How Do We Trace Requirements? An Initial Study of Analyst Behavior in Trace Validation Tasks

Wei-Keat Kong[1], Jane Huffman Hayes[1], Alex Dekhtyar[2], Jeff Holden[2]
[1]Department of Computer Science, University of Kentucky, Lexington, Kentucky
[2]Department of Computer Science, California Polytechnic State University, San Luis Obispo, California
wkkong1@uky.edu, hayes@cs.uky.edu, [dekhtyar, jholden]@calpoly.edu

ABSTRACT
Traceability recovery is a tedious, error-prone, person-power intensive task, even if aided by automated traceability tools. Human analysts must vet candidate traceability links retrieved by such tools and must often go looking for links that such tools fail to locate as they build a traceability matrix. This paper examines a research version of the traceability tool REquirements TRacing On target (RETRO) that logs analyst actions. We examine the user logs in order to understand how analysts work on traceability recovery tasks. Such information is a pre-requisite to understanding how to better design traceability tools to best utilize analyst time while developing a high quality final traceability matrix.

Categories and Subject Descriptors
D.2.1 [**Requirements/Specifications**]: Tools.

General Terms
Measurement, Experimentation, Human Factors.

Keywords
Traceability, Study of the Analyst, Logging, Effort.

1. INTRODUCTION
Traceability is defined as "the ability to describe and follow the life of a requirement, in both a forward and backward direction, e.g. from its origins, through its development and specification, to its subsequent deployment and use, and through periods of ongoing refinement and iteration in any of these phases" [8]. Traceability matrices (TMs) document traceability relationships between pairs of documents of the software engineering process. TMs are developed as part of a typical software engineering life cycle in order to support activities such as regression testing, change impact analysis, maintenance, and verification and validation. In safety- and mission-critical development efforts, the TM often plays an important role in the "certification" of the system; to ensure, for example, that all safety requirements have been implemented.

At present, traceability research faces an important dilemma. The traditional process of tracing throughout the software engineering life cycle, as well as the process of trace recovery used in verification and validation of software, is conducted by human analysts with minimal software support. This requires significant analyst effort, and makes the process tedious and potentially error-prone. Traceability matrices for mission- and safety-critical projects, however, must be certified by human analysts who bear the responsibility for validation of the constructed software. The immediate implication of these observations is two-fold. On one hand, automation of the tracing processes is highly desired, as it has the potential of drastically decreasing the time it takes to perform these tasks. On the other hand, the process must directly involve human analysts to certify TMs. This has lead to two directions in traceability research: studying the automating techniques or methods and studying the analyst.

Automation of the tracing process can be achieved by applying information retrieval (IR), data mining, and text mining techniques to the documents being traced. A number of studies in the past eight years examined the effectiveness of various automated techniques in tracing software engineering documents [1][12][2][14]. The study of human analysts working with automated tools, however, has only just commenced. Although anecdotal evidence [9][11] was presented five years ago indicating that human analysts working with tracing software do not necessarily improve the accuracy of candidate traces suggested by the software, systematic study of this topic started only recently. In 2010, Cuddeback et al. [5] undertook a study of 27 participants who were asked to validate an automatically generated candidate trace for a pair of textual documents (a trace is deemed *candidate* until it is vetted by a human) with the User Interface (UI) of RETRO [13]. The study confirmed what has been suspected for some time: analysts did not necessarily improve the accuracy of the candidate TM presented to them. While the study did show that the accuracy of the candidate TM had an effect on the accuracy of the TM submitted by the analysts, one of the most significant discoveries was that the best work was performed by analysts given the lowest accuracy candidate TMs . This left a number of questions to be investigated further. One of the important questions our research group has asked is "*how do the analysts use tracing software when they are working on trace validation tasks?"*

To answer such questions, we modified our research tool REquirements TRacing On-target (RETRO) to include an event logger that allowed us to collect information about the actual actions performed by analysts when undertaking trace recovery and trace validation tasks. We conducted a follow-up study involving 13 participants at two locations and obtained 13 logs detailing the analyst's work. In this paper, we present our initial analysis of the logs and our observations.

The paper is organized as follows. In Section 2, we provide some background on requirements tracing and the role of human

ICSE'11, May 21-28, 2011, Waikiki, Honolulu, HI, USA

analysts. Section 3 describes the experiment we conducted, the logging tool, and the information captured. Section 4 describes the results of our study and presents our analysis. Section 5 discusses the challenges and questions that the proposed research will address in the future.

2. REQUIREMENTS TRACING AND THE ROLE OF HUMAN ANALYSTS

Research has shown that automated traceability techniques retrieve traceability links faster than manual techniques [1][10]. Typically, the accuracy of a traceability matrix is measured using *recall:* the percent of true links that were retrieved, *precision:* the percent of retrieved links that were true, and the *f-measure*, the *harmonic mean* of recall and precision. Most studies over the past eight years indicated that automated techniques are capable of retrieving most of the true traceability links, and thus, have *high recall* [1][12][2][14]. However, the recall comes at the cost of also generating many false positives leading to relatively *low precision.*

An automated method for trace recovery takes as input a pair of textual documents from the software engineering process: a source document and a target document. Both documents are split into individual *elements.* For each element in the source document, the trace recovery task determines which elements in the target document are linked to it. Information retrieval- or text mining-based automated methods study the contents of the elements in the two documents and provide a *relevance estimate* for each pair of elements.

