected by a survey of over 2000 WWW users [24]. Selected tasks focused on finding some specific information, such as the dates for an upcoming theatre event, or specific items, such as posters of characters that appeared in a recent movie. For instance, one of the tasks was:

Antz Task: After installing a state of the art entertainment center in your den and replacing the furniture and carpeting, your redecorating is almost complete. All that remains to be done is to purchase a set of movie posters to hang on the walls. Find a site where you can purchase the set of four Antz movie posters depicting the princess, the hero, the best friend, and the general.

Users were encouraged to perform the tasks as they would typically, but they were also instructed to think out loud [25] as they performed their tasks.

4.2. Instrumentation

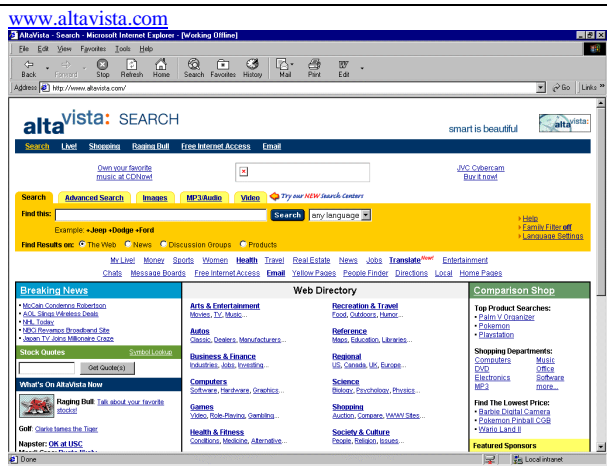
(BEFORE-NAVIGATE	(http://altavista.com/)	105.331s	0.100s	951763010	10:36:50)
(DOC-MOUSEMOVE	(881 122)	105.431s	0.100s	951763010	10:36:50)
(NAVIGATE-COMplete	(http://www.altavista.com/)		105.632s	0.201s	951763011	10:36:51)
(EYETRACKER-SYNC	(103)	106.242s	0.610s	951763011	10:36:51)
(DOCUMENT-COMplete	(http://www.altavista.com/)		106.773s	0.531s	951763012	10:36:52)
(SCROLL-POSITION	(0 0 759 1181)	106.853s	0.080s	951763012	10:36:52)
(DOC-MOUSEMOVE	(874 123)	107.024s	0.171s	951763012	10:36:52)
(DOC-MOUSEMOVE	(874 123)	107.044s	0.020s	951763012	10:36:52)
(DOC-MOUSEMOVE	(874 123)	107.214s	0.170s	951763012	10:36:52)
(EYETRACKER-SYNC	(104)	107.244s	0.030s	951763012	10:36:52)
(CHAR	(a	874 123)	108.125s	2.904s	951763013 10:36:53)
(EYETRACKER-SYNC	(105)	108.245s	1.001s	951763013	10:36:53)
(DOC-KEYPRESS	(a INPUT)	108.446s	0.201s	951763013	10:36:53)

Figure 3. WebLogger event log fragment.

Performance on the tasks was recorded using an instrumentation package (Figure 2) that included: (a) WebLogger [26], which is a program that tracks user keystrokes, mouse-movements, button use, and browser actions, (b) an eye tracker, and (c) video recordings that focused on the screen display. Details of the instrumentation used are given in [23]. Keystrokes, mouse-movement, browser controls, and browser actions were recorded using our WebLogger system. Eye-movements are handled by our WebEyeMapper system. At *experiment-time*, a user works on a specific task with a WWW browser (Figure 2). As the user performs various actions with the browser, including navigating to different pages and scrolling within pages, these events are recorded on videotape, by the WebLogger program, and by an eye tracker resulting in the databases enumerated in the figure. WebLogger, whose development was inspired by the development of remote usability evaluation tools in [27], instruments the WWW browser and records all significant events and display states. The program works with the Windows NT/2000 operating system and Microsoft®'s Internet Explorer (IE) Web browser.

When WebLogger starts, it simultaneously launches an instance of IE. It produces a text file, the *event log* (Figure 3), which documents user and application events that occur during a user's interaction with this instance of IE. Each event has an event type, parameters significant to that event, and clock times recorded in several convenient formats. WebLogger events include browser events related to the state of the display. Logged browser events include events that change the Web page displayed, the portion of the page displayed, or the position or size of the IE window relative to the screen. WebLogger also has a content-saving feature. It can save the actual Web content (i.e. the text, images, scripts, etc.) that a user looked at during a browsing session. It does this by saving a cache of all pages and associated content that was viewed by the user. Because of the ever-changing nature of the Web, it is difficult to recall the exact content that a user saw during a browsing session, but WebLogger's content-saving feature makes this possible. In sum, WebLogger records all the application-level *content elements* of interest including the Web page elements that the browser renders on the display – text, images, hyperlinks, buttons, input

Table 1. A WWW Protocol Transcript for the initial portion of S1's work on the Antz task.

