A Machine Learning Approach for Tracing Regulatory Codes to Product Specific Requirements

Jane Cleland-Huang, Adam Czauderna, Marek Gibiec, and John Emenecker
Systems and Requirements Engineering Center (SAREC)
DePaul University
Chicago, USA

jhuang@cs.depaul.edu; {aczauderna, mgibiec, jaemenecker}@gmail.com

ABSTRACT

Regulatory standards, designed to protect the safety, security, and privacy of the public, govern numerous areas of software intensive systems. Project personnel must therefore demonstrate that an as-built system meets all relevant regulatory codes. Current methods for demonstrating compliance rely either on after-the-fact audits, which can lead to significant refactoring when regulations are not met, or else require analysts to construct and use traceability matrices to demonstrate compliance. Manual tracing can be prohibitively time-consuming; however automated trace retrieval methods are not very effective due to the vocabulary mismatches that often occur between regulatory codes and product level requirements. This paper introduces and evaluates two machine-learning methods, designed to improve the quality of traces generated between regulatory codes and product level requirements. The first approach uses manually created traceability matrices to train a trace classifier, while the second approach uses web-mining techniques to reconstruct the original trace query. The techniques were evaluated against security regulations from the USA government's Health Insurance Privacy and Portability Act (HIPAA) traced against ten healthcare related requirements specifications. Results demonstrated improvements for the subset of HIPAA regulations that exhibited high fan-out behavior across the requirements datasets.

Categories and Subject Descriptors

D.2.1 [**Software Engineering]:** Requirements/Specifications; H.3.3 [**Information Storage and Retrieval**: Information Search and Retrieval.

General Terms

Documentation, Legal Aspects

Keywords

Traceability, requirements classification, regulatory compliance.

1. INTRODUCTION

Software systems designed to support safety, security, or financially critical applications must conform to an increasing number of regulatory codes. For example, financial software systems in the USA must comply with the Sarbanes-Oxley act of 2002 (SOX), which establishes wide ranging standards for all U.S. public company boards, management, and public accounting firms. Similarly, all healthcare related products in the USA must comply with the Health Insurance Portability and Accountability Act (HIPAA) [1] which requires covered entities to use administrative and technical safeguards to protect patient medical information and governs privacy-related practices, including the use and disclosure of patient medical information for patient care and research. Embedded software systems often have to satisfy a staggeringly large number of regulatory codes. For example, two authors of this paper were recently involved in the traceability of a large software intensive system that included over 30,000 requirements and 300 different sets of relevant regulatory codes [4]. Traceability costs for this system alone were estimated by the industrial requirements specialist at over USD$3 Million!

Current methods for demonstrating regulatory compliance rely on the standard software engineering practice of *requirements traceability*, which refers to the "ability to track a requirement from its origins back to its rationale and downstream to various work products that implement that requirement in software" [15]. Full life-cycle compliance can be achieved by tracing regulatory codes to product level requirements and from there to design documents, code, and test cases. For example, the U.S. Food and Drug Administration (FDA) states that traceability analysis must be used to verify that a software design implements all of its specified software requirements, that all aspects of the design are traceable to software requirements, and that all code is linked to established specifications and established test procedures.

To trace regulatory codes, software developers are often forced to manually pore over documentation manuals to identify relevant sections, and then painstakingly trace them to the product level requirements or implemented code [4]. Traces are typically documented in a trace matrix. Unfortunately, numerous case studies have shown that organizations struggle to implement successful and cost-effective traceability, primarily because creating, maintaining, and using traces is a time-consuming, costly, arduous, and error prone activity [7, 15, 16, 25].

These traceability problems have been partially addressed through automated tracing methods using the Vector Space Model, Latent Semantic Indexing, probabilistic networks [2, 7, 20, 22] and other similar approaches, which return a set of candidate links, in much the same way that Google returns results in response to a user's query for information. Automated methods have generally been quite effective, returning a candidate set of traces that contain 85-90% of the targeted links at precision rates of 10-50%. Unfortunately, these methods have limited success for tracing regulatory codes[4] due to the significant disparity in terminology that can exist between the codes and product level requirements.

