Do Better IR Tools Improve the Accuracy of Engineers' Traceability Recovery?

Markus Borg
Dept. of Computer Science
Lund University
Lund, Sweden
markus.borg@cs.lth.se

Dietmar Pfahl
Dept. of Computer Science
Lund University
Lund, Sweden
dietmar.pfahl@cs.lth.se

ABSTRACT

Large-scale software development generates an ever-growing amount of information. Multiple research groups have proposed using approaches from the domain of information retrieval (IR) to recover traceability. Several enhancement strategies have been initially explored using the laboratory model of IR evaluation for performance assessment. We conducted a pilot experiment using printed candidate lists from the tools RETRO and ReqSimile to investigate how different quality levels of tool output affect the tracing accuracy of engineers. Statistical testing of equivalence, commonly used in medicine, has been conducted to analyze the data. The low number of subjects in this pilot experiment resulted neither in statistically significant equivalence nor difference. While our results are not conclusive, there are indications that it is worthwhile to investigate further into the actual value of improving tool support for semi-automatic traceability recovery. For example, our pilot experiment showed that the effect size of using RETRO versus ReqSimile is of practical significance regarding precision and F-measure. The interpretation of the effect size regarding recall is less clear. The experiment needs to be replicated with more subjects and on varying tasks to draw firm conclusions.

Categories and Subject Descriptors

D.2.1 [**Software Engineering**]: Requirements/Specifications—*Traceability*

General Terms

Documentation, Experimentation

Keywords

requirements traceability, information retrieval, controlled experiment, equivalence testing

1. INTRODUCTION

Development and maintenance of software often result in information overload. Knowledge workers are in general forced to spend more and more time to extract useful information. Maintaining traceability links between software artifacts is one approach to structure the information space of software development projects. Introducing information taxonomies and manually maintaining links is, however, an approach that does not scale very well. As a result, several researchers have proposed to support traceability recovery with tools based on IR methods, utilizing the fact that artifacts often have textual content in natural language.

Traceability recovery tools are generally evaluated by verifying how many suggested links above a certain similarity threshold are correct compared to a hand-crafted set of correct links. The laboratory model of IR evaluation is applied to calculate the measures recall and precision, sometimes extended by a harmonic mean, the F-measure. Recall and precision are commonly reported using graphs, like the one shown in Figure 4. The "X" in Figure 4 marks the recall and precision values of RETRO [13] and ReqSimile [16] for the first ten links proposed by these tools. For "X", the recall and precision values of RETRO are 50% respectively 60% larger than those of ReqSimile. RETRO is a well-known traceability recovery tool. ReqSimile is a research tool developed at Lund University for the purpose to support information retrieval in the context of market-driven requirements engineering. However, since the real question is to what extent tools like RETRO and ReqSimile actually help engineers in performing traceability recovery tasks, one may wonder whether it is worthwhile to keep hunting for recall-precision improvements of traceability recovery tools.

To tackle this questions, we conducted a controlled experiment with 8 subjects to study how tool support affects the tracing process performed by engineers. The purpose of our experiment was to explore how the output from two traceability recovery tools, RETRO and ReqSimile, impacted a task requiring traceability information.

We analyzed our results using a test of equivalence, trying to prove that one treatment is indistinguishable from another. In equivalence testing, the null hypothesis is formulated such that the statistical test is a proof of similarity i.e., checking whether the tracing accuracies of engineers using different tools differ by more than a tolerably small amount Δ [20]. Our null hypothesis is the following:
Hypothesis: The engineers' accuracy of traceability recovery supported by RETRO *differs by more than* Δ compared to that supported by ReqSimile.

The alternative hypothesis is that the difference between the engineers' accuracies is smaller than Δ, which implies that the treatments can be considered equivalent. Accuracy is expressed in terms of recall and precision.

2. RELATED WORK

The last decade, several researchers proposed semi-automatic support to the task of traceability recovery. For example, traceability recovery tools have been developed implementing techniques based on algebraic or probabilistic models [1], data mining [23] and machine learning [19]. Several researchers have expressed the tracing task as an IR problem. The query in such a tool is typically the software artifact you want to link to other artifacts. The answer to a query is normally a ranked list of artifact suggestions, most often sorted by the level of textual similarity. The ranked list is analogous to the output of search engines used on the web. Items in the list can be either relevant or irrelevant for the given task.

In 2000, Antoniol *et al.* did pioneering work on traceability recovery when they used the standard vector space model (VSM) and probabilistic models to suggest links between source code and documentation in natural language [1]. Marcus and Maletic introduced Latent Semantic Indexing (LSI), another vector space approach, to recover traceability in 2003. Their work showed that LSI can achieve good results without the need for stemming, which is fundamental in VSM and the probabilistic models [15]. The same year Spanoudakis *et al.* used a machine learning approach to establish traceability links. By generating traceability rules from a set of artifacts given by the user, links were derived in the document set. Zhang *et al.* proposed automatic ontology population for traceability recovery [23]. They developed a text mining system to semantically analyze software documents. Concepts discovered by the system were used to populate a documentation ontology, which was then aligned with a source code ontology to establish traceability links.