The key reason for studying automated methods for tracing is to replace menial analyst effort. In some settings where tracing occurs, e.g., post-deployment activities such as reverse engineering, fully automated tracing is a feasible alternative to the manual tracing procedures of today. However, trace recovery and trace validation tasks for mission- or safety-critical projects must include a human analyst who validates and updates, as necessary, any automatically generated traces. In such settings, automated tracing tools are still appropriate, as they can "cover more ground" much faster and present a decent *candidate trace* to an analyst in a matter of minutes. But *it is the accuracy of the final TM, delivered and certified by the analyst* that serves as the final judgment of success or failure of the tracing process.

Figure 1 depicts the results from the 2010 study [5] in the precision-recall space. Participants received candidate TMs with different accuracies. Each participant's performance is represented by a vector with the tail indicating the accuracy of the initial (assigned) TM and the head (arrow) indicating the accuracy of the submitted TM. The results of the study confirmed initial observations: human analysts that get more accurate candidate TMs do not always produce more accurate final TMs. In fact, one of the most important observations from the study was that the analysts who were provided the least accurate candidate TMs were the only ones who *consistently and significantly improved the accuracy of the TM* while performing the trace validation task.

In the absence of a human analyst, *recall* and *precision* provide a clear way of determining which automated method is better: methods that lead to higher accuracy for automatically generated TMs. However, the study described above makes it clear that this may not be the right way of determining the best automated tracing method to be used to generate candidate TMs for analyst validation. This creates a real challenge for the traceability community: *without understanding how analysts work with automated tracing software, it is impossible to successfully automate the tracing process.*

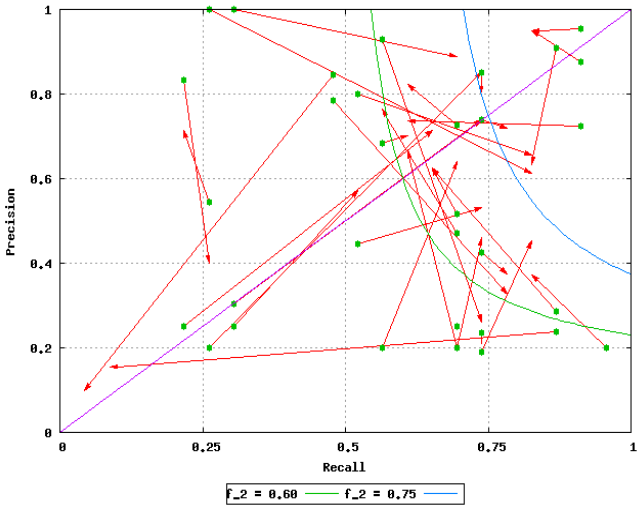

Figure 1. Cuddeback et al. study results

3. EXPERIMENT DESIGN

To better understand the work of the analysts with tracing software, we conducted an experimental study that is outlined below.

3.1 Overview

Our study was conducted in two upper-division Software Engineering classes: one at the University of Kentucky and one at Cal Poly. The participants of the study were senior and graduate students majoring in Computer Science and Software Engineering. Prior to the study, a pre-survey was given to gauge each participant's level of software engineering and tracing expertise, as well as their confidence in their ability to perform tracing. Participants were given access to a special-purpose requirements tracing tool and a small training example in order to familiarize themselves with the tool. Tichy [15] observes that appropriately trained students are adequate for determining trends. In the study, the participants used a version of the tool enhanced with a logging mechanism to validate a candidate TM, modifying the TM as needed: removing false links or discovering true links outside of the candidate TM. Participants submitted the final TM and the user activity log at the end of the study. A post-study survey asked questions about the participants' experience with the tracing task, the tracing software, and their self-assessment on how well-prepared they were.

3.2 Dataset

The dataset for the study was constructed from the documentation for a BlueJ plugin Java code formatter named ChangeStyle. It was chosen because (a) the domain is easily understood by study participants, and (b) its size makes trace validation tasks achievable in about one hour. The dataset contains 32 requirements (source elements) and 17 system tests (target elements). The research team generated and validated the *golden standard* TM containing 23 true links [3].

3.3 Software Tool

For the experiment, we added analyst action logging functionality to a somewhat more user-friendly version of RETRO called RETRO.NET [6]. We used this tool to deliver pre-computed candidate TMs assigned to each participant.

Figure 2 shows the UI of the tool. The participant starts the task by logging in to the tool. Next, they are presented with the assigned candidate TM to trace. On the left side of the UI, the list of source elements and the text of the current source element are displayed. On the right side of the UI, the list of target elements and their text is shown. The participant evaluates each candidate link and renders a *Link/Not a Lin*k decision (initially all candidate links are labeled *Default*). The participant can also mark source elements as Satisfied/Partially Satisfied/Not Satisfied by target elements. The UI also allows a participant to perform simple keyword searches in both source and target elements, view all links, as well as perform a few other actions that were less relevant to the direct task of trace validation.

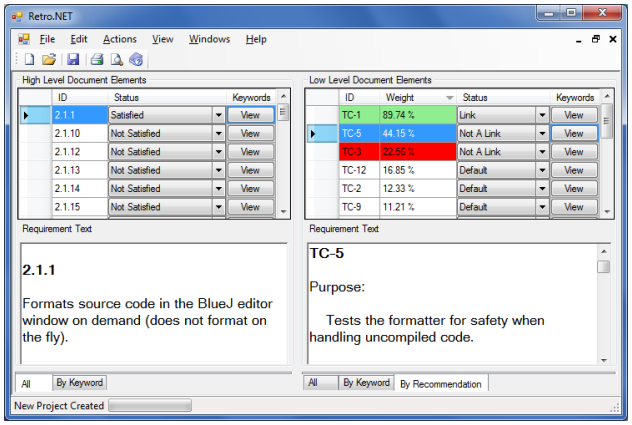

Figure 2. RETRO.NET UI

3.4 The Logger

To understand the participant decision-making process, we could have asked them to record what they were thinking as they performed the task. In fact, Cuddeback et al. [5] collected a simple handwritten task log that allowed for some crude estimate of the participant effort. However, a more detailed manually generated task log would invariably affect the performance of the task, forcing the participant to switch between the tracing task and documenting their decision-making process. Besides causing them to switch mental activities, this would also increase the amount of time required to perform the tracing task.