	SYSTEM	OBSERVED ACTIONS AND TRANSCRIPT	MODEL INTERPRETATION
4:25:00		(Task #2:ANTZ) (Reads Question) Ok, Um.	(O*READ-QUESTION)
		Let's go to Altavista because I like Altavista.	(G*GO-TO SITE "Altavista")
4:49:22		Click: www.altavista.com in pulldown menu Um, and let's look up Antz. And see what's generally available.	(O*GO-TO SITE "www.altavista.com") (G*SEARCH WEB www.altavista.com "Antz")
		Type search: Antz	(O*SEARCH WEB "www.altavista.com" "Antz")
4:55:00	http://www.altavista.com/cgi-bin/query?pg=q&sc=on&hl=on&q=antz&kl=XX&stpe=stext	"The official Antz web ring," that sounds appropriate.	(O*EVAL LINK "The official Antz web ring" hi)
		Click: The Official ANTZ Webring link	(O*FOLLOW LINK "The official Antz web ring")
5:00:07	http://www.geocities.com/Area51/sShire/3303/ANTZ/webring.html	Uh, "list of sites,	(O*EVAL LINK "list of sites" null)
		random sites,	(O*EVAL LINK "random sites" null)
		how to join."	(O*EVAL LINK "how to join" null)
		Let's have a look at the list of sites.	(G*FOLLOW LINK "list of sites")
5:06:19	http://www.webring.org/cgi-bin/webring?ring=z4195:list	Click: A List of Sites link "The anomaly, a few MX goodies mixed in,"	(O*FOLLOW LINK "list of sites") (O*EVAL LINK "The Anomaly" null "a few MX goodies mixed in")

boxes, etc. From these data we know what content was displayed at what display location at every point in time throughout the user’s task.

At *analysis-time*, one basic problem is that eye tracking data must be mapped onto data recorded by WebLogger in order to determine on what content the user was visually focused at any given time, even though the user might be scrolling the window or moving it on the screen. This is called the *points-to-elements mapping* problem [26] and is solved by the WebEyeMapper program (Figure 2). Every 1/60th second, the eye tracker records the *point-of-regard* of the eye, the inferred x,y screen point at which the eye is gazing. In order to analyze data from a user’s browsing session, WebLogger launches an instance of Internet Explorer and maintains a pointer to the instance of Internet Explorer. Using WebLogger event log data, raw eye tracker data, and content from WebLogger’s content-saving feature, WebEyeMapper begins a “playback” of a browsing session. WebEyeMapper maintains an analyst-controlled simulation clock to coordinate the replay of WebLogger events and eye fixations. As the simulation clock advances, WebEyeMapper directs Internet Explorer to load the same Web pages that the user was viewing at the time indicated by the simulation clock, and directs Internet Explorer to alter its scroll position, window position, and window size as the user did at experiment-time. In this manner, WebEyeMapper restores the display state of the browser to the same state, moment-by-moment, as the user viewed it at experiment-time. WebEyeMapper can then take eye fixation points, align them in time with the simulation clock, align them in space with the browser window, and determine what is rendered in the browser at the time of each fixation. For each fixation, WebEyeMapper writes the fixation start time and duration, screen, window, and scroll system coordinates, element fixated, and element text fixated, to a database.

4.3. Protocol Analysis

Videotapes of users thinking aloud provide additional data about users’ goals and subgoals, attention, and information representation [25]. WebLogger and WebEyeMapper data are used to produce (a) a Web Behavior Graph (WBG) (See Figure 4) and (b) a Web Protocol Transcript (see Table 1).

A Web behavior graph is an application to WWW behavior of a problem behavior graph (Newell and Simon, 1972) and visualizes user behavior as a search through a problem space. Each box in the diagram represents a state in one of several problem spaces. Each arrow depicts the execution of an operator, moving the state to a new state. Double vertical arrows indicate the return to a previous state, augmented by the experience of having explored the consequences of some possible moves. Thus, time in the diagram proceeds left to right and top to bottom.