ICSE '10, May 2-8 2010, Cape Town, South Africa

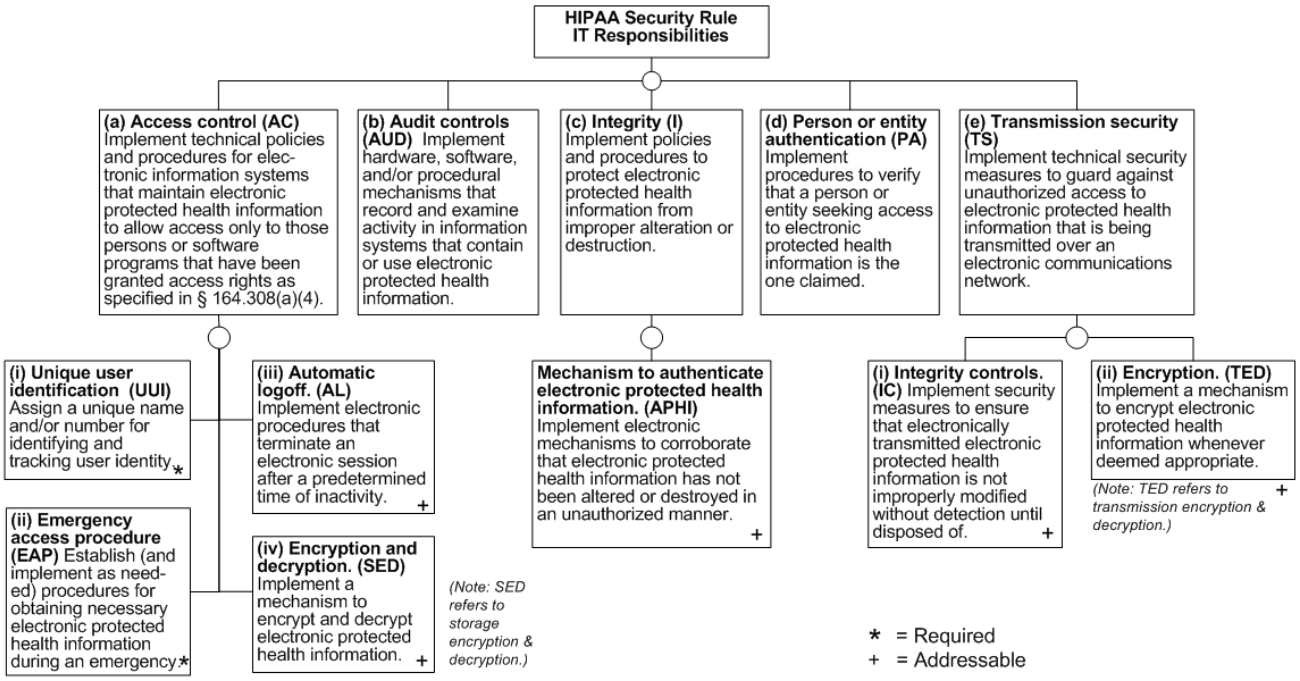


Figure 1. HIPAA regulations to be implemented in software.

In this paper we introduce and empirically evaluate two machine learning techniques designed to improve trace results from regulatory codes to product level requirements. Both techniques resulted in improvements for the subset of HIPAA regulations that were hard to trace using traditional trace-retrieval methods. The first approach uses a manually created traceability matrix to train a classifier to trace regulatory codes, while the second approach utilizes a novel information retrieval method to mine terms and phrases from domain specific documents in order to replace or augment the original trace query. Both techniques are illustrated and evaluated against the technical safeguards of the USA's Health Insurance Portability and Accountability Act (HIPAA) of 1996. This act defines administrative, physical, and technical safeguards designed to reduce the unauthorized use and disclosure of private healthcare related data [1]. The security safeguards, which include requirements for access control, audit controls, authentication, and transmission security, are depicted in Figure 1.

The experiments described in this paper are based on the software requirements specifications (SRS) for ten patient healthcare products. These SRS documents, which are summarized in Table 1, were obtained from a variety of sources including open source products, IT healthcare standards, requirements exemplars, and feature descriptions for commercial products. In cases where requirements were extracted from product-level documentation or from online forums, care was taken to retain the original language and text, and to ensure that no additional terminology was added. Four members of the research team at DePaul University manually constructed and validated traceability links between the HIPAA regulations and requirements in the SRS documents. For illustrative purposes, the traceability matrix for the *Automatic Logout* (AL) regulation is depicted in Table 2.

The next section of this paper describes a baseline experiment in which a standard trace retrieval method is used to generate traces between regulatory codes and the healthcare requirements. Sections 3 and 4 then describe the two proposed machine learning techniques and evaluate their effectiveness in comparison to the baseline. Sections 5 to 7 analyze these results, address the generality of the approach and discuss threats to validity. Finally Section 7 discusses related work, and section 8 concludes with a discussion of future work.

2. BASIC TRACE RETRIEVAL

As previously stated, there are many different approaches for automatically generating traces including the vector space model, probabilistic network (PN) models, and latent semantic indexing (LSI) [2, 7, 19, 23], however various studies have shown only minor and inconsistent differences between results obtained from these methods. The PN model is therefore used in our baseline experiment for pragmatic reasons as it was adopted in the tracing tool we developed for conducting industrial pilot studies [7].