An alternate way of getting this information is for the software tool to log participant actions during the task; this does not put any additional burden on the participant. The possible downside of this approach is that the research team analyzing the logs may misinterpret participant intent. Log analysis, however, can provide key insights into participant behavior that would otherwise be difficult to obtain without affecting the outcome of the task[1].

[1] In any such experiment there is the possibility of impacted/biased results due to the Hawthorne effect. This problem exists regardless of whether we log actions using the tool or ask the participants to document their mental process.

The action logger tracks the following actions in a log file along with a time stamp for each action:

1. User selects a source/target element in the TM.
2. User views recommended links, views all links, or performs a keyword search (using the tabs at the bottom of the RETRO.NET UI window).
3. User marks the observed source/target element pair as a (true) link or not a link.
4. User marks a source element as satisfied, partially satisfied, or not satisfied by target elements.

Figure 3 shows an example of actions performed during a particular task. We see from the log on row 1 that source element 2.0.0 was selected by the participant and target element TC-11 (row 2) was displayed at 12:52:03. The participant performed a keyword search for 'documentation' seven seconds later and TC-14 was displayed. Ten seconds later, the participant confirmed TC-14 as a link to 2.0.0 (row 5). Logs are stored by the tool in comma-separated value format. Log analysis includes running automated scripts to parse and process actions of interest for further analysis.

12:52:03	2.0.0	Selected
12:52:03	TC-11	Selected
12:52:10	LowLevelID	Keyword search: documentation
12:52:10	TC-14	Selected
12:52:20	TC-14	Marked Link
12:52:28	1.0.4	Selected
12:53:04	TC-11	Selected
12:53:17	LowLevelID	By Recommendation selected.
12:53:45	TC-11	Selected
12:53:52	TC-11	Marked Link
12:54:01	LowLevelID	All links selected.
12:54:02	TC-2	Selected
12:54:08	TC-13	Selected
12:55:13	TC-13	Marked Not A Link
12:55:15	TC-8	Selected
12:55:16	TC-12	Selected
12:55:17	TC-19	Selected
12:55:19	TC-5	Selected
12:55:37	TC-5	Marked Link

Figure 3. Sample log output from RETRO.NET

4. RESULTS AND DISCUSSION

The experiment collected thirteen responses: eight responses from one of the universities and five responses from the other.

Table 1 (at the end of this paper) summarizes the work of the study participants. It shows the accuracy of the candidate TMs presented to each participant, the accuracy of the final TM submitted by the participants, and the change in the TM accuracy. The accuracy is reported as recall, precision, and the *f2-measure*, the variant of the *f-measure* that values recall higher than precision[2]. For example,

[2] For an extensive discussion of why *f2-measure* is used instead of *f-measure*, we refer the reader to earlier work [12]. In short, it

UserA was presented with a TM that had 7 true links out of 35 candidate links (30.4% recall, 20% precision, and 27.6% F2). At the end of the task, UserA submitted a TM that contained 15 true links out of 28 total links (65.2% recall, 53.6% precision, and 62.5% F2), significantly improving the quality of the TM (difference of 34.8% recall, 33.6% precision, and 34.9% F2). The information in this table only tells us the beginning and the end of the user's story. As with Figure 1, which showed the overall change in the TM accuracy for participants in the earlier study [5], we graph the Table 1 data in Figure 4. To better understand the "middle" of the user story for the 13 participants, we proceed as follows: two user logs are examined in detail; all logs are analyzed and graphed for trends; and observations are made.

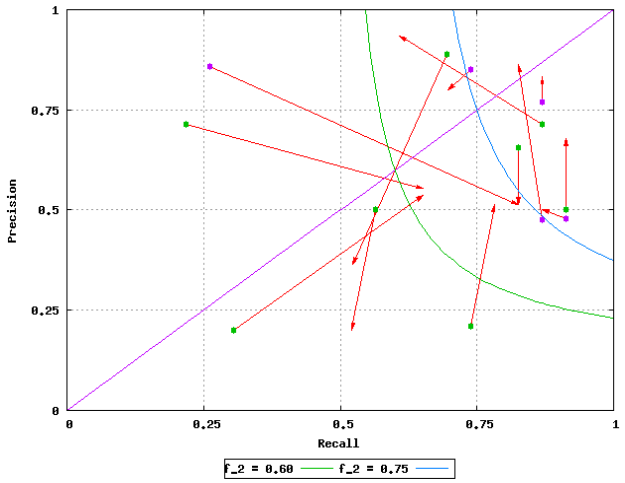

Figure 4. Performance of the 13 study participants plotted in the precision-recall space

4.1 Analyst Logs

Delving into the log of an analyst's actions reveals a wealth of information about what happened during the task. For example, did the participant read all of the source elements before beginning to mark links for any source elements? How much time was spent searching for links not in the candidate TM? We illustrate what can be gleaned from individual logs by examining two sample user logs.

UserM spent nearly four minutes on source element 1.9.5 early on in the task, then took about 30 seconds to skim through the remaining links before starting back at the top and marking links for about ten minutes. Then, about four minutes were spent reviewing the TM. The last thirteen minutes of the task were spent performing keyword searches, which resulted in one dropped true link being added back into the TM.

UserF had difficulty with the first few source elements, spending six minutes on them before continuing on, then going back and spending another two minutes to mark them. From there, marking the rest of the links took about eight minutes. Then two minutes were spent reviewing links.