The WBG is particularly good at showing the structure of the search. Color surrounding the boxes in the diagram represents different Web sites. Oval boxes are distinguished in order to show hit lists from a search. An **X** following a node indicates that the user exceeded the time limits for the task and that the task was therefore a failure. A loop has been drawn around different problem spaces, showing how the users pass from one problem space into another as the operators in one become less effective. Evident in Figure 4 is the hub-and-spoke structure of the behavior, in which the user follows a trail out from the hit list of a search until that trail goes cold, retreats to the hit list page, and finds another link to try.

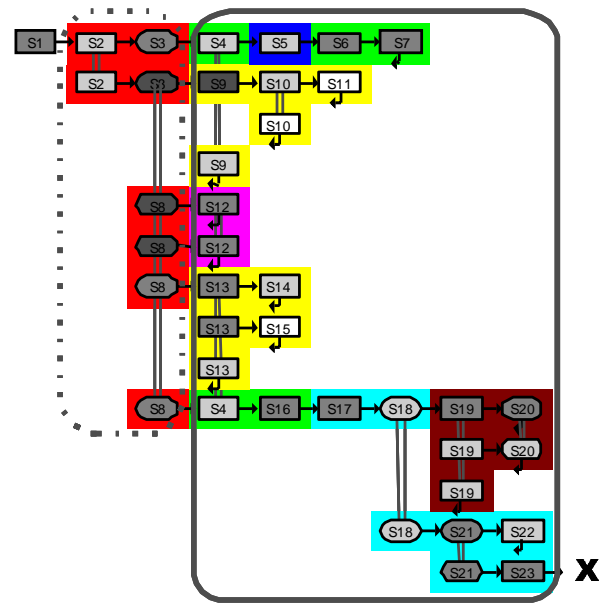


Figure 4. A Web Behavior Graph for user S1 working on the Antz task.

The Web Protocol Transcript (Table 1) starts with a selection of interactions recorded by the WebLogger and adds to these (a) transcribed data, including viewed URLs, eye movements, verbalizations, and observed actions, which are presented side-by-side with (b) a *model coding* of the inferred cognitive action that is associated with the data.³ Table 1 is a Web Protocol Transcript for the initial half minute (approximately) of WWW interaction by a user (S1) performing the Antz task described above. This task took just over 10 minutes. The System column records significant IE events, which includes which WWW pages were visited. The Observed Actions and Transcript column contains a transcript of user verbalizations and actions. The Model Coding column contains codes that indicate an analyst’s interpretation of cognitive events based on the data in the columns to the left. Using a standardized WWW Protocol Coding Guide,⁴ we have found the inter-coder reliability to be 0.93. Each row of Table 1 corresponds to a significant, codeable, cognitive event. More details on WBGs and Web Protocol Transcripts are presented in Card et al. [23]

If one reads down the Observed Actions and Transcript column in Table 1, one can see that during this initial portion of the task, the user (S1) read the task question, decided to start at the AltaVista search engine, navigated there using a bookmark, and entered a search query. S1 then scanned the results page and clicked on a link that took S1 to another page (“Official Antz Web ring”). Then S1 scanned that page and clicked on a link for another page (“List of sites”). The WWW Protocol Coding guide provides a detailed explanation of the codes presented in Table 1. For example, in Table 1, the code

(G*SEARCH WEB www.altavista.com “Antz”),

³ The Web Protocol Transcript presented in Table 1 omits some details for the purposes of presentation here. For instance, we have only included one WWW page image here, but the original transcript contains an image on every line of the transcript where a URL is listed in the System column. We have also omitted several timing indicators used to coordinate the transcript with the WebLogger files.

⁴ The WWW Protocol Coding Guide is available from the authors.

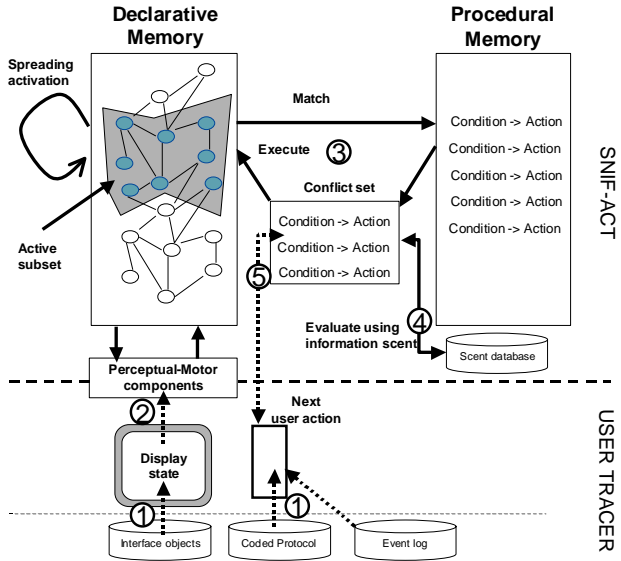


Figure 5. The SNIF-ACT user modeling architecture (top) builds on the ACT-R theory. The user tracer architecture (bottom) the SNIF-ACT simulation and compares it to user trace data. See text for details.