Common to those papers is that they have a technical focus and present no or limited evaluations using software engineers solving real tasks. The majority of the published evaluations of traceability tools do not go beyond reporting recall-precision graphs or other measures calculated without human involvement. Exceptions include studies comparing subjects working with tool support to manual control groups. Huffman Hayes *et al.* developed a traceability recovery tool named RETRO and evaluated it using 30 student subjects [13]. The students were divided into two groups, one working with RETRO and the other working manually. Students working with the tool finished a requirements tracing task faster and with a higher recall than the manual group, the precision however was lower. De Lucia *et al.* conducted a controlled experiment with 32 students on the usefulness of supported traceability recovery [9]. They found that subjects using their tool completed a task related to tracing various software artifacts faster and more accurately than subjects working manually, i.e. without any support from a dedicated traceability recovery tool. In another study, De Lucia *et. al* observed 150 students in 17 software development projects and concluded that letting them use IR-based tool support is helpful when maintenance of traceability information is a process requirement [10]. An experiment similar to ours was conducted by Cuddeback *et. al*, using students and student artifacts [8]. They had 26 subjects vet candidate requirements traceability matrices (RTMs) of varying accuracy. They concluded that subjects receiving the most inaccurate RTMs drastically improved them and that subjects in general balanced recall and precision.

Several researchers proposed ways to obtain better tool output, either by enhancing existing tools implementing standard IR techniques, or by exploring new or combined approaches. Using a thesaurus to deal with synonymy is one proposed enhancement strategy explored by different researchers [18, 12]. Zou *et al.* investigated term based improvement strategies such as including a part-of-speech tagger to extract key phrases and using a project glossary to weight certain terms higher [24]. Recently, Cleland-Huang *et al.* [6] and Asuncion *et al.* [2] used a machine learning approach, Latent Direchlet Allocation, to trace requirements. Furthermore, Chen has done preliminary work on combining IR-methods and text mining in a traceability recovery tool and reported improved results [5].

Even though enhancements lead to better tool outputs in certain cases, their general applicability and the benefit they generate for engineers performing a specific task remain uncertain. Oliveto *et al.* studied the impact of using four different methods for traceability recovery. In their empirical study, VSM, LSI and the Jensen-Shannon method resulted in almost equivalent results wrt. tracing accuracy [17]. LDA however, while not resulting in better accuracy, was able to capture different features than the others. As far as we know, no studies except Cuddeback *et. al* [8], have been published comparing how different quality levels of tool output impact of an engineer in a specific traceability task. If more empirical studies with humans were available, one could conduct a meta-analysis to investigate this matter. Since this is not the case, our approach is instead to compare in an experimental setting the effect of using support tools with differently accurate outputs on traceability tasks performed by humans.

3. EXPERIMENTAL SETUP

This section describes the definition, design and setting of the experiment, following the general guidelines by Wohlin *et al.* [22]. An overview of our experimental setup is shown in Figure 1.

3.1 Experiment Definition and Context

The *goal* of the experiment was to study the tool-supported traceability recovery process of engineers, for the *purpose* of evaluating the impact of traceability recovery tools' accuracies, with respect to the engineers' accuracy of traceability recovery, from the *perspective* of a researcher evaluating whether quality variations between IR tool outputs significantly affect the tracing accuracy of engineers.

3.2 Subjects and Experimental Setting

The experiment was executed at Lund University, Sweden. Eight subjects involved in software engineering research participated in the study. Six subjects were doctoral students, two subjects were senior researchers. Most subjects had industrial experience of software development.

The experiment was conducted in a classroom setting, the subjects worked individually. Each subject was randomly seated and supplied with a laptop with two electronic documents containing the artifacts that were to be traced in PDF format. Each subject also received a printed list per

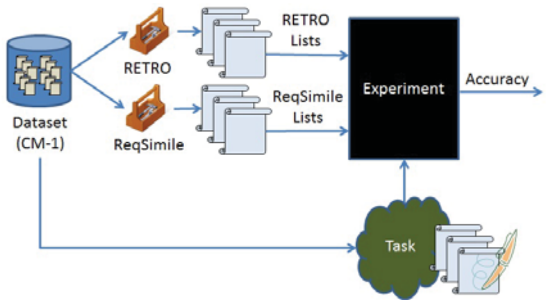


Figure 1: Overview of the experimental setup

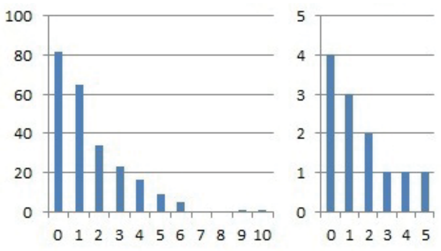


Figure 2: Histograms showing the link densities of CM-1 (left) and the subset used as the experimental sample (right).

Number of traceability links: 361		
Characteristic	High-level Reqs.	Low-level Reqs.
Items	235	220
Words	5 343	17 448
Words/Items	22.7	79.3
Avg. word length	5.2	5.1
Unique words	1 056	2 314
Gunning Fog Index	7.5	10.9
Flesch Reading Ease	67.3	59.6

Table 1: Statistics of the CM-1 data, calculated using the *Text Content Analyzer* on UsingEnglish.com.

artifact to trace, containing candidate links as described in section 3.5. Four subjects received lists with candidate links generated by RETRO, the other four received candidate lists generated by ReqSimile. The lists were distributed randomly. The subjects received a pen, a two-page instruction, an answer sheet and a debriefing questionnaire. The subjects were supposed to navigate the PDF documents as they preferred, using the candidate link lists as support. All individual requirements were clickable as bookmarks, and keyword searching using the Find tool of their PDF viewer was encouraged.