2.1 The probabilistic tracing model

To prepare text for tracing, a pre-parsing phase is conducted to remove common words known as 'stop' words, and to stem the remaining words to their root forms. A similarity score is then computed based upon the frequency and distribution of terms in the regulations and requirements to be traced. In the probabilistic network method, the basic probability of a link between a query artifact q and a traceable requirement r_j is defined as follows:

$$pr(r_j|q) = \frac{\left[\sum_{i=1}^{k} pr(r_j|t_i) pr(q,t_i)\right]}{pr(q)} \quad (1)$$

The first two parts of the formula, $pr(r_j|t_i)$ and $pr(q,t_i)$ represent the dispersion of the term t_i within the requirement r_j and query q (i.e. a regulatory code) respectively. These are estimated by computing the normalized frequency of terms. For example $pr(r_j|t_i)$ is calculated by considering the frequency with which t_j occurs in the requirement, normalized over the total number of words in the requirement. This is represented as $pr(r_j|t_i) = freq(r_j,t_i/\sum_k freq(r_j,t_k)$. Similarly $pr(q,t_i)$ is calculated by

Table 1. Healthcare related datasets used in experiments.

ID	Description	Source	All Reqs	HIPAA Related Requirements											
				Tot	AC	AUD	AL	EAP	I	PA	SED	TED	TS	IC	JUI
1	**Care2x**: An open source Hospital Info. System (HIS).	Care2x User Manual and Forum. http://www.care2x.org	44	6	1	1	1	0	0	1	1	1	0	0	0
2	**CCHIT:** Reqs for Ambulatory, Health Info Exchange, EHR certification.	The Certification Commission for Healthcare Technology http://www.cchit.org	1064	78	17	33	1	1	5	12	2	2	2	0	3
3	**ClearHealth**: Open source HER.	ClearHealth Web Site http://www.clear-health.com	44	11	1	4	1	0	1	0	1	1	0	1	1
4	**Physician:** Electronic exchange of info between clinicians.	Use case documents http://www.hmss.org/content/files/CTC_use_Case.pdf	147	15	7	2	0	2	3	0	0	0	1	0	0
5	**iTrust:** Open source HER North Carolina State University.	Use case documents http://agile.csc.ncsu.edu/itrust/wiki/doku.php	184	47	3	35	1	0	0	6	0	0	0	0	2
6	**Trial Implementations:** Priority info exchange. National Coord. for Health Info. technology (HIT).	Use case documents http://healthit.hhs.gov	100	31	4	6	0	0	4	13	0	0	2	0	2
7	**PatientOS**: Open source healthcare info. System.	PatientOS Web Site http://www.patientos.org	90	13	1	2	3	1	0	0	3	1	1	0	1
8	**PracticeOne**: A Suite of healthcare info. Systems.	Practice One Website http://www.practiceone.com	34	7	3	1	0	0	1	1	0	0	1	0	0
9	**Lauesen**: Sample requirements specification for an HER	Sample specs: http://www.itu.dk/~slauesen/Papers/RequirementsSL-07.doc	66	21	11	0	1	0	3	5	0	0	0	0	1
10	**WorldVistA**: Open source version of Veteran Administrations EHR	VisA Manuals, HER Application Help files http:/worldvista.org	116	15	6	2	2	0	0	4	0	0	0	0	1
		Totals:	1889	244	54	86	10	4	17	42	7	5	7	1	11

Table 2. Requirements traced to HIPAA regulation for Automatic Logout.

Implement electronic procedures that terminate an electronic session after a predetermined time of inactivity.		
Dataset	**Req ID**	**Requirement**
ClearHealth	ch37f	System has a timer allowing automatic logoff
Care2x	x5	System will implement session time outs and use cookies to terminate an electronic session
SL	SL28	The system must time out if the user has been away from the system for some time.
iTrust	iT1.4	Electronic session must terminate after a pre-determined period of inactivity. Administrator must be able to specify this period
World Vista	WV45	The system shall timeout after a period of inactivity.
World Vista	WV45-1	The system shall ask the user if the user wants to continue using the system before timing out.
PatientOS	POS85	Automatic timeout can be specified by location.
PatientOS	POS84	Automatic timeout can be specified by role.
PatientOS	POS86	When the system timeouts, the user is returned to the login page to sign in again.
CCHIT	CCHIT 0539	The system upon detection of inactivity of an interactive session shall prevent further viewing and access to the system by that session by terminating the session, or by initiating a session lock that remains in effect until the user reestablish-es access using appropriate identification and authentication procedures. The inactivity timeout shall be configurable.

considering the frequency at which term t_i occurs in query q, normalized over the total number of potential queries in which t_i occurs.

In the third part of the formula, *pr(q)* is computed as $pr(q) = \sum_i pr(q, t_i)$, using simple marginalization techniques, and represents the relevance of the term t_i to describe the query concept q. The resulting probability is inversely proportional to the number of requirements containing the index term, reflecting the assumption that rarer index terms are more relevant than common ones in detecting potential links. A more complete description of the probabilistic network model is provided in several related papers [9, 31].