---

is known in the field of traceability that finding an omitted link is more difficult/time-consuming than rejecting a false positive (in fact, we actually directly observe similar behavior in this study). Hence, in determining the accuracy of a TM using a single measure, we value recall more.

From these two logs, we start to see a pattern of difficulty with certain elements early on in the task, especially with source element 1.9.5.

4.2 Log Analysis

The examples above suggest that looking at the logs side-by-side may reveal some common trends. Log analysis revealed that participants spent an average of 32.5 minutes on the task (min. 18 minutes, max. 48 minutes, std. dev. 9.4 minutes). Participants spent an average of 5.6 minutes to find and make a decision on the first true link in the TM (min. 2 minutes, max. 10 minutes, std. dev. 2.3 minutes). The discovery that participants took a significant amount of time to start marking links leads us to look further into the logs as to possible causes of such behavior.

Log analysis also identified various strategies used by participants during the task, i.e., review recommended links most of the time; review all links most of the time; review recommended links first then review all links; review recommended links first then search for keywords; and alternate between recommended links, keyword search, and all links. From log analysis and the final TM metrics, it appears that participants starting with high recall TMs tend to end up with slightly lower recall but increased precision, and participants starting with low recall TMs tend to end up with higher recall but lower precision TMs. Almost all participants confirmed TMs with at least 65% recall and at least 50% precision, which was acceptable for recall, and excellent for precision based on a classification of results by Hayes et al. [12].

In the user logs, we looked for factors that influence when a participant decides to search outside the recommended list for additional links (and whether these searches are fruitful). We noticed that certain links were dropped by most participants, leading us to analyze these links to identify factors that prevent participants from correctly identifying them. This analysis is still incomplete but will provide insight into the design of future traceability tools as well as provide advice for assisting software engineers to write more easily traceable documents.

4.3 Log Depiction

With the above insights in mind, we developed a number of ways to examine the user logs. We depicted the thirteen logs and observed trends. For example, we noticed that the thirteen participants exhibited four different patterns of behavior over the life-time of the task: some found links early, some found links later, some found links early but then began to make significant mistakes, and some found correct links and made mistakes throughout the entire task.

Figures 5, 6, and 7 depict the progress of the thirteen participants throughout the task using two sets of graphs. All participants start with an empty ***confirmed TM***, hence, the starting accuracy is 0% recall and 0% precision and 0% F2-measure. Precision, recall, and F2-measure of the ***confirmed TM*** changes as correct and incorrect links are *confirmed* by each participant. One set of graphs plots the change in precision vs. recall. A directional arrow (not drawn in the graphs) from the (red) circle to the last precision/recall point of the task corresponds to the graph shown in Figure 4. The other set of graphs plots the F2-measure of the ***confirmed* TM** over elapsed task time. F2-measure increases as participants make correct decisions (either confirm a true candidate link or discover an omitted true link) and decreases with each incorrect decision (confirmation or inclusion of a false positive). A rejected true link is also an incorrect action, but it does not alter the F2 value.

Confirmed true links are marked as (green) circles, confirmed false positive links are marked as (red) Xs, and rejected true links are marked as (red) triangles. The graphs also contain a horizontal line signifying the F2-measure of the starting TM.

Figure 5 plots the decisions made by the six participants who ***started slowly, sometimes with a number of incorrect decisions, but after a certain point stopped making mistakes.*** The observation that we made from reading two user logs in an earlier section is seen here: participants in this group have difficulty identifying correct links until after they have spent at least 20 minutes on the task. Log analysis shows that half of the participants in this group were reviewing *all links* during the earlier part of the task, which could contribute to the delay in reaching the true links in the rest of the candidate TM.

Figure 6 shows the progress of a group of four participants ***who were able to locate correct links earlier in the task and made very few mistakes throughout the task***. Log analysis reveals that while most of these participants still had a 'delay' in marking links; they were able to get past the hurdle quickly and then were able to go through links at a faster pace (compared to the participants shown in Figure 5). They made a few occasional mistakes: two participants made some mistakes at the very end, while the other two made a few individual mistakes in the first half of the task.