3.3 Task and Description of the Dataset

It was decided to reuse a publicly available dataset and a task similar to previous tracing experiments to enable comparison to old results. The task, in which traceability recovery was required, was to estimate impact of a change request on the CM-1 dataset. For twelve given requirements, the subjects were asked to identify related requirements on a lower abstraction level. The task was given a realistic scenario involving time pressure, by having the subjects assume they should present their results in a meeting 45 minutes later. Before the actual experiment started, the subjects were given a warm-up exercise to become familiar with the document structure and the candidate link lists.

The CM-1 data is a publicly available[1] set of requirements with complete traceability information. The data originates from a project in the NASA Metrics Data Program and has been used in several traceability experiments before [13, 14, 24]. The dataset specifies parts of a data processing unit and consists of 235 high-level requirements and 220 corresponding low-level requirements specifying detailed design. Many-to-many relations exist between abstraction levels. The link density of CM-1 and the representative subset used in the experiment are presented in Figure 2. This figure depicts histograms with the X-axis representing the number of low-level requirements related to one high-level requirement. Due to the rather unintuitive nature of the dataset, having many unlinked system requirements, the subjects received a hint saying that *"Changes to system requirements normally impact zero, one or two design items. Could be more, but more than five would really be exceptional"*.

Descriptive statistics of CM-1, including two commonly reported text complexity measures, are presented in Table 1. Farbey proposed calculating *Gunning Fog Index* as a complexity metric for requirement specifications written in English [11]. The second complexity metric reported is the *Flesch Reading Ease*, previously reported by Wilson *et al.* for requirement specifications from NASA [21].

[1]www.coest.org

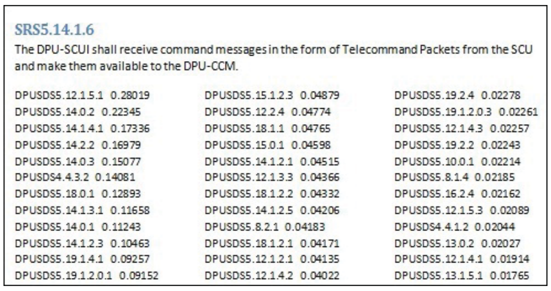
3.4 Decription of the Tools

RETRO, developed by Huffman Hayes *et al.*, is a tool that supports software development by tracing textual software engineering artifacts [13]. The tool generates RTMs using standard information retrieval techniques. The evolution of RETRO accelerated when NASA analysts working on independent verification and validation projects showed interest in the tool. The version of the software we used implements VSM with features having term frequency-inverse document frequency weights. Similarities are calculated as the cosine of the angle between feature vectors [3]. Stemming is done as a preprocessing step by default. For stop word removal, an external file must be provided, a feature we did not use. We used the RETRO version V.BETA, Release Date February 23, 2006.

ReqSimile, developed by Natt och Dag *et al.*, is a tool with the primary purpose to provide semi-automatic support to requirements management activities that rely on finding semantically similar artifacts [16]. Examples of such activities are traceability recovery and duplicate detection. The tool was intended to support the dynamic nature of market-driven requirements engineering. ReqSimile also implements VSM and cosine similarities. An important difference to RETRO is the feature weighting; terms are weighted as $1 + log_2(freq)$ and no inverse document frequencies are considered. Preprocessing steps in the tool include stop word removal and stemming. We used version 1.2 of ReqSimile.

3.5 Experimental Variables

In the context of the proposed experiment, the independent variable was the quality of the tool output given to the


SRS5.14.1.6

The DPU-SCUI shall receive command messages in the form of Telecommand Packets from the SCU and make them available to the DPU-CCM.

DPUSDS5.12.1.5.1	0.28019	DPUSDS5.15.1.2.3	0.04879	DPUSDS5.19.2.4	0.02278
DPUSDS5.14.0.2	0.22345	DPUSDS5.12.2.4	0.04774	DPUSDS5.19.1.2.0.3	0.02261
DPUSDS5.14.1.4.1	0.17336	DPUSDS5.18.1.1	0.04765	DPUSDS5.12.1.4.3	0.02257
DPUSDS5.14.2.2	0.16979	DPUSDS5.15.0.1	0.04598	DPUSDS5.19.2.2	0.02243
DPUSDS5.14.0.3	0.15077	DPUSDS5.14.1.2.1	0.04515	DPUSDS5.10.0.1	0.02214
DPUSDS4.4.3.2	0.14081	DPUSDS5.12.1.3.3	0.04366	DPUSDS5.8.1.4	0.02185
DPUSDS5.18.0.1	0.12893	DPUSDS5.18.1.2.2	0.04332	DPUSDS5.16.2.4	0.02162
DPUSDS5.14.1.3.1	0.11658	DPUSDS5.14.1.2.5	0.04206	DPUSDS5.12.1.5.3	0.02089
DPUSDS5.14.0.1	0.11243	DPUSDS5.8.2.1	0.04183	DPUSDS4.4.1.2	0.02044
DPUSDS5.14.1.2.3	0.10463	DPUSDS5.18.1.2.1	0.04171	DPUSDS5.13.0.2	0.02027
DPUSDS5.19.1.4.1	0.09257	DPUSDS5.12.1.2.1	0.04135	DPUSDS5.12.1.4.1	0.01914
DPUSDS5.19.1.2.0.1	0.09152	DPUSDS5.12.1.4.2	0.04022	DPUSDS5.13.1.5.1	0.01765