is a code that is defined in the WWW Protocol Coding Guide by the entry

(G*SEARCH structure-type structure need)	Indicates the goal of using a search engine to search some kind of <i>structure-type</i> (e.g., the web; a page), the particular <i>structure</i> searched, and an indication of the <i>need</i> . Note: If info-need is implicit, and not directly stated by the user, use the placeholder "null" for that info-need.
--	--

In general, the prefix "G*" codes that the user has indicated that they have some goal. The prefix "O*" indicates an operator, which is an elementary cognitive, perceptual, or motor act that changes the user's mental state or task environment. In all, the WWW Protocol Coding Guide currently contains codes related to the goals and operators for dealing with the task (such as reading or recalling the task description), navigation and search (such as clicking on links, using search engines), evaluations of content (such as links and pages), reformulating information needs, and making mental notes about the encountered content.

The protocol analysis provides data that are not available from WebLogger and WebEyeMapper. This can be seen in Table 2. The first column of Table 2 contains the Model Interpretation from Table 1. The second column of Table 2 contains actions that can be directly parsed from the WebLogger event log (indicated by the prefix "wla*" for "WebLogger action"). Note that the protocol coding picks up the user's reading and judgment of links (indicated by the code O*EVAL), which has no correspondence in the WebLogger database. Also, users will often state their goals, plans, or heuristics, such as which sites they will visit and what sorts of pages they might be seeking. These verbalizations yield insights into user cognition that could not possibly be recorded by WebLogger or an eye tracker.

5. COGNITIVE-PERCEPTUAL SIMULATION MODEL

As indicated in Figure 2, the outputs of WebLogger, WebEyeMapper, and protocol analysis are a set of databases that include the Web Protocol Transcript, WebLogger event log, WebLogger cached pages, WebEyeMapper element database, and the WebEyeMapper fixation table. These databases constitute the data that provide us with a user trace: a detailed analysis of all the states and events that correspond to a user's performance of a task using the WWW. The goal of our modeling effort is to develop a computer program that simulates the user in enough detail to reproduce the same user trace data.

5.1. SNIF-ACT Model

SNIF-ACT is the model that we are currently developing to simulate WWW users. To understand how the model tracing architecture works, it is necessary to understand the structure of this type of model. SNIF-ACT is an extension of the ACT-R theory and simulation environment [28], a general *production system*-based architecture designed to model human psychology. By using this system to model Web behavior, we link our analysis to the same principles used to model cognitive behavior in general. ACT-R contains principles concerning: (1) knowledge representation, (2) knowledge deployment (performance), and (3) knowledge acquisition (learning). ACT-R has been recently extended to the analysis of visual attention and other perceptual-motor processes. There are two major components in the ACT-R architecture: a *declarative knowledge* component and a *procedural knowledge* component (Figure 5). Declarative knowledge corresponds to things that we are aware we know and that can be easily described to others, such as the content of WWW links, or the functionality of browser buttons. Declarative knowledge is represented formally as *chunks* in ACT-R. Procedural knowledge is knowledge (skill) that we display in our behavior without conscious awareness, such as knowledge of how to ride a bike, or how to point a mouse to a menu item. Procedural knowledge specifies how declarative knowledge is transformed into active behavior. ACT-R has two kinds of memory for these two different kinds of knowledge.