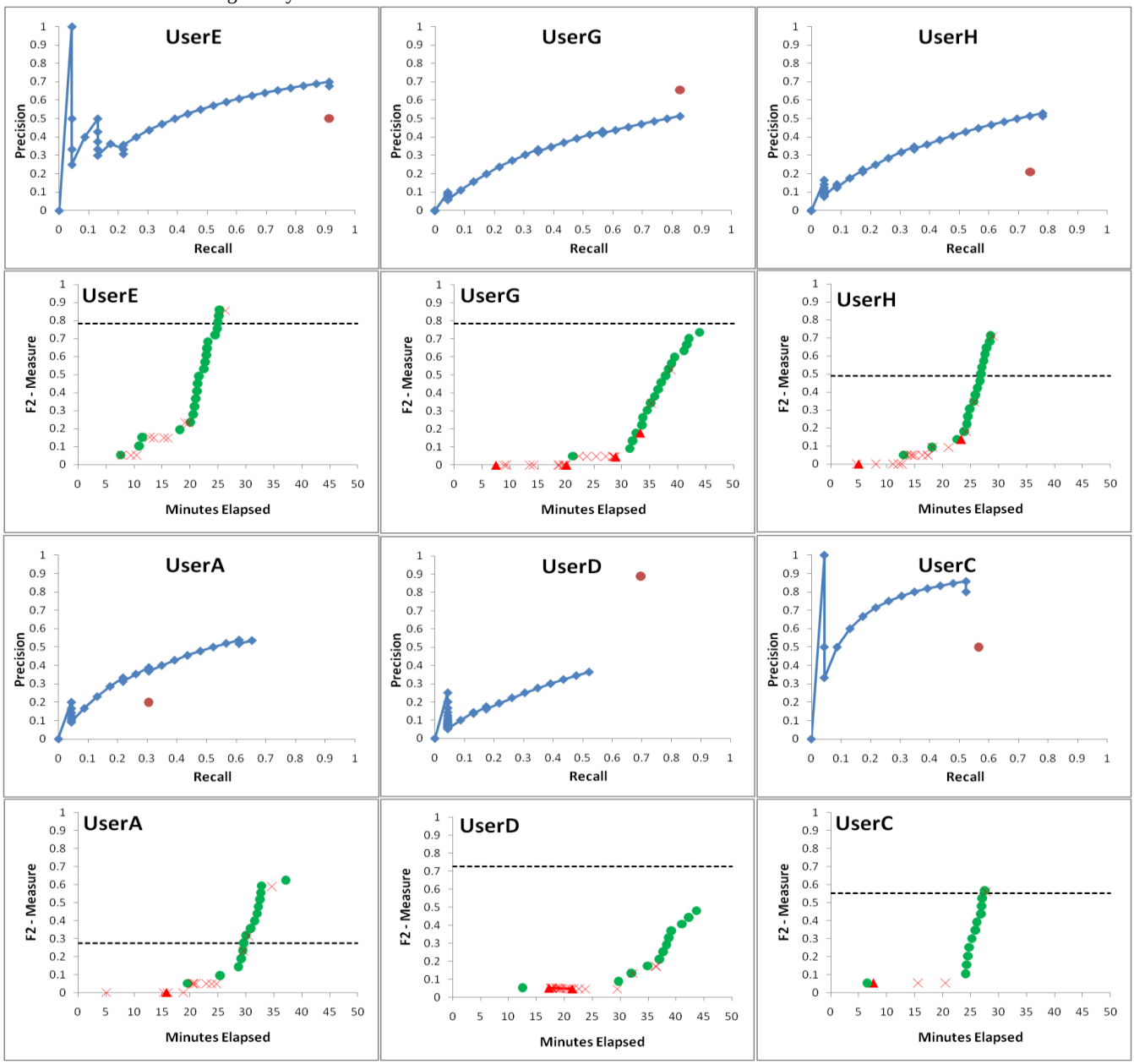

Figure 5. Group of users finding links later

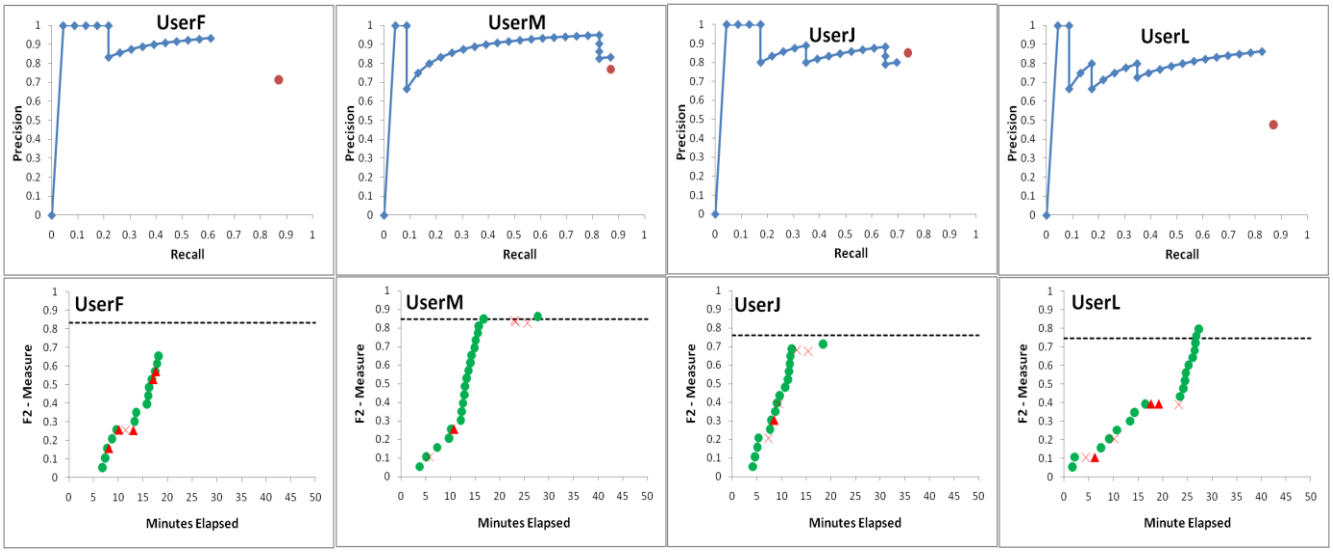

Figure 6. Group of users finding links earlier

Figure 7 presents the work of two participants who showed a period of "tiredness" during which they made many incorrect decisions in a row: at the very end of the task for one participant, in the middle of the task for the other participant. Log analysis reveals that one participant, UserB, had finished going through the recommended links in the TM and was adding additional links outside of the recommended list. About 40% of the false links added by the other participant came from links to a single source element, 1.9.5. The other participant, UserK, actually showed behavior similar to that of UserM and UserJ (Figure 6), but with a more pronounced bout of final mistakes.

Figure 8 shows the work of UserI who interspersed correct decisions with occasional mistakes evenly throughout the task. The recommended TM for this participant was very small, which resulted in the participant searching outside the recommended TM almost the whole time. The graphs capture the change in the nature of UserI's activity after UserI "ran out" of candidate links to confirm.