Figure 3: Example of top part of a candidate link list.

subjects. For each item to trace, i.e. for each high-level requirement, entire candidate link lists generated by the tools using default settings were used. No filtering was applied in the tools. The output varied between 47 and 170 items, i.e. each item representing a low-level requirement. An example of part of such a list is presented in Figure 3, showing high-level requirement SRS5.14.1.6 and the top part of a list of candidate low-level requirements and their cosine similarities. The two tools RETRO [13] and ReqSimile [16] are further described in Section 3.4. RETRO has outperformed ReqSimile wrt. accuracy of tool output in a previous experiment on the CM-1 dataset [4].

The lists were printed with identical formatting to ensure the same presentation. Thus, the independent variable was given two treatments, printed lists of candidate links ranked by RETRO (Treatment RETRO), and printed lists of candidate links ranked by ReqSimile (Treatment ReqSimile). The recall-precision graphs for the two tools on the experiment sample are presented in Figure 4, extended by the accuracy of the tracing results, i.e. the answer sets returned by subjects as described in Section 4.

The dependent variable, the outcome observed in the study, was the accuracy of the tracing result. Accuracy was measured in terms of recall, precision and F-measure. Recall measures the percentage of correct links traced by a subject, while precision measures the percentage of traced links that were actually correct. The F-measure is the harmonic mean of recall and precision. The time spent on the task was limited to 45 minutes, creating realistic time pressure. We also recorded the number of requirements traced by the subjects.

3.6 Experiment Design and Procedure

A completely randomized design was chosen. The experiment was conducted during one single session. The design was balanced, i.e. both treatments, RETRO and ReqSimile, were assigned to the same number of subjects. The two treatment were given to the subjects at random. Each subject received the same tasks and had not studied the system previously. When the 45 minutes had passed, the subjects were asked to answer a debriefing questionnaire.

3.7 Statistical Analysis

The null hypothesis was formulated as existence of a difference in the outcomes bigger than Δ. Δ defines the interval of equivalence, i.e., the interval where variation is considered to have no practical value. For this pilot study, we decided to set Δ to 0.05 for both recall, precision and F-measure. This means that finishing the task with 0.05 better or worse recall and precision does not have a practical value.

The two one-sided test (TOST) is the most basic form of equivalence testing used to compare two treatments. Confidence intervals for the difference between two treatments must be defined. In a TOST analysis, a (1 - 2α)100% confidence interval is constructed [20]. We selected $\alpha = 0.05$, thus we reject the null hypotheses that the outcomes of the treatments differ by at least Δ, if the 90% confidence interval for the difference is completely confined within the endpoints -Δ and +Δ. The 90% confidence intervals are calculated as follows:

$$\frac{\begin{array}{c} point\_estimate\_outcome_{RETRO} - \\ point\_estimate\_outcome_{ReqSimile} \pm \end{array}}{2.353\sqrt{std\_dev^2_{RETRO} + std\_dev^2_{ReqSimile}}}$$

4. RESULTS AND DATA ANALYSIS

The experiment had no dropouts and as a result we collected 8 valid answer sheets and debriefing questionnaires. The answer sets were compared to the gold standard available for the datasets and the corresponding values for recall (Rc), precision (Pr) and F-measure (F) were calculated. The descriptive statistics for Rc, Pr, F, and the number of requirements traced are presented in Table 2. We also calculated the effect sizes using Cohen's d (cf. last column in Table 2). Results from the questionnaire are shown in Table 3.

Most subjects experienced the task as challenging and did not have enough time to finish. The list of common acronyms provided to assist the subjects, as was done in a previous case study using the CM-1 dataset [13], was not considered enough to appropriately understand the domain. Generally, the subjects considered the printed candidate link lists as supportive and would prefer having tool support if performing a similar task in the future.

Table 4 characterizes the tool outputs of RETRO and ReqSimile as well as the tracing results provided by the subjects participating in the experiment. The upper part of the table shows the data for the treatment with RETRO, the lower part that for the treatment with ReqSimile. Each row in the table provides the following data: the ID of the high-level requirement (Req. ID), the number of low-level requirements suggested by the tool (#Links), the cosine similarities of the first and last link in the list of suggested low-level requirements (Sim. 1st link, Sim. last link), and for each subject (A to H) the number of reported links and the associated recall and precision (Sub. A: # / Rc / Pr). Bold values represent fully accurate answers. A hyphen indicates that a subject did not provide any data on that high-level requirement. IDs of high-level requirements printed in italics have no associated low-level requirements links, thus a correct answer would have been to report 0 links. For those requirements we define rc and pr equal to 1 if a subject actually reported 0 links, otherwise rc and pr equal 0. When subjects reported 0 links for high-level requirements that actually have low-level requirements, we define rc and pr equal to 0.