Declarative chunks in ACT-R have sub-symbolic *activation* values (This part of ACT-R is like a connectionist model). Activation may be interpreted metaphorically as a kind of mental energy that drives cognitive processing. Activation spreads from the current focus of attention, including goals, through *associations* among chunks in declarative memory. These associations are built up from experience, and they reflect how ideas co-occur in cognitive processing. Generally, activation-based theories of memory predict that more activated knowledge structures will receive more favorable processing. Chunks with higher activation values take less time to use (and have a greater chance) to have an impact on behavior. Activation is a way of quantifying the degree of relevance of declarative information to the current focus of attention. At any point in time, there is a stack of goals encoding the user's intentions. Goals are also represented as chunks. ACT-R is always trying to achieve the goal that is on top of that stack, and at any point in time, it is focused on a single goal. Figure 6 provides a graphical representation of the some of the declarative memory chunks that appear in the SNIF-ACT model of the user trace that corresponds to Table 1. The notation roughly corresponds to the network graphs of human memory presented in standard textbooks [29]. Each oval represents a chunk. Arrows indicate labeled relations

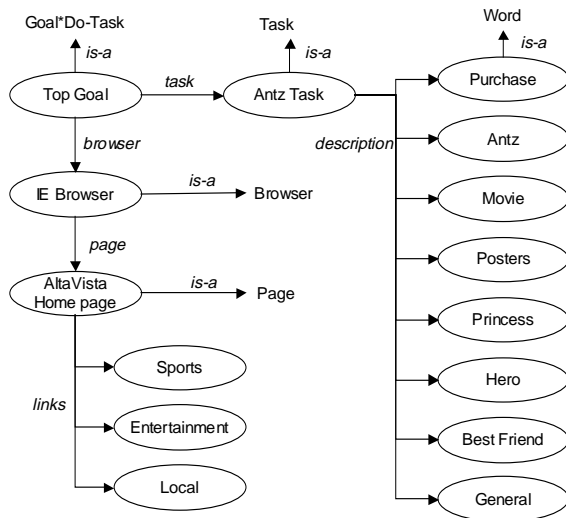


Figure 6. A subset of the declarative knowledge in the SNIF-ACT simulation of user S1 working on the Antz task. Chunks representing the top goal, task description, and browser display are depicted.

among chunks, or indicate the type of chunks using the label “is-a.” The chunk Top-goal represents he user’s main goal, and it is related (*points*) to the chunk “Antz task” which points to a description of the concepts involved in the task. The Top-goal chunk also points to a chunk representing the user’s perception of the IE Browser, which points to the user’s perception of the AltaVista home page that is displayed in the browser and the links that have been perceived on that page.

Procedural knowledge is represented as condition-action pairs, or *production rules* (Figure 5). For instance, our SNIF-ACT simulation of user S1 on the Antz task contains the production rule (summarized in English):

```

Use-Search-Engine:
IF  the goal is Goal*Start-Next-Patch
    & there is a task description
    & there is a browser
    & the browser is not at a search engine
THEN
    Set a subgoal Goal*Use-search-engine
  
```

The production (titled Use-search-engine) applies in situations where the user has a goal to go to a WWW site (represented by the tag Goal*Start-Next-Patch), has processed a task description, and has a browser in front of them. The production rule specifies that a subgoal will be set to use a search engine. The condition (IF) side of the production rule is matched to the current goal and the active chunks in declarative memory, and when a match is found, the action (THEN) side of the production rule will be executed. Roughly, the idea is that each elemental step of cognition corresponds to a production. At any point in time, a single production fires. When there is more than one match, the matching rules form a *conflict set*, and a mechanism called *conflict resolution* is used to decide which production to execute (see Figure 5). The conflict resolution mechanism is based on a utility function. The expected utility of each matching production is calculated based on this utility function, and the one with the highest expected utility will be picked. In modeling WWW users, the utility function is provided by information foraging theory, and specifically the notion of information scent

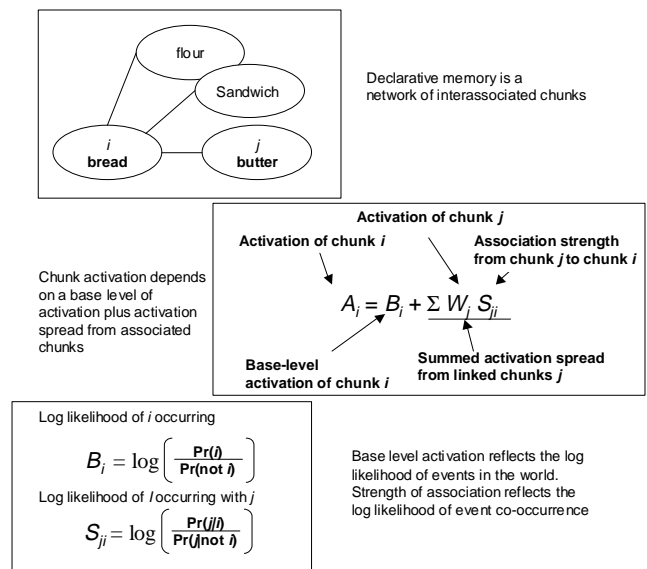


Figure 7. Strength and spreading activation.