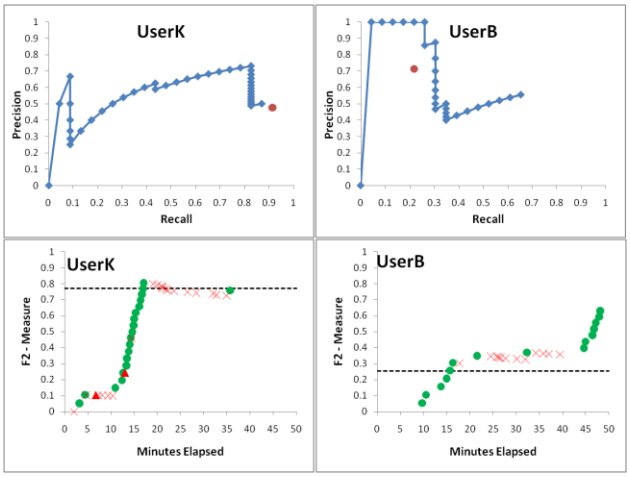

Figure 7. Participants making mistakes at certain points of the task

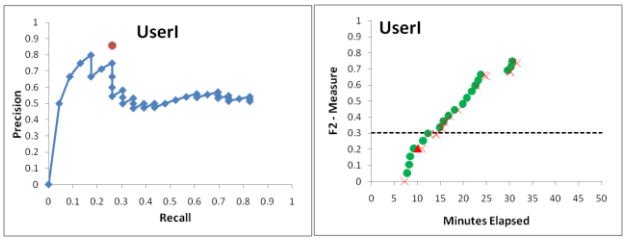

Figure 8. Participant making mistakes evenly throughout

It is clear from Figures 5 through 8 that all participants had an "upward hill" climb during which they were able to find correct links. Log analysis reveals that the last 18 or so links from the bottom of the recommended list were marked quicker due to presumably a much clearer link between the source element and the target element. The variability of the "climb" seems to be in how quickly the participant started to climb, and whether or not the participant made mistakes after the steep uphill climb (the two participants shown in Figure 7). Further analysis of the individual links involved needs to be undertaken to see if the links that contributed to the initial delay in making good decisions are the same ones that contributed to the "drop off" of good work in some user sessions.

In Figure 9, we present an additional depiction of the user log based on the effort spent on each true link. The real source and target element names in the figure have been altered since this dataset is still used in current studies. An automated script parses the log for actions related to true links and sums the time spent on each link. Each row of the table represents one of the 23 true links in the TM. Link 1.9.8->TC16 (Row 8), for example, was viewed by eight out of the 13 participants (black squares indicate that the participant did not even view the link). UserE spent 0.2 minutes on the true link before confirming it as a true link. On the other hand, UserF spent one minute on the same link and ended up rejecting the true link. UserG initially rejected the true link but changed their decision right away, which was most probably due to selecting the wrong option in the UI. Overall, around 25% of the decisions required the participant to spend at least 30 seconds or more, of which about 75% of the decisions were correct. There were a number of participants who wavered in their decision on certain links in the TM, but there was no particular link that caused this

behavior (this can be seen from the ^ and – links in the table). In most cases, participants spent additional time on these source elements, trying to decide whether the element pair was a link or not, perhaps due to some ambiguity in the description of the elements. Note that this reinforces a similar observation made by Egyed et al. [7] in a manual tracing experiment. Focusing on these "ambiguous" links will allow us to address such issues in future traceability research.

Link\User	A	B	C	D	E	F	G	H	I	J	K	L	M
1.9.5->TC-11	8.8-	7.7	1.9	7.2!	5.6^	4.6	4.2!	0.6!	0.0*	0.0*	1.4	0.1!	4.6
2.0.4->TC-3		0.1		0.2	0.4	3.9	0.1!		0.1	0.0*	1.6		
2.0.2->TC-16	0.1	0.0*		0.1!	0.7	1.0!	0.3	0.2	0.0*	0.0*	1.3!	0.3!	0.8
1.9.9->TC-15	0.6		0.8	0.2	0.2	0.6	0.2!	0.2	0.2	4.2	0.2	0.7	1.0
1.9.4->TC-15		0.4	1.0!	0.6^	0.4	0.7	0.0	0.0	0.1	0.7	0.6	0.2	0.8
1.9.2->TC-15	0.4	0.6	0.9	0.0*	0.3	1.1	0.0	0.3!	0.3	0.2	0.3	0.4	0.6
1.9.6->TC-11		0.3		0.6	0.2	0.5!	0.3	0.5	0.1	0.0*	0.9!	0.6!	0.7!
1.9.8->TC-16					0.2	1.0!	0.1^	0.1	0.0		0.2	0.7	0.5
2.0.0->TC-14		0.2	0.2	0.3	0.3	0.2	0.1	0.2	0.1	0.2	0.2	0.2	0.3
1.0.4->TC-1	0.4	0.5	0.3	0.1*	0.1	0.3	0.1	0.4	0.3	0.4	0.4	0.2	0.3
1.6.0->TC-2	0.2		0.2		0.2	0.2*	0.2	0.3	0.1	0.3	0.3	0.3	0.3
1.6.5->TC-13	0.1	0.9	0.5	0.3	0.9	1.2	0.2	0.1	0.1	1.1	0.1	0.3	0.2
1.7.5->TC-8	0.1	0.2	0.1	0.6	0.1	0.4	0.1	0.2	0.1	0.2	0.2	0.2	0.1
1.8.5->TC-12		0.4		0.4	0.2	0.2	0.1	0.3	0.0*	0.3^	0.2	0.2	0.4
1.5.5->TC-19	0.1	0.1		0.1	0.2	0.1*	0.0	0.1	0.1	0.3	0.3	0.5	0.4
1.8.7->TC-5	0.1	0.0	0.3	0.1	0.2	0.6	0.0	0.2	0.1	0.1	0.4	0.4	0.3
1.8.8->TC-5	0.2	0.0*	0.0*	0.0*	0.1^	0.3!	0.0	0.2	0.2	0.3	0.5	0.1	0.7
1.0.5->TC-4	0.4	0.0*	0.2	0.3*	0.2	0.3	0.1	0.2	0.1	0.2	0.2	0.3	0.2
1.1.0->TC-6	0.2	3.0	0.1	0.3	0.1	0.2!	0.1	0.2	1.1!	0.6-	0.3	0.3	0.6^
2.2.2->TC-6	0.1	0.6-^	0.1	0.3*	0.2	0.2	0.2	0.6	0.8	0.2	0.3	0.2	0.2
2.2.5->TC-7	0.1^	1.1	0.4	0.5	0.1	0.3	0.1	0.1	0.4	0.3	0.3	0.2	0.2
1.8.0->TC-9	0.1*						0.3!		0.1		0.0*	0.9	
1.7.0->TC-10				0.8			0.1	0.1*	0.1	0.1	0.1	0.3	0.3