The number of high-level requirements the subjects had time to investigate during the experiment varied between three and twelve. On average, RETRO subjects investigated eight items and ReqSimile subjects investigated 8.75.

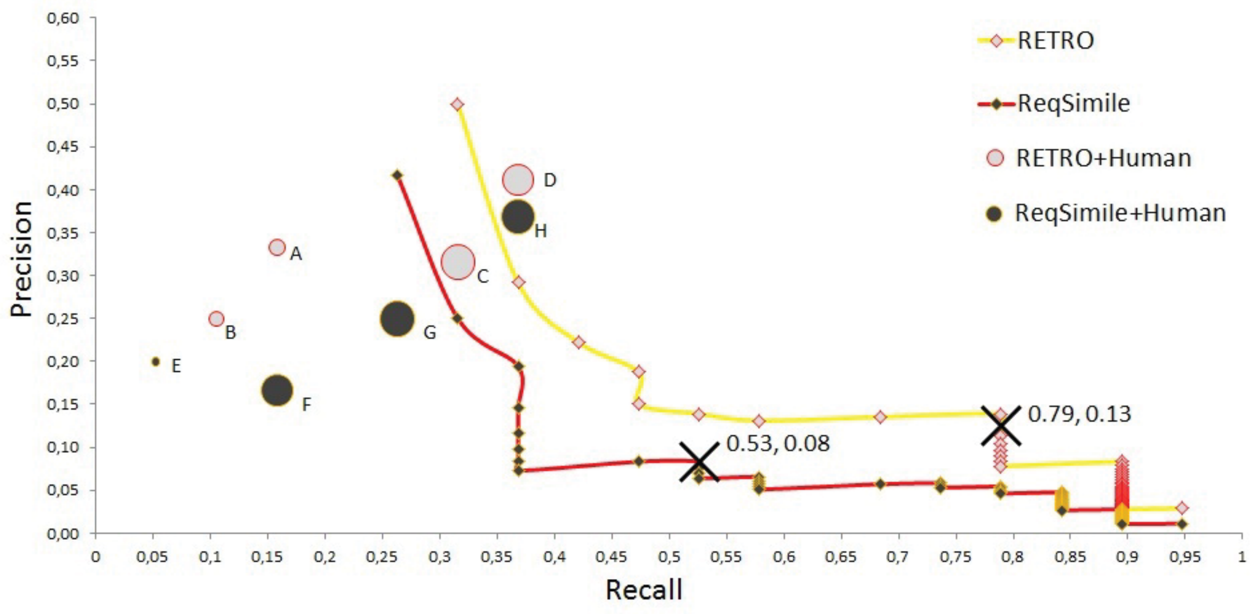


Figure 4: Recall-Precision graph for RETRO and ReqSimile for requirements tracing (our sample). The 'X'-symbols mark candidate link lists of length 10. Overall accuracy of answer sets returned by subjects is presented as circles, the diameter represents the relative number of links in the answer set. For a picture where also tool output is presented with relative sizes, see Figure 5.

All subjects apparently proceeded in the order the requirements were presented to them. Since subjects A and E investigated only three and four high-level requirements respectively, they clearly focused on quality rather than coverage. However, the precision of their tracing results does not reflect this focus. The mean recall for subjects supported by RETRO was higher than for subjects supported by ReqSimile, and also the mean precision. The standard deviations were however high, as expected when using few subjects. Not surprisingly, subjects reporting more links in their answer set reached higher recall values.

The debriefing questionnaire was also used to let subjects briefly describe their tracing strategies. Most subjects expressed focus on the top of the candidate lists. One subject reported the strategy of investigating the top 10 suggestions. Two subjects reported comparing similarity values and investigating candidate links until the first "big drop". Two subjects investigated links on the candidate lists until several in a row were clearly incorrect. Only one subject explicitly reported considering links after position 10. This subject investigated the first ten links, then every second until position 20, then every third until the 30th suggestion. This proved to be a a time-consuming approach and the resulting answer set was the smallest in the experiment. The strategies explained by the subjects are in line with our expectation that presenting more than 10 candidate links per requirement adds little value.

As Figure 4 shows, a naïve strategy of just picking the first one or two candidate links returned by the tools would in most cases result in better accuracy than the subjects achieved. Also, there is a trend that subjects supported by RETRO handed in more accurate answer sets. Pairwise comparison of subjects ordered according to accuracy, i.e. B to E, A to F, C to G, D to H, indicates that the better accuracy of RETRO actually spills over to the subjects' tracing result.

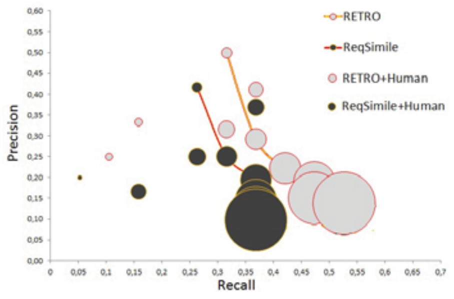


Figure 5: Circle diameters show relative number of links in answer sets. Tool output is plotted for candidate link lists of length from 1 to 6.

Figure 5 shows relative sizes of answer sets returned by both human subjects and the tools, presenting how the number of tool suggestions grows linearly. The majority of human answer sets contained between one or two links per requirement, comparable to tools generating one or two candidate links.