2.2 Trace retrieval metrics

Experimental information retrieval results are typically evaluated using recall and precision metrics [26]. Recall measures the fraction of relevant links that are retrieved and is computed as:

$$Recall = \frac{Relevant\ links \cap Retrieved\ Links}{Relevant\ Links}$$

while precision measures the fraction of retrieved links that are relevant [26] and is computed as:

$$Precision = \frac{Relevant\ Links \cap Retrieved\ Links}{Retrieved\ Links}$$

For experimental purposes, a threshold score is established such that all links above or at the threshold are retrieved and all links below the threshold are rejected.

Because it is not feasible to achieve identical recall values across every trace query, it can be difficult to compare recall and precision results across experiments. Another metric known as the F-Measure, which computes the harmonic mean of recall and precision is therefore often used for comparative purposes. In this paper we adopt a variant of the F-measure, known as the F_2-Measure, which weights recall values more highly than precision. This weighting is appropriate in the traceability domain where it is essential to recall as many of the correct links as possible. The version of F_2 Measure used in this paper is computed as follows:

$$F_2 Measure = \frac{3 \times Precision \times Recall}{(2 \times Precision) + Recall}$$

Recall, precision, and F-Measure metrics are unable to differentiate between results in which relevant links are closer to the top of the ranked listing of results, versus those in which relevant links are lower down in the list. We therefore introduce one additional metric, Average Precision, which favors results that return relevant links nearer to the top of the list. An additional benefit of this metric is that it does not depend on an arbitrary threshold value. Average precision is computed as:

$$Average\ Precision = \frac{\sum_{r=1}^{N}(P(r) \times relevant(r))}{Number\ of\ relevant\ documents}$$

where r is the rank, N is the number of retrieved documents, relevant() is a binary function assigned 1 if the rank is relevant and 0 otherwise, and P(r) is the precision computed after truncating the list immediately below that ranked position. These four metrics are used throughout the remainder of the paper to document and compare results from each of the experiments.

2.3 Basic Automated Tracing Experiment

In the first experiment, the probabilistic network model described in section 2.1 was used to generate traces from the ten targeted HIPAA regulations shown in Figure 1 to requirements in each of the 10 patient healthcare systems. Trace queries were formulated directly from the text of each HIPAA security regulation. Recall, precision, and F_2-measure metrics were computed individually for each of the HIPAA regulations by establishing threshold values that optimized the F_2-measure. The decision to optimize recall and precision across each regulatory type was used consistently for all experiments reported in this paper. Average precision, which is not dependent upon a threshold value was also computed. The results are reported in Table 3.

For four of the HIPAA regulations, namely *storage encryption*, *emergency access procedures*, *transmission encryption*, and *transmission security* it was possible to recall all of the relevant requirements. For many of the HIPAA regulations, the precision values were unacceptably low, in some cases returning precision values close to 0.02, however; even for regulations with lower levels of precision, the trace performed the useful function of excluding a large number of unlikely links. For example, the *transmission encryption* regulation returned recall results of 1.000 and precision of only 0.06, but successfully excluded 1,806 of the potential 1,889 links. Average precision results were highest for *automatic logoff*, *emergency access, and storage encryption,* meaning that for these HIPAA regulations, the trace retrieval algorithm returned results that were generally closer to the top of the list than for the other regulations.

Unfortunately, prior studies have demonstrated that users lose confidence in a traceability tool that returns imprecise results. Furthermore human error is introduced when human analysts are asked to evaluate a long list of candidate links [20]. The remainder of this paper therefore describes two techniques for improving the accuracy of traces for regulatory codes.

3. MACHINE LEARNING APPROACH

Machine learning methods are particularly appealing for tracing regulatory codes, because the upfront effort of training a classifier can be potentially recouped when those same codes are applied across future projects. Although there are many potential classification techniques, we adopted an algorithm that was previously developed for classifying non-functional requirements [10]. Prior studies demonstrated that this algorithm matched or outperformed standard classification techniques including the naïve bayes classifier, standard decision tree algorithm (J48), feature subset selection (FSS), correlation-based feature subset selection (CFS), and various combinations of the above for the specific task of classifying NFRs in the studied datasets [21].

Table 3. Results from basic trace retrieval

HIPAA regulation	Max recall	Precision	F2-Measure	Avg precision
Access Control (AC)	0.944	0.038	0.105	0.176
Audit Control (AUD)	0.919	0.049	0.133	0.144
Automatic logoff (AL)	0.900	0.225	0.450	0.626
Transmission Encrypt (TED)	1.000	0.060	0.161	0.373
Emergency Access (EAP)	1.000	0.333	0.600	0.542
Storage Encryption (SED)	1.000	0.778	0.913	0.933
Integrity (I)	0.529	0.020	0.057	0.073
Personal authentication (PA)	0.571	0.063	0.154	0.082
Transmission security (TS)	1.000	0.100	0.250	0.294
Unique user ID (UUI)	0.727	0.140	0.304	0.263

3.1 The Classification Process

The trace classification process includes the three primary stages of preparation, training, and classifying.