[3]. This constitutes a major extension of the ACT-R theory and is described in greater detail below.

5.2. Information Scent

As users browse the WWW, they make judgments about the utility of different courses of action available to them. Typically, they must use local cues, such as link images and text, to make navigation decisions. Information scent refers to the local cues that users process in making such judgments. The analogy is to organisms that use local smell cues to make judgments about where to go next (for instance in pursuing some prey). In earlier work [3, 30] we extended ACT-R to produce a theory called ACT-IF (where IF stands for “information foraging”). ACT-IF included a formal model of information scent that predicted how users would use text presented in browsers to make navigation decisions. The model of users’ judgments of information scent is based on spreading activation.

A specific version of spreading activation [28] uses mechanisms based on the analysis of the requirements of an optimal memory system [31, 32]. This is the version of spreading activation used to develop a model of information scent. The basic idea is that a user’s information goal activates a set of chunks in a user’s memory, and text on the display screen activates another set of chunks. Activation spreads from these chunks to related chunks in a *spreading activation network*. The amount of activation accumulating on the goal chunks and display chunks is an indicator of their mutual relevance. The amount of activation is used to evaluate and select productions. The activation of chunks matched by production rules can be used to determine the utility of selecting those production rules. For instance, the following Click-link production rule matches when a WWW link description has been read,

```

Click-link:
IF  the goal is Goal*Process-element
    & there is a task description
    & there is a browser
    & there is a link that has been read
    & the link has a link description
THEN
    Click on the link
  
```

If selected, the rule will execute the action of clicking on the link. The chunks associated with the task description and the link description will have a certain amount of activation. That combined activation will be used to evaluate the rule. If there are two Click-link productions matching against chunks for two different links, then the one with more highly activated chunks will be selected. As we describe next, the activation level will tend to reflect the degree of relevance of the link text to the task description.

The spread of activation from one cognitive structure to another is determined by weighting values on the associations among chunks. These weights determine the rate of activation flow among chunks. Figure 7 presents a summary of the rational analysis and computation of spreading activation in ACT-R. Figure 8 presents a scenario for a spreading activation analysis. Suppose a user is looking for “information on new medical treatments and procedures for cancer” using a WWW browser. The representation of this goal in declarative memory is depicted by the small set of concepts linked to the concept “Cancer Task” in Figure 8 (the nodes labeled NEW, MEDICAL, TREATMENTS, PROCEDURES, CANCER). (For the purposes of presentation, this is basically a reduced version of a network such as the one presented in Figure 6). These nodes represent the main meaningful concepts making up the users’ goal. In Figure 8, the user is looking at a browser that has retrieved a set of documents. One of the WWW links is summarized by the text “cell, patient, dose, beam, cancer”. The representation of this part of the browser display is depicted by the network of nodes linked to “WWW LINK” in Figure 8. Figure 8 also shows that there are links between the goal concepts and the text summary concepts. These are associations between words that come from past experience. The associations reflect the fact that these words co-occur in the users’ linguistic environment. For instance, the word “medical” and “patient” co-occur quite frequently and they would have a high weighting of inter-association. Spreading activation would flow from the goal, which is the focus of attention, through the inter-word associations, to words in the link summary. The stronger the associations (higher weights that reflect higher rates of co-occurrence) the greater the amount of activation flow. If the goal and link text are strongly associated, then we expect people to judge them as being highly relevant to one another. At least implicitly, this is what the interface designers of browsers are trying to do when they select small text snippets to communicate the content of large documents to users. They are trying to pick words that people will judge as relevant to their queries. Spreading activation may be used to predict these memory-based judgments of reminding and relevance that are key components of surfing the World Wide Web. A significant extension of the ACT-R theory is the use of information scent, as calculated by spreading activation, to evaluate and select production rules.

As indicated in Figure 7, the SNIF-ACT information scent computation requires estimates of the probabilities of word occurrences and of word co-occurrences. In past research [3, 30], we derived these estimates from the Tipster corpus [33].⁵ This database contained frequency of occurrence and frequency of word co-occurrence data collected from 742,833 full-text documents (mostly news). This database contained statistics relevant to setting the base-level activations of 200 million word tokens and the inter-word association strengths of 55 million

Table 2. Correspondence of SNIF-ACT production firings to the Web Protocol Transcript (Table 1) and WebLogger events for user S1 working on the Antz task.