The 90% confidence intervals of the differences between RETRO and ReqSimile are presented in Figure 6. Since none of the 90% confidence intervals of recall, precision, and F-measure are covered by the interval of equivalence, there is *no statistically significant equivalence of the engineers' ac-*

TREATMENT	**Reqs. Traced** (number)			
	Mean	Median	Std. Dev.	Eff. Size
RETRO	8.00	8.50	2.74	
ReqSimile	8.75	10.0	3.70	-0.230
	Recall			
	Mean	Median	Std. Dev.	Eff. Size
RETRO	0.237	0.237	0.109	
ReqSimile	0.210	0.211	0.118	0.232
	Precision			
	Mean	Median	Std. Dev.	Eff. Size
RETRO	0.328	0.325	0.058	
ReqSimile	0.247	0.225	0.077	1.20
	F-Measure			
	Mean	Median	Std. Dev.	Eff. Size
RETRO	0.267	0.265	0.092	
ReqSimile	0.218	0.209	0.116	0.494

Table 2: Descriptive statistics of experimental results.

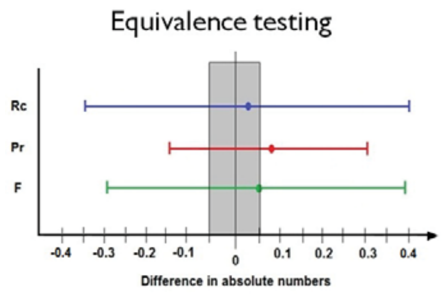


Figure 6: Differences in recall, precision and F-measure between RETRO and ReqSimile. The horizontal T-shaped bars depict confidence intervals. The interval of equivalence is the grey-shaded area.

curacies of traceability recovery, when using our choice of Δ. For completeness, we also did difference testing with the null hypothesis: The engineers' accuracy of traceability recovery supported by RETRO is equal to that supported by ReqSimile. This null hypothesis could not be rejected neither by a two-sided T-test nor a two-sided Wilcoxon rank-sum test with α=0.05. Consequently, there were *no statistically significant differences on the engineers' accuracies of traceability recovery* when supported by candidate link lists from different tools.

Our tests of significance are accompanied by effect-size statistics. Effect size is expressed as the difference between the means of the two samples divided by the root mean square of the variances of the two samples. On the basis of the effect size indices proposed by Cohen, effects greater or equal 0.5 are considered to be of medium size, while effect sizes greater or equal than 0.8 are considered large [7]. The effect sizes for precision and F-measure are high and medium respectively. Most researchers would consider them as being of practical significance. For recall, the effect size is too small to say anything conclusive.

5. THREATS TO VALIDITY

The entire experiment was done during one session, lowering the risk of maturation. The total time for the experiment was less than one hour to help subjects keep focused. As the answers in the debriefing questionnaire suggests, it is likely that different subjects had different approaches to the process of artifact tracing, and the chosen approach might have influenced the outcome more than the different treatments. This is a threat to the internal validity that. The fully randomized experiment design was one way to mitigate such effects. Future replications should aim at providing more explicit guidance to the subjects.

A possible threat to construct validity is that using printed support when tracing software artifacts is not representing how engineers would actually interact with the supporting IR tools, but it straightens the internal validity.

The CM-1 dataset used in the experiment, has been used in several previous tracing experiments and case studies. The dataset is not comparable to a large-scale industrial documentation space but is a representative subset. The CM-1 dataset originates from a NASA project, and is probably the most referenced dataset for requirements tracing. The subjects all do research in software engineering, most have also worked as software developers in industry. Since the dataset is complex, dealing with embedded development in a safety critical domain, it was challenging for the subjects to fully understand the information they received to perform their task. In that regard, the subjects are comparable to newly employed software engineers in industry. This limits the generalizability of the experimental results, as software engineers normally would be more knowledgeable.

QUESTIONS		
(1=Strongly agree, 5=Strongly disagree)	RETRO	ReqSimile
1. I had enough time to finish the task.	4.0	3.3
2. The list of acronyms gave me enough understanding of the domain to complete the task.	4.3	3.8
3. The objectives of the task were perfectly clear to me.	2.5	1.5
4. I experienced no major difficulties in performing the task.	3.3	4.3
5. The tool output (proposed links) really supported my task.	2.3	2.0
6. If I was performing a similar task in the future, I would want to use a software tool to assist.	2.3	1.8

Table 3: Results from the debriefing questionnaire. All questions were answered using a five-level Likert item. The means for each group are shown.

6. DISCUSSION AND FUTURE WORK

The results of the pilot experiment are inconclusive. The low number of subjects did not enable us to collect strong empirical evidence. The equivalence test (TOST) did not reject difference and the difference test (two-sided T-test) did not reject similarity. Thus, in this experiment, neither the evidence against equality nor difference was strong enough to reject either null hypothesis.