During the preparatory stage, a training set of regulatory codes, product level requirements, and their associated traces is constructed. Each requirement is then pre-processed using the techniques described in section 2.1.

A probabilistic weight is assigned to each term found in the requirements with respect to each of the regulatory codes. This weight reflects the degree to which a term represents a specific regulatory code. For example, the term *timeout* occurs frequently in requirements related to the HIPAA rule for *automatic logoff* and occurs much less frequently in other types of requirements. It is therefore assigned a relatively strong weighting with respect to *automatic logoff*. In contrast, a term such as *session,* which is found not only in *automatic logoff* requirements but also across a variety of other types of requirements, is assigned a much lower indicator weighting with respect to the *automatic logoff* regulation. The formulas for weighting indicator terms are as follows.

Let q be a specific regulation such as the HIPAA rule for *automatic logoff*. Indicator terms of type q are mined by considering the set S_q of all requirements in the training set that are related to regulation q. The cardinality of S_q is defined as N_q. Each term t is assigned a weight score $Pr_q(t)$ that measures how well the term identifies a requirement that matches regulation q. The weight score $Pr_q(t)$ corresponds to the probability that a particular term t identifies a requirement as being associated to regulation q based on the standard Information Retrieval assumption that terms indicating relevance for a certain targeted regulation must be present in the document to be classified. The frequency $freq(r_q,t)$ of occurrence of term t in requirement r_q is computed for each requirement in S_q. $Pr_q(t)$ is then computed as:

$$Pr_q(t) = \frac{1}{N_q}\sum_{r_q \in S_q} \frac{freq\ (r_q,t)}{|r_q|} \cdot \frac{N_q(t)}{N(t)} \cdot \frac{NP_q(t)}{NP_q} \qquad (2)$$

The first factor $\frac{1}{N}\sum_{r_q \in S_q} \frac{freq\ (r_q,t)}{|r_q|}$ represents the term frequency component that is standard in Information Retrieval [26] and shows that the weight score $Pr_q(t)$ increases if term t occurs frequently in requirements related to regulation q. It is

computed as the average term frequency of term t in type q requirements r_q rescaled by the documents size $|r_q|$.

The remaining component of the expression in (1) measures inverse document frequency and penalizes the weight score if the term occurs in requirements related to other regulations. The second factor $N_q(t)/N(t)$ is the percentage of q type documents in S_q containing t with respect to all requirements in the training set containing t, whose number is denoted by $N(t)$. This factor decreases if the indicator term t is used broadly throughout the requirements specification. If the term is only used in q type requirements, it will evaluate to 1 for that type. The third factor $NP_q(t)/NP_q$ is the ratio between the number $NP_q(t)$ of projects containing type q documents with term t and the number NP_q of all projects in the training set with type q NFRs. The purpose of this rescaling factor with values ranging between zero and one is to decrease the weight $Pr_q(t)$ for terms that are project specific. It is equal to one only if a term appears in all the projects containing type q documents.

A probability score $Pr_q(t)$ is computed for each term t with respect to q, and terms are then ranked by decreasing order according to $Pr_q(t)$. Results from prior experiments with the NFR detector [10], and a series of initial experiments with the HIPAA security regulations showed that selecting the top 10 terms for each of the q types returned optimal classification results in comparison to other selection methods. This approach was therefore adopted for all of the reported experiments.

To classify requirements, a score $Pr_q(r)$ is computed that evaluates the probability that a certain requirement r is associated with the regulation q. This probability score depends on the lexical content of requirement r, and is based on the assumption that type q requirements are more likely to contain indicator terms for that type.

Let I_q be the set of indicator terms for regulation q identified during the training phase following expression (2). We assume that the weighted indicator terms in I_q are identified and their weights computed from a training set that contains correctly pre-categorized requirements. The indicator terms are mined using the expression in (2). The classification score that a requirement r belongs to regulation q is then defined as follows:

$$Pr_q(r) = \frac{\sum_{t \in r \cap I_q} Pr_q(t)}{\sum_{t \in I_q} Pr_q(t)} \qquad (3)$$

where the numerator is computed as the sum of the term weights of all type q indicator terms that are contained in r, and the denominator is the sum of the term weights for all type q indicator terms. The probabilistic classifier for a given type q will assign higher score $Pr_q(r)$ to a requirement r that contains several strong indicator terms for q.

