www.ijecs.in
International Journal Of Engineering And Computer Science ISSN: 2319-7242
Volume 5 Issue 5 May 2016, Page No. 16369-16375

Enhancing Web Navigation Appropriateness by Correlating Actual and Predictable Practice

Ashwini R,

4th Sem, M.Tech ,CSE,BGSIT, Ashwinircs@gmail.com

Abstract—a new method to identify navigation- related Web usability problems based on correlating Actual and Predictable usage patterns. The actual usage patterns can be extracted from Web server logs routinely recorded for operational websites by first processing the log data to identify users, user sessions, and user task-oriented transactions, and then applying an usage mining algorithm to discover patterns among actual usage paths. The anticipated usage, including information about both the path and time required for user-oriented tasks, is captured by our ideal user interactive path models constructed by cognitive ex- perts based on their cognition of user behavior. The comparison is performed via the mechanism of test oracle for checking results and identifying user navigation difficulties. The deviation data produced from this comparison can help us discover usability issues and suggest corrective actions to improve usability. A software tool was developed to automate a significant part of the activities involved. With an experiment on a small service-oriented website, we identified usability problems, which were cross-validated by do- main experts, and quantified usability improvement by the higher task success rate and lower time and effort for given tasks after suggested corrections were implemented.

Index Terms—Cognitive user model, sessionization, software tool, test oracle, usability, usage pattern, Web server log.

I. INTRODUCTION

World Wide Web becomes prevalent today, building and ensuring easy-to-use Web systems is becoming core competency of business. Usability is defined as the effectiveness, efficiency, and satisfaction with which specific users can complete specific tasks in a particular environment Three basic Web design principles, i.e., structural firmness, functional convenience, and presentational delight, were identified to help improve users' online experience [42]. Structural firmness relates primarily to the characteristics that influence the website security and performance. Heuristic evaluation by experts and user-centered testing are typically used to identify usability issues and to ensure satisfactory usability . However, significant challenges exist, including 1) accuracy of problem identification due to false alarms common in expert evaluation,2) unrealistic evaluation of usability due to differences between the testing environment and the actual usage environment , and 3) increased cost due to the prolonged evolution and maintenance cycles typical for many Web applications . On the other hand, log data routinely kept at Web servers represent actual usage. Such data have been used for usage-based testing and quality assurance, and also for understanding user behavior and guiding user interface design .

We propose to extract actual user behavior from Web server logs, capture anticipated user behavior with the help of cognitive user models , and perform a comparison between the two. This deviation analysis would help us identify some navigation related usability problems. Correcting these problems would lead to better functional convenience as characterized by both better effectiveness (higher task completion rate) and efficiency (less time for given tasks). This new method would complement. The rest of this paper is organized as follows: Section II introduces the related work. Section III presents the basic ideas of our method and its architecture. Section IV describes how to extract actual usage patterns from Web server logs. Section V describes the construction of our ideal user interactive path (IUIP) models to capture anticipated Web usage. Section VI presents the com-

prison between actual usage patterns and corresponding IUIP models. Section VII describes a case study applying our method to a small service-oriented website. Section VIII validates our method by examining its applicability and effectiveness. Section IX discusses the limitations of our method. Conclusions and perspectives are discussed in Section X.

II. RELATED WORK

A. Logs, Web Usage and Usability

Two types of logs, i.e., server-side logs and client-side logs, are commonly used for Web usage and usability analysis. Server-side logs can be automatically generated by Web servers, with each entry corresponding to a user request. By analyzing these logs, Web workload was characterized and used to suggest performance enhancements for Internet Web servers. Because of the vastly uneven Web traffic, massive user population, and diverse usage environment, coverage-based testing is insufficient to ensure the quality of Web applications . Therefore, server-side logs have been used to construct Web usage models for usage-based Web testing , or to automatically generate test cases accordingly to improve test efficiency.

Server logs have also been used by organizations to learn about the usability of their products. For example, search queries can be extracted from server logs to discover user information needs for usability task analysis . There are many advantages to using server logs for usability studies. Logs can provide insight into real users performing actual tasks in natural working conditions versus in an artificial setting of a lab. Logs also represent the activities of many users over a long period of time versus the small sample of users in a short time span in typical lab testing . Data preparation techniques and algorithms can be used to process the raw Web server logs, and then mining can be performed to discover users' visitation patterns for further usability analysis . For example, organizations can mine server-side logs to predict users' behavior and context to satisfy users' need . Users' revisitiation patterns can be discovered by mining server logs to develop guidelines for browser history mechanism that can be used to reduce users' cognitive and physical effort .