Although not statistically significant, we could see a trend that subjects supported with the better tool performed more

Treatment RETRO							
Req. ID	**#Links**	**Sim. 1st link**	**Sim. last link**	**Sub. A # / Rc / Pr**	**Sub. B # / Rc / Pr**	**Sub. C # / Rc / Pr**	**Sub. D # / Rc / Pr**
SRS5.1.3.5	134	0.551	0.019	**1 / 1 / 1**	**1 / 1 / 1**	3 / 1 / 0.33	**1 / 1 / 1**
SRS5.1.3.9	116	0.180	0.005	4 / 0.2 / 0.25	1 / 0 / 0	1 / 0 / 0	2 / 0 / 0
SRS5.12.1.11	156	0.151	0.004	2 / 0 / 0	2 / 0 / 0	2 / 0 / 0	1 / 0 / 0
SRS5.12.1.8	125	0.254	0.005	2 / 0.5 / 0.5	0 / 0 / 0	3 / 0.5 / 0.33	0 / 0 / 0
SRS5.14.1.6	101	0.280	0.006	-	1 / 0 / 0	1 / 0.25 / 1	3 / 0.25 / 0.33
SRS5.14.1.8	117	0.173	0.005	-	1 / 0 / 0	**0 / 0 / 0**	1 / 0 / 0
SRS5.18.4.3	47	0.136	0.009	-	2 / 1 / 0.5	3 / 1 / 0.3	2 / 1 / 0.5
SRS5.19.1.10	135	0.140	0.004	-	-	1 / 0 / 0	1 / 0 / 0
SRS5.19.1.2.1	101	0.151	0.006	-	-	0 / 0 / 0	2 / 1 / 0.5
SRS5.2.1.3	127	0.329	0.005	-	-	3 / 0.67 / 0.67	4 / 0.67 / 0.5
SRS5.9.1.1	163	0.206	0.003	-	-	2 / 0 / 0	-
SRS5.9.1.9	159	0.240	0.005	-	-	-	-
Treatment ReqSimile							
Req. ID	**#Links**	**Sim. 1st link**	**Sim. last link**	**Sub. E # / Rc / Pr**	**Sub. F # / Rc / Pr**	**Sub. G # / Rc / Pr**	**Sub. H # / Rc / Pr**
SRS5.1.3.5	145	0.568	0.004	3 / 1 / 0.33	4 / 1 / 0.25	2 / 1 / 0.5	**1 / 1 / 1**
SRS5.1.3.9	142	0.318	0.029	1 / 0 / 0	2 / 0 / 0	3 / 0 / 0	1 / 0 / 0
SRS5.12.1.11	166	0.315	0.029	1 / 0 / 0	1 / 0 / 0	2 / 0 / 0	**0 / 1 / 1**
SRS5.12.1.8	111	0.335	0.022	-	0 / 0 / 0	1 / 0 / 0	4 / 0.5 / 0.25
SRS5.14.1.6	134	0.397	0.021	-	4 / 0.25 / 0.25	3 / 0.25 / 0.33	2 / 0.25 / 0.5
SRS5.14.1.8	170	0.397	0.029	-	2 / 0 / 0	2 / 0 / 0	2 / 0 / 0
SRS5.18.4.3	143	0.259	0.021	-	3 / 1 / 0.33	**1 / 1 / 1**	1 / 0 / 0
SRS5.19.1.10	160	0.340	0.025	-	2 / 0 / 0	**0 / 1 / 1**	3 / 0 / 0
SRS5.19.1.2.1	146	0.433	0.021	-	-	2 / 0 / 0	3 / 1 / 0.66
SRS5.2.1.3	151	0.619	0.018	-	-	2 / 0.66 / 1	1 / 0.33 / 1
SRS5.9.1.1	167	0.341	0.019	-	-	2 / 0 / 0	0 / 1 / 1
SRS5.9.1.9	157	0.527	0.018	-	-	1 / 0 / 0	**1 / 1 / 1**

Table 4: Characterization of tool outputs and tracing results provided by the subjects participating in the experiment.

accurately. Somewhat surprisingly, the precision of the subjects was not higher than that of the results the tools produced. For our specific task, with a time pressure, just using the tool output would generally be better than letting our subjects solve the task, using the tool output as support. One could argue that our subjects actually made the results worse. One direction of future research could be to explore under which circumstances this is the case.

Our experiment is in line with the finding of Cuddeback et. al [8], stating that subjects seem to balance recall and precision. We observed this trend in a very different experimental setup. Foremost, our task was to trace a subset of artifacts under time pressure using printed candidate link lists as support, as opposed to vet a complete RTM without time pressure using a tool. Other differences include: types of subjects and artifacts, and a sparser golden standard RTM.

Is it meaningful to study a tracing task with subjects that are not very knowledgeable in the domain? Embedded software development in the space industry is not easy to simulate in a classroom setting. Yet there is a need to understand the return on investment of improved accuracy of IR tools in support of traceability recovery. Controlled experiments can be one step forward. Our ambition is to replicate this pilot experiment using a larger number of student subjects, to explore whether any statistically significant results appear.

Recall and precision of traceability recovery tools are not irrelevant measures, but the main focus of research should be broader. For example, Figure 4 shows that the accuracy of RETRO is clearly better than that that of ReqSimile. However, the effect of using RETRO in a specific traceability recovery task is not that clear, as our pilot experiment suggests. Therefore, our future work aims at moving closer to research about information access, especially to the subdomain of enterprise search, where more holistic approaches are explored. For example, we think that assessing quality aspects such as findability in the software engineering context would mature the traceability recovery research.