3.2 Experimental Evaluation

To evaluate the effectiveness of the regulatory classifier, a leave-one-out cross validation experimental design was used for tracing HIPAA security rules against the patient healthcare systems. The manually created trace matrices were used as both the training and testing sets for this experiment. During each iteration of the experiment, nine datasets were used to train the classifier, while the remaining dataset was used for testing purposes. As a result, following ten iterations of the experiment each dataset was tested one time. In each iteration, indicator terms were extracted from the requirements in the training set, and were weighted using formula (2). The top ten scoring indicator terms for each regulatory code were selected and used to classify requirements in the test set using formula (3). A multiple classification scheme was used so that for any targeted HIPAA regulation, all requirements that scored higher than a certain threshold value were classified as retrieved for that regulation. This meant that a single requirement could be assigned to more than one HIPAA regulation. As with the previous experiment, threshold values were set individually for each query so as to maximize F_2-measure values.

3.3 Results

The results are reported in the confusion matrix of Table 4, which provides a useful visual means for analyzing classification results [14], and has the ability to depict true and false positives as well as true and false negatives. The true positives, are depicted on the diagonal, and have been highlighted in the diagram. The matrix shows, for example, that 48 of the 54 access control requirements were correctly classified, but that 5 of them were incorrectly classified as *audit control*, and 1 as *emergency access protocol* etc. As with the previous experiment, the F_2 measure and average precision are also reported.

Figure 2 depicts the average precision scores obtained using both the basic trace retrieval method and the machine learning method. Four of the regulatory codes showed very marked improvements in average precision when using the machine learning technique. For example the average precision for *access control* traces increased from 0.176 to 0.370, and for *audit control* from 0.144 to 0.647. Major improvements were also observed in *transmission encryption and decryption* and *personal authentication*. An examination of the requirements associated with each of these successful cases indicated that they had relatively strong themes that wove across the ten datasets, making the training process relatively straightforward. Furthermore, three of these cases had significantly larger than average training sets. *access control* had a total of 54 requirements across the 10 datasets, *audit control* had 86, and *personal authentication* had 42. The only HIPAA goal with a small training set which performed well under the machine learning approach was *transmission encryption and decryption* with a training set of only 5 requirements.

For the remaining HIPAA regulations, the machine learning technique did not improve average precision results in comparison to the basic trace retrieval results. However, in addition to having smaller training sets, five of these regulations, namely *automatic logoff*, *emergency access protocols*, *transmission security*, *storage encryption and decryption,* and *unique user ID*, had already all returned relatively strong average precision values for the basic traceability approach ranging from 0.264 to 0.933. These observations are discussed further in Section 5 of this paper. For the final regulatory code of *integrity*, neither the basic trace method nor the machine learning method were very effective. An analysis of the HIPAA integrity regulation suggests that the code itself is rather broadly-defined stating only that the product should "*Implement policies and procedures to protect electronic health information from improper alteration or destruction.*" A total of 17 requirements from the ten datasets were traced to this integrity code; however an analysis of these requirements showed that they covered a relatively broad gamut of topics such as a very high level requirement that "*the system must warn the user if format, consistency, or validity are in doubt*" and another one that stated

Table 4: Confusion matrix showing results from basic machine learning trace method.

	Total	AC	AUD	AL	TED	EAP	SED	I	PA	TS	UUI	Recall	Precision	F_2 Meas.	Avg. Prec
Access Control (AC)	54	48	5	0	0	1	0	4	29	8	6	0.889	0.121	0.285	0.370
Audit Control (AUD)	86	33	81	3	0	3	0	8	29	4	5	0.942	0.305	0.555	0.647
Automatic logoff (AL)	10	5	0	7	0	0	0	1	4	0	0	0.700	0.350	0.525	0.373
Encryption (TED)	5	0	0	0	5	2	3	1	0	2	0	1.000	0.417	0.682	0.596
Emerg. access proc. (EAP)	4	4	0	0	0	4	0	0	2	0	1	1.000	0.098	0.246	0.354
Encryption & decrypt (SED)	7	0	0	0	5	1	7	1	1	2	0	1.000	0.583	0.807	0.530
Integrity (I)	17	1	6	0	0	0	0	3	6	0	4	0.176	0.011	0.029	0.002
Personal authentication (PA)	42	15	5	0	0	1	0	3	39	1	11	0.929	0.101	0.249	0.550
Transmission security (TS)	7	3	1	0	1	1	1	2	0	4	0	0.571	0.063	0.155	0.037
Unique user ID (UUI)	11	3	2	0	0	1	0	0	7	1	8	0.727	0.056	0.146	0.175
Non-HIPAA		285	166	10	1	27	1	259	269	41	108				
Total retrieved		397	266	20	12	41	12	282	386	63	143				