Client-side logs can capture accurate comprehensive usage data for usability analysis, because they allow low-level user interaction events such as keystrokes and mouse movements to be recorded . For example, using these client-side data, the evaluator can accurately measure time spent on particular tasks or pages as well as study the use of "back" button and user click streams . Such data are often used with task- based approaches and models for usability analysis by com- paring discrepancies between the designer's anticipation and a user's actual behavior . However, the evaluator must program the UI, modify Web pages, or use an instrumented browser with plug-in tools or a special proxy server to collect such data. Because of privacy concerns, users generally do not want any instrument installed in their computers. Therefore, logging actual usage on the client side can best be used in lab-based experiments with explicit consent of the participants.

B. Cognitive User Models

In recent years, there is a growing need to incorporate insights from cognitive science about the mechanisms, strengths, and limits of human perception and cognition to understand the human factors involved in user interface design . For ex- ample, the various constraints on cognition (e.g., system complexity) and the mechanisms and patterns of strategy selection can help human factor engineers develop solutions and apply technologies that are better suited to human abilities. Software engineering techniques have also been applied to develop intelligent agents and cognitive models . On the one hand, higher level programming languages simplify the encoding of behavior by creating representations that map more directly to a theory of how behavior arises in humans. On the other hand, as these designs are adopted, adapted, and reused, they may become design patterns.

III. ARCHITECTURE OF A NEW METHOD

Our research is guided by three research questions:

1)*RQ1:*What usability problems are addressed?

2)*RQ2:* How to identify these problems?

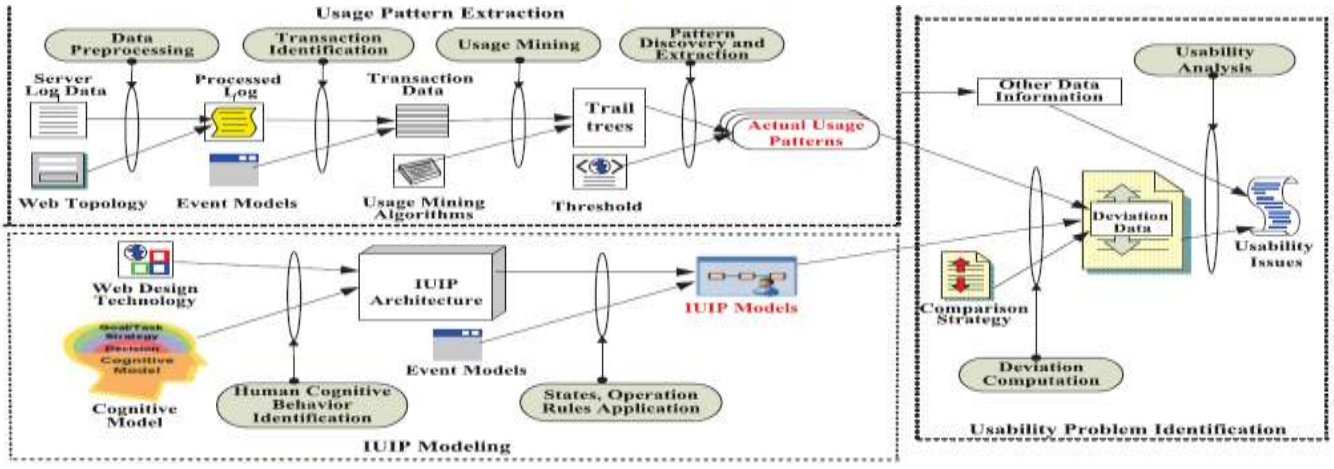
3)*RQ3:* How to validate our approach?

As described in the previous section, Web server logs have been used for usage-based Web testing and quality assurance. They have also been used for understanding user behavior and guiding user interface design. These works are extended in this study to focus on the functional convenience aspect of usability. In particular, we focus on identifying navigation related problems as characterized by an inability to complete certain tasks or excessive time to complete them (RQ1).

Usability engineers often use server logs to analyze users' behavior and understand how users perform specific tasks to improve their experience. On the other hand, many critics pointed out that server logs contain no information about the users' goals in visiting websites [37]. Usability engineers cannot generalize from server log data as they can from data collected by perform ing controlled experiments. However, these weaknesses can be alleviated by applying the cognitive user models we surveyed in the previous section. Such cognitive models can be constructed with our domain knowledge and empirical data to capture anticipated user behavior. They may also provide clues to users' intentions when they interact with Web systems.