Acknowledgements

This work was funded by the Industrial Excellence Center EASE - Embedded Applications Software Engineering[2]. Parts of this work were funded by the Excellence Center at Linköping - Lund in Information Technology (ELLIIT).

7. REFERENCES

[1] G. Antoniol, G. Canfora, A. De Lucia, and G. Casazza. Information retrieval models for recovering traceability links between code and documentation. In *IEEE International Conference on Software Maintenance*, pages 40–49, 2000.

[2] H. U. Asuncion, A. U. Asuncion, and R. N. Taylor. Software traceability with topic modeling. In *32nd*

[2]http://ease.cs.lth.se

ACM/IEEE International Conference on Software Engineering, pages 95–104, 2010.

[3] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. Addison-Wesley, 1999.

[4] L. Brodén. Requirements traceability recovery - a study of available tools. Master's thesis, Dept. Computer Science, Lund University, http://sam.cs.lth.se/ExjobGetFile?id=377, 2011.

[5] X. Chen. Extraction and visualization of traceability relationships between documents and source code. In *IEEE/ACM International Conference on Automated Software Engineering*, pages 505–510, 2010.

[6] J. Cleland-Huang, A. Czauderna, M. Gibiec, and J. Emenecker. A machine learning approach for tracing regulatory codes to product specific requirements. In *ACM/IEEE International Conference on Software Engineering*, pages 155–164, 2010.

[7] J. Cohen. *Statistical Power Analysis for the Behavioral Sciences*. Academic Press, New York, NY, 1988.

[8] D. Cuddeback, A. Dekhtyar, and J. Huffman Hayes. Automated requirements traceability: The study of human analysts. In *Proceedings of the 18th International Requirements Engineering Conference*, pages 231 – 240, Sydney, 2010.

[9] A. De Lucia, F. Fasano, R. Oliveto, and G. Tortora. Recovering traceability links in software artifact management systems using information retrieval methods. *ACM Trans. Softw. Eng. Methodol.*, 16(4):13, 2007.

[10] A. De Lucia, R. Oliveto, and G. Tortora. Assessing IR based traceability recovery tools through controlled experiments. *Empirical Software Engineering*, 14(1):57 – 92, 2009.

[11] B. Farbey. Software quality metrics: considerations about requirements and requirement specifications. *Information and Software Technology*, 32(1):60–64, 1990.

[12] J. Huffman Hayes, A. Dekhtyar, and S. Sundaram. Advancing candidate link generation for requirements tracing: The study of methods. *IEEE Transactions on Software Engineering*, 32:4–19, 2006.

[13] J. Huffman Hayes, A. Dekhtyar, S. Sundaram, E. Holbrook, S. Vadlamudi, and A. April. REquirements TRacing on target (RETRO): improving software maintenance through traceability recovery. *Innovations in Systems and Software Engineering*, 3:193–202, 2007.

[14] A. Mahmoud and N. Niu. Using semantics-enabled information retrieval in requirements tracing: An ongoing experimental investigation. *Annual International Computer Software and Applications Conference*, pages 246–247, 2010.

[15] A. Marcus and J. I. Maletic. Recovering documentation-to-source-code traceability links using latent semantic indexing. In *ICSE '03: Proceedings of the 25th International Conference on Software Engineering*, pages 125–135, 2003.

[16] J. Natt och Dag, V. Gervasi, S. Brinkkemper, and B. Regnell. A linguistic-engineering approach to large-scale requirements management. *IEEE Softw.*, 22(1):32–39, 2005.

[17] R. Oliveto, M. Gethers, D. Poshyvanyk, and A. De Lucia. On the equivalence of information retrieval methods for automated traceability link recovery. In *2010 IEEE 18th International Conference on Program Comprehension*, pages 68–71, 2010.

[18] R. Settimi, J. Cleland-Huang, O. Ben Khadra, J. Mody, W. Lukasik, and C. DePalma. Supporting software evolution through dynamically retrieving traces to UML artifacts. In *7th International Workshop on Principles of Software Evolution*, pages 49–54, Kyoto, Japan, 2004.

[19] G. Spanoudakis, A. d'Avila-Garces, and A. Zisman. Revising rules to capture requirements traceability relations: A machine learning approach. In *Proceedings of the 15th International Conference in Software Engineering and Knowledge Engineering (SEKE 2003)*, pages 570–577, San Francisco, 2003.

[20] S. Wellek. *Testing Statistical Hypotheses of Equivalence*. Chapman & Hall/CRC, Boca Raton, FL, 2002.

[21] W. Wilson, L. Rosenberg, and L. Hyatt. Automated analysis of requirement specifications. *International Conference on Software Engineering*, pages 161–171, 1997.

[22] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, and A. Wesslen. *Experimentation in Software Engineering An Introduction*. Kluwer Academic Publishers, 2000.

[23] Y. Zhang, R. Witte, J. Rilling, and V. Haarslev. Ontological approach for the semantic recovery of traceability links between software artefacts. *Software, IET*, 2(3):185–203, June 2008.

[24] X. Zou, R. Settimi, and J. Cleland-Huang. Improving automated requirements trace retrieval: a study of term-based enhancement methods. *Empirical Software Engineering*, 15:119–146, 2010.