! Dropped ^ Dropped then added - Added then dropped * Viewed but no decision

Figure 9. Participant effort spent on each true link

4.4 Observations

Based on the logs and the depiction of the logs, a number of observations can be made:

The quality of the final TM is influenced by the quality of the initial TM. In addition, analysts given low quality initial TMs tend to make the best decisions as they develop a final TM, validating the observations made in the Cuddeback et al. study [5]. We observe that certain links are very troublesome for the analysts while others tend to be very intuitive and easy to identify. When an analyst spends very little time on a link they tend to make the correct decision. On difficult links, where the analyst struggles to make a decision, they frequently commit to the incorrect decision. One of the key observations that we discovered using the log depictions was that all analysts eventually settle into a pattern where they make multiple correct decisions in a row. In several of the cases, this behavior lasts a short time, leading to a second "incorrect link" trend. This "incorrect streak" often occurred when their confirmed TM recall approached the candidate TM recall. This seems to occur when analysts did not search outside their candidate TM to locate missing links; instead they focused on rejecting incorrect links. In most cases these decisions were confirming links rather than rejecting incorrect links or searching for a missing link. This adds additional support to the notion that validating a link is a simpler task than discovering a new link [12].

An additional key observation was that analysts tend to cause more errors after the nature of the task changed. This can be seen when an analyst was presented with an initial TM with low recall and high precision: such candidate TMs are small. In our study, only two participants, UserB and UserI, were assigned such TMs. Both participants quickly ran out of candidate links, appeared to conclude that more links needed to be discovered and, thus, were forced to search for omitted links. Both participants confirmed many false links past the point where the nature of their task changed. While anecdotal at this point, if this is confirmed in later studies, we can utilize this information as an essential requirement for future tracing tools: the tool should not produce results with too few links for the analyst to validate, because the *switch* from link confirmation to link discovery causes errors of judgment to be introduced.

A final key observation is that, for the most part, analysts were able to use RETRO.NET effectively with minimal training and guidance. The analysts tended to use the tool as intended, explored a range of functionality available to them in the tool, and were able to successfully perform the tracing task.

5. FUTURE DIRECTIONS

Trace validation is a required undertaking for any mission- or safety-critical software project, requiring humans to make the final decision on links in the TM. As the final TM is the determiner of a successful or unsuccessful trace validation task, traceability research must consider how human analysts work in concert with output from an automated tracing tool. The discovery that humans given the lowest accuracy TMs made the best decisions when validating those TMs was a profound one, surely causing all traceability researchers to question our long quest for an automated tool that provides perfectly accurate candidate TMs.

The problem of automating parts of the tracing process continues to be a topic for researchers to address. These new discoveries, however, strongly suggest that the original goal to produce automated candidate TMs of highest accuracy might not be the right one. With the research described in this paper, we are finally able to go beyond simple accuracy numbers and actually look inside the tracing process to see the causes for both positive and negative analyst behavior. This is not only the first step towards understanding the role of the analyst, but also the first step toward building the proper automated tools for this process.

That being said, this study represents the first step in what should be a very long journey toward understanding user behavior. Specifically, we study the behavior of analysts to obtain key insights as to how to best focus their efforts and reinforce their confidence in the tool's ability to locate true links. We are looking to conduct further experiments and analysis to determine what is keeping many users from reaching the "uphill climb," good behavior pattern, quickly. One idea we are looking to pursue is characterizing "hard links," by doing so we could add a training session to begin the validation task. Based on this information, we could add a training session to precede the user's validation task. By doing this, we can attempt to "teach" the user how to decide on difficult link pairs in an efficient manner. We could design the tool in such a way that the analyst is required to pass the "validation task" prior to proceeding to the real trace. This idea is similar to our "educating the user" [4] concept proposed in a different paper submitted for publication.