We propose a new method to identify navigation related us- ability problems by comparing Web usage patterns extracted from server logs against anticipated usage represented in some cognitive user models (RQ2). Fig. 1 shows the architecture of our method. It includes three major modules: Usage Pattern Ex- traction, IUIP Modeling, and Usability Problem Identification. First, we extract actual navigation paths from server logs and discover patterns for some typical events. In parallel, we construct IUIP models for the same events. IUIP models are based on the cognition of user behavior and can represent anticipated paths for specific user-oriented tasks. The result checking em- ploys the mechanism of test oracle. An oracle is generally used to determine whether a test has passed or failed [6]. Here, we use IUIP models as the oracle to identify the usability issues related to the users' actual navigation paths by analyzing the deviations between the two. This method and its three major modules will be described in detail in Sections IV-VI.

We used the Furniture Giveaway (FG) 2009 website as the case study to illustrate our method and its application. Addition- ally, we also used the server log data of the FG 2010 website, the next version of FG 2009, to help us validate our method. All the usability problems in FG2009 identified by our method were fixed in FG2010. The functional convenience aspect of usability for this website is quantified by its task completion rate and time to complete given tasks. The ability to implement recommended changes and to track quantifiable usability im- provement over iterations is an important reason for us to use this website to evaluate the applicability and effectiveness of our method (RQ3).



DOI: 10.18535/ijecs/v5i5.10

Fig 1 Architecture of new method for identifying usability problems

The FG website was constructed by a charity organization to provide free furniture to new international students in Dal- las. Similar to e-commerce websites, it provided registration, selection, and removal of goods, submission of orders, and other services. It was partially designed and developed with the well- known templated page pattern . All outgoing Web pages go through a one-page template on their way to the client. Four templated pages were designed for the furniture catalog, furniture details, account information and selections. The FG website was implemented by using PHP, MySQL, AJAX and other dy- namic Web development techniques. It included 15 PHP scripts to process users' requests, 5 furniture catalog pages, about 200 furniture detail pages, and additional pages related to user in- formation, selection rules, registration and so on.

IV. USAGE PATTERN EXTRACTION

Web server logs are our data source. Each entry in a log contains the IP address of the originating host, the timestamp, the requested Web page, the referrer, the user agent and other data. Typically, the raw data need to be preprocessed and converted into user sessions and transactions to extract usage patterns.

A. Data Preparation and Preprocessing

The data preparation and preprocessing include the following domain-dependent tasks. 1) *Data cleaning:* This task is usually site-specific and involves removing extraneous references to style files, graphics, or sound files that may not be important for the purpose of our analysis. 2) *User identification:* The remaining entries are grouped by individual users. Because no user authentication and cookie information is available in most server logs, we used the combination of IP, user agent, and referrer fields to identify unique users.

Path #	Transaction	Frequency
1	a, b, e	10
2	b, d, b, c	4
3	b, c, e	9
4	a, b, e, g	5
5	a, d, b	12
6	b, d, b, e	8

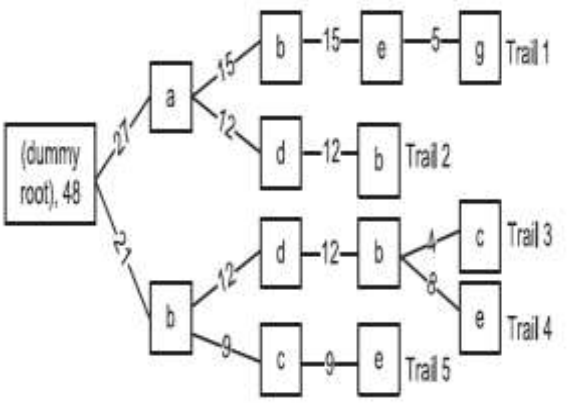


Fig. 2. Example of a trail tree (right) and associated transaction paths (left).

3) *User session identification:* The activity record of each user is segmented into sessions, with each representing a single visit to a site. Without additional authentication information from users and without the mechanisms such as embedded session IDs, one must rely on heuristics for session identification [3], [23]. For example, we set an

elapse time of 15 min between two successive page accesses as a threshold to partition a user activity record into different sessions.

4) *Path completion:* Client or proxy side caching can often

result in missing access references to some pages that have been cached. These missing references can often be heuristically inferred from the knowledge of site topology and referrer information, along with temporal information from server logs .