<i>Model Coding (from Table 1)</i>	<i>WebLogger Actions</i>	<i>Production firings (from Figure 9)</i>
(O*READ-QUESTION)		Read-question
		Start-task
		Look-at-new-page
		Use-search-engine
		Go-to-search-engine
(O*GO-TO SITE "www.altavista.com")	(wla*go-to)	Go-to-site-by-bookmark
		Look-at-new-page
(O*SEARCH WEB "www.altavista.com" "Antz")	(wla*search ...)	Search-using-engine
		Look-at-new-page
		Start-process-page
		Process-element-on-page
		Attend-to-element
(O*EVAL LINK "The official Antz web ring" hi)		Read-and-interpret
(O*FOLLOW LINK "The official Antz web ring")	(wla*follow ...)	Click-link

word pairs. Unfortunately, the Tipster corpus does not contain many of the novel words that arise in popular media such as the WWW. For instance, the movie title “Antz” does not occur in the Tipster corpus. Consequently, we augment the statistical database derived from Tipster by estimating word frequency and word co-occurrence statistics from the WWW itself using a program that calls on the AltaVista search engine to provide data. As indicated in Figure 5, the statistics needed to perform the scent computations are stored in a *scent database* that is accessed when production evaluations are computed by SNIF-ACT.

6. USER TRACE COMPARATOR

6.1. Matching the User Trace

Figure 9 presents a small portion of a trace of the SNIF-ACT simulation of user S1 working on the Antz task. Figure 9 corresponds to the portion of the task presented in Table 1. Each box represents a goal and each arrow represents the *firing* (match and execution) of a production rule in the SNIF-ACT simulation. The sequence of processing is depth-first, left-to-right in the diagram. Table 2 presents the correspondence of production rule firings in Figure 9 to the protocol in Table 1. Note that that many of the production firings cannot be associated with anything from the protocol or WebLogger. The SNIF-ACT model simulates aspects of internal user cognition do not have correspondence to overt behaviors.

Figure 5 shows how the user tracer controls the SNIF-ACT simulation model and matches the simulation behavior to the user trace data (each step is indicated by a circle in Figure 5):

⁵ Using a database provided to us by Hinrich Schuetze.

for determining navigation choices. The user tracing architecture enables the integration of multiple sources of user information to be tested against models that are intended to be scientific interpretations of detailed user behavior.

Currently, SNIF-ACT does not model the visual search of WWW pages. ACT-R has been used successfully to model visual search in other domains [34], so this is a natural next step for our SNIF-ACT simulation. So far, SNIF-ACT has not been used to model learnability and transfer, but these are phenomena that have been addressed with great success in previous versions of ACT [35]. Finally, we intend to further develop the version of SNIF-ACT that runs in a mode where it interacts directly with the WWW. It is this version that we expect to be used as a cognitive engineering model for making predictions about the effects of WWW designs.

8. ACKNOWLEDGEMENTS

This work is supported by an Office of Naval Research grant No. N00014-96-C-0097 to P. Pirolli and S. Card.