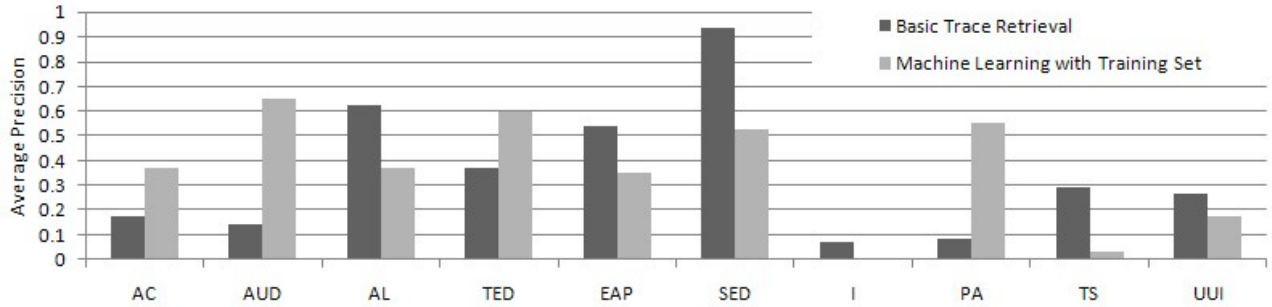


Figure 2. Average precision scores obtained using Machine Learning method versus Basic Trace Retrieval.

that "*the system shall… retain all original documentation.*" The breadth of integrity requirements suggested that the HIPAA regulation itself was perhaps at too high a level to be concretely traced to requirements.

In summary, the machine learning approach did not improve traceability for the five HIPAA goals that performed best under the basic traceability approach; however it did significantly improve results for four of the five HIPAA goals that had not previously performed well. The machine learning approach might therefore be used to improve traceability of regulatory codes that do not tend to trace easily into requirements documents.

However this approach is only feasible when a training set is available. For purposes of these experiments, we spent approximately 125 person hours developing the training set for the HIPAA regulations, even though it involved only 11 individual requirements. The vast majority of this time was spent finding and processing the requirements for 10 healthcare related products, and constructing the associated trace matrices. This effort could be minimized if the training set were built as a natural by-product of developing and tracing software systems in compliance to a given set of regulatory codes. It could also be distributed across multiple organizations if a specific user community worked collaboratively to build the training set. Furthermore, one could envision regulatory bodies publishing regulations, augmented with terms to support automated trace retrieval processes. Unfortunately without this type of support, the sheer number and breadth of regulatory codes suggests that it will not always be possible or cost-effective to construct training sets using the manual techniques described in this paper. The following section of this paper therefore proposes a novel and alternate approach for automatically discovering indicator terms through mining domain-specific documents.

4. WEB MINING APPROACH

The proposed approach is based on the idea that when a training set is not available, a relevant set of indicator terms can be learned from domain specific documents mined from the Internet. As the experiments reported in this section will demonstrate, this approach generally improved traces for the same set of HIPAA regulations that were improved by the machine learning approach. The benefit of the web-mining approach is that it bypasses the time-consuming step of manually constructing a training set. The approach involves three steps. First a set of relevant domain specific documents are identified. Second, the documents are analyzed to extract a set of domain specific terms. Finally these terms are composed into a new query which is used to execute the trace. These steps are now described in more detail.

4.1 Retrieving domain specific documents

Domain specific documents are identified for each regulatory code through extracting one or more representative terms or phrases and using them to issue a web-based search query. For the HIPAA regulations, the initial query was formulated using the title provided for each regulation. In cases for which the search results were deemed unsatisfactory, the user manually refined the query by adding or removing additional terms and phrases. Following an initial series of experiments, we established the rule of only rejecting a retrieved document if it contained small amounts of text i.e. less than 200 words in a single block, because short websites were found to frequently represent product advertisements containing potentially limited and biased terminology. Websites that were judged by the researcher to be

entirely non-relevant were also rejected. In the future, the process of document selection will be automated and standard disambiguation techniques will be used to help reject non-relevant documents.

4.2 Term Mining

The retrieved documents were processed using an algorithm for extracting domain concepts from natural text [30]. Q-TAG, a part-of-Speech tagger, was used to parse the documents and identify single nouns and two-term noun phrases. The following metrics were then computed for each of these terms and term-phrases:

Domain term frequency (DTF) computes term frequency information for term *t* across multiple documents as follows:

$$DTF(t) = \left[\sum_{t \in D} freq(t, D)/|D|\right]$$

where $\sum_{t \in D} freq(t, D)$ is the total number of occurrences of term *t* across the domain-specific document collection *D,* and |D| is the total number of documents.