These tasks are time consuming and computationally intensive, but essential to the successful discovery of usage patterns. Therefore, we developed a tool to automate all these tasks except part of path completion. For path completion, the designers or developers first need to manually discover the rules of missing references based on site structure, referrer, and other heuristic information. Once the repeated patterns are identified, this work can be automatically carried out. Our tool can work with server logs of different Web applications by modifying the related pa- rameters in the configuration file. The processed log data are stored into a database for further use.

B.TransactionIdentificati

E-commerce data typically include various task-oriented events such as order, shipping, and shopping cart changes. In most cases, there is a need to divide individual data into corresponding groups called Web transactions. A transaction usually has a well-defined beginning and end associated with a specific task. For example, a transaction may start when a user places something in his shopping cart and ends when he has completed the purchase on the confirmation screen . A transaction differs from a user session in that the size of a transaction can range from a single page to all the visited pages in a user session.

In this research, we first construct event models, also called task models, for typical Web tasks. Event models can be built by Web designers or domain experts based on the use cases in the requirements for the Web application. Based on the event models, we identify the click operations (pages) from the click- stream of a user session as a transaction. For the FG website we constructed four event models for the following four typical tasks.

1) *Task 1:* Register as a new user.

2) *Task 2:* Select the first piece of furniture.

3) *Task 3:* Select the next piece of furniture.

4) *Task 4:* Change selection.

The example below shows the event model constructed for

Task2("FirstSelection"):

[post register.php] .*? [post process. php].

Here, "[ ]" indicates beginning and end pages; ".*?" indicates a minimal number of pages in a sequence between the two pages. For this task, we extracted a sequence of pages which started with the page "post register.php" and ended with the first appearance of the page "post process.php" for each session. Such a sequence of pages forms a transaction for a user.

We call the sequence a "path."

C. Trail Tree Construction

The transactions identified from each user session form a

collection of paths. Since multiple visitors may access the same pages in the same order, we use the trie data structure to merge the paths along common prefixes. A trie, or a prefix tree, is an ordered tree used to store an associative array where the keys are usually strings [35]. All the descendants of a node have a common prefix of the string associated with that node. The root is associated with the empty string.

We adapted the trie algorithm to construct a tree structure that also captures user visit frequencies, which is called a *trail tree* in our work. In a trail tree, a complete path from the root to a leaf node is called a *trail*. Each node corresponds to the occurrence of a specific page in a transaction. It is annotated with the number of users having reached the node across the same trail prefix. The leaf nodes of the trail tree are also annotated with the trail names.

An example trail tree is shown in Fig. 2. The transaction paths extracted from the Web server log are shown in the table to its left, together with path occurrence frequencies. Paths 1, 4, and 5 have the common first node *a*; therefore, they were merged together. For the second node of this subtree, Paths 1 and 4 both accessed Page *b*; therefore, the

two paths were combined at Node *b*. Finally, Paths 1 and 4 were merged into a single trail, Trail 1, although Path 1 terminates at Node *e*. By the same method, the other paths can be integrated into the trail tree. The number at each edge indicates the number of users reaching the next node across the same trail prefix.

Based on the aggregated trail tree, further mining can be per- formed for some "interesting" pattern discovery. Typically, good mining results require a close interaction of the human experts to specify the characteristics that make navigation patterns inter- esting. In our method, we focus on the paths which are used by a sufficient number of users to finish a specific task. The paths can be initially prioritized by their usage frequencies and selected by using a threshold specified by the experts. Application-domain knowledge and contextual information, such as criticality of specific tasks, user privileges, etc., can also be used to identified "interesting" patterns. For the FG 2009 website, we extracted
30 trails each for Tasks 1, 2, and 3, and 5 trails for Task
4.

IV. IDEAL USER INTERACTIVE PATH MODEL CONSTRUCTION

Our IUIP models are based on the cognitive models surveyed
in Section II, particularly the ACT-R model. Due to the com- plexity of ACT-R model development [9] and the low-level rule- based programming language it relies on [12], we constructed our own cognitive architecture and supporting tool based on the ideas from ACT-R.