Table 1. Initial and Final TMs for each participant

User	Begin true links	Begin total links	Recall	Precision	F2	Final true links	Final total links	Recall	Precision	F2	Diff Recall	Diff Precision	Diff F2
UserA	7	35	30.4%	20.0%	27.6%	15	28	65.2%	53.6%	62.5%	34.8%	33.6%	34.9%
UserB	5	7	21.7%	71.4%	25.3%	15	27	65.2%	55.6%	63.0%	43.5%	-15.9%	37.8%
UserC	13	26	56.5%	50.0%	55.1%	12	15	52.2%	80.0%	56.1%	-4.3%	30.0%	1.0%
UserD	16	18	69.6%	88.9%	72.7%	12	33	52.2%	36.4%	48.0%	-17.4%	-52.5%	-24.7%
UserE	21	42	91.3%	50.0%	78.4%	21	31	91.3%	67.7%	85.4%	0.0%	17.7%	7.0%
UserF	20	28	87.0%	71.4%	83.3%	14	15	60.9%	93.3%	65.4%	-26.1%	21.9%	-17.9%
UserG	19	29	82.6%	65.5%	78.5%	19	37	82.6%	51.4%	73.6%	0.0%	-14.2%	-4.9%
UserH	17	81	73.9%	21.0%	49.1%	18	35	78.3%	51.4%	70.9%	4.3%	30.4%	21.7%
UserI	6	7	26.1%	85.7%	30.3%	19	37	82.6%	51.4%	73.6%	56.5%	-34.4%	43.3%
UserJ	17	20	73.9%	85.0%	75.9%	16	20	69.6%	80.0%	71.4%	-4.3%	-5.0%	-4.5%
UserK	21	44	91.3%	47.7%	77.2%	20	40	87.0%	50.0%	75.8%	-4.3%	2.3%	-1.4%
UserL	20	42	87.0%	47.6%	74.6%	19	22	82.6%	86.4%	83.3%	-4.3%	38.7%	8.7%
UserM	20	26	87.0%	76.9%	84.7%	20	24	87.0%	83.3%	86.2%	0.0%	6.4%	1.5%

Additional log analysis also needs to be undertaken with the data from the 13 participants, as well as future studies. We are trying to characterize links and determine what makes them simple or difficult for analysts to validate. By performing this analysis, we look to find key "traits" that can make the tracing task easier and use this to design a tool that caters to this user behavior.

6. ACKNOWLEDGMENTS

This work is funded in part by the National Science Foundation under NSF grant CCF-0811140 and in part by a grant from Lockheed Martin Corporation. We thank John Dalbey and David Cuddeback for the ChangeStyle dataset. We thank Clark Savage Turner for allowing the experiments in his course; we thank all the Cal Poly and UK participants. We thank Jody Larsen for RETRO.NET.

7. REFERENCES

[1] Antoniol, G., Canfora, G., Casazza, G., De Lucia, A. and Merlo, E. 2002. Recovering traceability links between code and documentation. *Software Engineering, IEEE Transactions on*. 28, 10 (2002), 970-983.

[2] Cleland-Huang, J., Chang, C.K., Sethi, G., Javvaji, K., Hu, H. and Xia, J. 2002. Automating Speculative Queries through Event-Based Requirements Traceability. *Requirements Engineering, IEEE International Conference on* (Los Alamitos, CA, USA, 2002), 289.

[3] Cuddeback, D. *Automated Requirements Traceability: the Study of Human Analysts*. Ph.D. Thesis, Cal Poly, CA., 2010.

[4] Cuddeback, D., Dekhtyar, A., Hayes, J.H., Holden, J. and Kong, W. New Ideas and Emerging Results Track: Towards Overcoming Human Analyst Fallibility in the Requirements Tracing Process. *Accepted, ICSE 2011 NIER track*.

[5] Cuddeback, D., Dekhtyar, A. and Hayes, J. 2010. Automated Requirements Traceability: The Study of Human Analysts. *Proceedings of the 2010 18th IEEE International Requirements Engineering Conference* (Washington, DC, USA, 2010), 231–240.

[6] Dekhtyar, A., Hayes, J.H. and Larsen, J. 2007. Make the Most of Your Time: How Should the Analyst Work with Automated Traceability Tools? *Proceedings of the Third International Workshop on Predictor Models in Software Engineering* (Washington, DC, USA, 2007), 4–.

[7] Egyed, A., Graf, F. and Grunbacher, P. 2010. Effort and Quality of Recovering Requirements-to-Code Traces: Two Exploratory Experiments. *Proceedings of the 2010 18th IEEE International Requirements Engineering Conference* (Washington, DC, USA, 2010), 221–230.

[8] Gotel, O. and Finkelstein, C. 1994. An analysis of the requirements traceability problem. (1994), 94-101.

[9] Hayes, J.H. and Dekhtyar, A. 2005. Humans in the traceability loop: can't live with 'em, can't live without 'em. *Proceedings of the 3rd international workshop on Traceability in emerging forms of software engineering* (New York, NY, USA, 2005), 20–23.

[10] Hayes, J.H., Dekhtyar, A. and Osborne, J. 2003. Improving Requirements Tracing via Information Retrieval. *Proceedings of the 11th IEEE International Conference on Requirements Engineering* (Washington, DC, USA, 2003), 138–.

[11] Hayes, J.H., Dekhtyar, A. and Sundaram, S. 2005. Text mining for software engineering: how analyst feedback impacts final results. *Proceedings of the 2005 international workshop on Mining software repositories* (New York, NY, USA, 2005), 1–5.

[12] Hayes, J.H., Dekhtyar, A. and Sundaram, S.K. 2006. Advancing Candidate Link Generation for Requirements Tracing: The Study of Methods. *IEEE Trans. Softw. Eng.* 32, 1 (2006), 4-19.

[13] Hayes, J.H., Dekhtyar, A., Sundaram, S.K., Holbrook, E.A., Vadlamudi, S. and April, A. 2007. REquirements TRacing On target (RETRO): improving software maintenance through traceability recovery. *ISSE*. 3, 3 (2007), 193-202.

[14] Marcus, A. and Maletic, J.I. 2003. Recovering Documentation-to-Source-Code Traceability Links using Latent Semantic Indexing. *Software Engineering, International Conference on* (Los Alamitos, CA, USA, 2003), 125.

[15] Tichy, W.F. 2000. Hints for Reviewing Empirical Work in Software Engineering. *Emp. Softw. Engg.* 5, (Dec. 2000), 309–312.