9. REFERENCES

1. Berners-Lee, T., *Information management: A proposal*. 1989, CERN: Geneva, Switzerland.
2. Pirolli, P. and S.K. Card. *Information foraging models of browsers for very large document spaces*. in *Advanced Visual Interfaces Workshop, AVI '98*. 1998. Aquila, Italy: ACM.
3. Pirolli, P. and S.K. Card, *Information foraging*. *Psychological Review*, 1999. **106**: p. 643-675.
4. Souza, R.K., *The best of retail site design*. 2000, Forrester Research Inc.: Cambridge, MA.
5. Manning, H., J.C. McCarthy, and R.K. Souza, *Why most web sites fail*. 1998, Forrester Research Inc.: Cambridge, MA.
6. Choo, C.W., B. Detlor, and D. Turnbull, *Web work: Information seeking and knowledge work on the World Wide Web*. 2000, Dordrecht: Kluwer Academic Publishers.
7. Brinck, T., D. Gergle, and S.D. Wood, *Usability for the Web: Designing Web sites that work*. 2001, San Francisco: Morgan Kaufman Publishers.
8. Krug, S., *Don't make me think*. 2000, New York: Circle.com.
9. Nielsen, J., *Designing Web usability*. 2000, Indianapolis, IN: New Riders.
10. Spool, J.M., et al., *Web site usability*. 1999, San Francisco, CA: Morgan Kaufman.
11. Garzotto, F., M. Matera, and P. Paolini. *Model-based heuristic evaluation of hypermedia usability*. in *Working Conference on Advanced Visual Interfaces*. 1998. L'Aquila, Italy: ACM Press.
12. User Interface Engineering, *Designing information-rich web sites*. 1999, Author: Cambridge, MA.
13. Spool, J. and W. Schroeder. *Testing Web sites: Five users is nowhere near enough*. in *Human Factors in Computing Systems, CHI '01*. 2001. Seattle, WA: ACM Press.
14. Newell, A. and S.K. Card, *The prospects for a psychological science in human-computer interactions*. *Human-Computer Interaction*, 1985. **2**: p. 251-267.
15. Card, S.K., T.P. Moran, and A. Newell, *The psychology of human-computer interaction*. 1983, Hillsdale, New Jersey: Lawrence Erlbaum Associates.
16. Paternò, F., V. Sabbatino, and C. Santoro. *Using information in task models to support design of interactive safety-critical applications*. in *Working Conference on Advanced Visual Interfaces, AVI 2000*. 2000. Palermo, Italy: ACM Press.
17. Pirolli, P., *Cognitive engineering models and cognitive architectures in human-computer interaction*, in *Handbook of applied cognition*, M.T.H. Chi, Editor. 1999, John Wiley & Sons: West Sussex, England. p. 441-477.
18. Lynch, G., S. Palmiter, and C. Tilt. *The Max model: A standard web site user model*. in *Human Factors and the Web*. 1999.
19. Pirolli, P., *A web site user model should at least predict something about users*. *internetworking*, 2000. **3**.
20. Byrne, M.D., et al. *The tangled web we wove: A taskonomy of WWW use*. in *Human Factors in Computing Systems, CHI '99*. 1999. Pittsburgh, PA: ACM Press.
21. Chi, E., et al. *Using information scent to model user needs and actions on the web*. in *Human Factors in Computing Systems, CHI 2001*. 2001. Seattle, WA.
22. Neisser, U., *Cognition and reality*. 1976, San Francisco, CA: Freeman.
23. Card, S., et al. *Information scent as a driver of Web Behavior Graphs: Results of a protocol analysis method for web usability*. in *Human Factors in Computing Systems*. 2001. Seattle, WA.
24. Morrison, J.B., P. Pirolli, and S.K. Card. *A taxonomic analysis of what World Wide Web activities significantly impact people's decisions and actions*. in *Conference on Human Factors in Computing Systems, CHI '01*. 2001. Seattle, WA: ACM Press.
25. Ericsson, K.A. and H.A. Simon, *Protocol Analysis: Verbal reports as data*. 1984, Cambridge, MA: MIT Press.
26. Reeder, R.W., P. Pirolli, and S.K. Card. *Web-Eye Mapper and WebLogger: Tools for analyzing eye tracking data collected in web-use studies*. in *Human Factors in Computing Systems, CHI 01*. 2001. Seattle, WA.
27. Hartson, H.R. and J.C. Castillo. *Remote evaluation for post-deployment usability improvement*. in *AVI '98, Proceedings of the Advanced Visual Interfaces*. 1998. L'Aquila, Ital.
28. Anderson, J.R. and C. Lebiere, *The atomic components of thought*. 2000, Mahwah, NJ: Lawrence Erlbaum Associates.
29. Anderson, J.R., *Cognitive psychology and its implications: Fifth edition*. 2000, New York: Worth.
30. Pirolli, P. *Computational models of information scent-following in a very large browsable text collection*. in *Conference on Human Factors in Computing Systems, CHI '97*. 1997. Atlanta, GA: Association for Computing Machinery.
31. Anderson, J.R. and R. Milson, *Human memory: An adaptive perspective*. *Psychological Review*, 1989. **96**: p. 703-719.
32. Anderson, J.R., *The adaptive character of thought*. 1990, Hillsdale, NJ: Lawrence Erlbaum Associates.
33. Harman, D. *Overview of the first text retrieval conference*. in *16th Annual International ACM/SIGIR Conference*. 1993. Pittsburgh, PA: ACM.
34. Anderson, J.R., M. Matessa, and C. Lebiere, *ACT-R: A theory of higher-level cognition and its relationship to visual attention*. *Human-Computer Interaction*, 1997. **12**: p. 439-462.
35. Singley, M.K. and J.R. Anderson, *Transfer of cognitive skill*. 1989, Cambridge, MA: Harvard University Press.