In general, the user behavior patterns can be traced with a sequence of states and transitions [30], [32]. Our IUIP consists of a number of states and transitions. For a particular goal, a sequence of related operation rules can be specified for a series of transitions. Our IUIP model specifies both the path and the benchmark interactive time (no more than a maximum time) for some specific states (pages). The benchmark time can first be specified based on general rules for common types of Web pages. For example, human factors guidelines specify the upper bound for the response time to mitigate the risk that users will lose interest in a website [22]. Diagrammatic notation methods and tools are often used to support interaction modeling and task performance evaluation [11], [15], [33]. To facilitate IUIP model construction and reuse, we used C++ and XML to develop our IUIP modeling tool based on the open-source visual diagram software DIA. DIA allows users to draw customized diagrams, such as UML, data flow, and other diagrams. Existing shapes and lines in DIA form part of the graphic notations in our IUIP models. New ones can be easily added by writing simple XML files. The operations, operation rules, and computation rules can be embedded into the graphic notations with XML schema we defined to form our IUIP symbols. Currently, about 20 IUIP symbols have been created to represent typical Web interactions. IUIP symbols used in subsequent examples are explained at the bottom of Fig. 3. Cognitive experts can use our IUIP modeling tool to develop various IUIP models for different Web applications. .

Fig. 3 shows the IUIP model constructed by the cognitive experts for the event "First Selection" for the novice users of the FG website, together with the explanation for the symbols. Each state of the IUIP model is labeled, and the benchmark time is shown on the top. "-" means there is no interaction with users. Those pages are only used to post data to the server.

V. USABILITY PROBLEM IDENTIFICATION

The actual users' navigation trails we extracted from the aggregated trail tree are compared against corresponding IUIP models automatically. This comparison will yield a set of deviations between the two. We can identify some common problems of actual users' interaction with the Web application by focusing on deviations that occur frequently. Combined with expertise in product internal and contextual information, our results can also help identify the root causes of some usability problems existing in the Web design.

Based on logical choices made and time spent by users at each page, the calculation of deviations between actual users' usage patterns and IUIP can be divided into two parts:

1) Logical deviation calculation:

a) When the path choice anticipated by the IUIP model is available but not selected, a single deviation is counted.

b) Sum up all the above deviations over all the selected user transactions for each page.

2)Temporal deviation calculation:

a) When a user spends more time at a specific page than the benchmark specified for the corresponding state in the IUIP model, a single deviation is counted.

b) Sum up all the above deviations over all the selected user transactions for each page.

CONCLUSION

We have developed a new method for the identification and

Improvement of navigation-related Web usability problems by checking extracted usage patterns against cognitive user models. As demonstrated by our case study, our method can identify areas with usability issues to help improve the usability of Web systems. Once a website is operational, our method can be continuously applied and drive ongoing refinements. In contrast with traditional software products and systems, Web- based applications have shortened development cycles and pro- longed maintenance cycles. Our method can contribute significantly to continuous usability improvement over these prolonged maintenance cycles. The usability improvement in successive iterations can be quantified by the progressively better effectiveness.

REFERENCES

[1] A. Agarwal and M. Prabaker, "Building on the usability study: Two explorations on how to better understand an interface," in *Human-Computer Interaction. New Trends*, J. Jacko, Ed. New York, NY, USA: Springer, 2009,pp. 385-394.

[2] J. R. Anderson, D. Bothell, M. D. Byrne, S. Douglass, C. Lebiere, and Y. Qin, "An integrated theory of the mind," *Psychol. Rev.*, vol. 111, pp. 1036-1060, 2004.

[3] M. F. Arlitt and C. L. Williamson, "Internet Web servers: Workload characterization and performance implications," *IEEE/ACM Trans. Netw.*, vol. 5, no. 5, pp. 631-645, Oct. 1997.

[4] C. M. Barnum and S. Dragga, *Usability Testing and Research*. White Plains, NY, USA: Longman, Oct. 2001.

[5] B. Beizer, *Software Testing Technique*. Boston, MA, USA: Int. Thomson Comput. Press, 1990.

[6] J. L. Belden, R. Grayson, and J. Barnes, "Defining and testing EMR

usability: Principles and proposed methods of EMR usability evalu- ation and rating," *Healthcare Information and Management Systems Society*, Chicago, IL, USA, Tech. Rep., (2009). [Online]. Available:http://www.himss.org/ASP/ContentRedirector.asp?ContentID=71733

[7] M. C. Burton and J. B. Walther, "The value of Web log data in use-based

design and testing," *J. Comput.-Mediated Commun.*, vol. 6, no. 3, p. 0, 2001.

[8] M. D. Byrne, "ACT-R/PM and menu selection: Applying a cognitive architecture to HCI," *Int. J. Human-Comput. Stud.*, vol. 55, no. 1, pp. 41-84, 2001