Domain specificity (DS) measures the extent to which a term or term phrase is specific to the domain document, as opposed to occurring frequently across a broad spectrum of topics. For our experiments, domain specificity *DS(t)* of term *t*, was estimated by comparing the relative frequency of the term within a domain-specific document, versus its relative frequency in a general corpus of documents [3]. It is calculated as follows:

$$DS(t) = ln\left[\frac{freq(t, D)}{\sum_{t \in D} freq(t, D)} \Big/ \frac{freq(t, G)}{\sum_{t \in G} freq(t, G)}\right]$$

where the first component $freq(t, D)/\sum_{t \in D} freq(t, D)$ is the normalized number of occurrences of term *t* in the domain-specific document collection *D*, and the second component is the normalized number of occurrences of t in the general corpus of documents.

Concept generality (CG) computes the fraction of domain specific documents in which a specific term occurs. Concept generality differentiates between terms that occur in multiple domain specific documents versus those that occur in only a few. The concept generality of term *t*, *CG(t),* is computed *as* the number of documents containing term *t,* $|D_t|$ over the total number of documents: $CG(t) = |D_t|/|D|$

Following a series of informal exploratory experiments the following heuristics were established for evaluating the set of generated terms. Terms were automatically rejected if CG(t) < 0.3 or the maximum DS(t) value was less than 5.

4.3 Trace Execution

In the final stage of the algorithm, the terms identified through domain concept mining are concatenated into a query which is issued using the standard traceability algorithm described in Section 2.1

4.4 Experimental Evaluation

To evaluate the web mining technique an experiment was conducted for tracing the HIPAA security regulations against requirements in the ten healthcare datasets. Initial queries for selecting domain specific documents were formulated by extracting the headings from each of the HIPAA regulations. A researcher then evaluated the top results returned by BING, GOOGLE, and YAHOO, and as a result modified the search queries to produce the final queries shown in Table 5. Ten documents were selected for each HIPAA regulation. For example, the documents selected for *Access Control* included a Wikipedia Article, Apache HTTP Server documentation, "principles surrounding access rights in the context of the Seventh Framework Programme of the European Community", Access Control Standard by State of Maryland Department of Information Technology and Access Control Standards by Purdue University. Each document was analyzed to extract domain concepts using the methods described in section 4.2, and the ten top terms were identified for each regulation.

Table 5. Queries used to select domain documents.

HIPAA regulation	Final query
Access Control (AC)	access control
Audit Control (AUD)	audit controls
Automatic logoff (AL)	automatic logoff, auto logout, terminate electronic session
Emergency Access (EAP)	emergency access procedure
Integrity (I)	integrity, data integrity
Personal authentication (PA)	person entity authentication, person entity verification
Storage Encryption (SED)	encryption decryption
Transmission Encrypt (TED)	encryption
Transmission security (TS)	transmission security data
Unique user ID (UUI)	unique user id, unique user identification

Table 6. Indicator terms mined for Access Control.

Term	Concept Generality	Sum of normalized term frequencies	Average Specificity
access	1.0	0.50	14.13
control	0.9	0.24	6.85
role	0.5	0.13	63.11
resourc	0.7	0.12	20.80
password	0.5	0.12	139.24
secur	0.9	0.10	14.01
standard	0.8	0.09	7.15
right	0.6	0.09	51.26
privileg	0.7	0.07	35.21
authent	0.5	0.07	32.14

For illustrative purposes, the indicator terms mined for the Access Control HIPAA regulation are shown in stemmed form in Table 6. Several terms, such as *access* and *control,* also occurred in the original regulation, however additional terms such as *role, right* and *privilege* were discovered, and several relatively non-useful terms such as *persons* and *software* that occurred in the original regulation were omitted from the reconstituted query. It is also interesting to note that several terms such as *right,* and *privilege* were discovered by both the standard machine learning method and the term mining approach. A new trace query was formulated by concatenating the identified terms and phrases. For example, the *access control* query was transformed from its original form shown in Figure 1 to *"access control role resource password secur standard right privilege authent"* and executed using the probabilistic formula (1) described in Section 2.1 of this paper.

Results from the experiment are reported in Table 7, and compared against the other two techniques in Figure 3. Four HIPAA regulations, namely *Encryption*, *Emergency Access Procedure*, *Encryption and Decryption*, and *Transmission Security* were traced with 100% recall. These were the same four that had been recalled at 100% by the original basic algorithm.

Table 7. Confusion matrix showing results from query reconstitution trace method.

	Total	AC	AUD	AL	TED	EAP	SED	I	PA	TS	UUI	Recall	Precision	F_2 Meas.	Avg. Prec
Access Control (AC)	54	47	13	0	13	7	3	29	14	8	27	0.870	0.170	0.367	0.236
Audit Control (AUD)	86	16	74	0	8	5	0	18	15	3	19	0.860	0.301	0.531	0.290
Automatic logoff (AL)	10	2	0	4	0	0	0	0	1	0	4	0.400	1.000	0.500	0.405
Encryption (TED)	5	1	0	0	5	0	1	2	2	2	0	1.000	0.034	0.096	0.147
Emerg. access proc. (EAP)	4	2	0	0	0	4	0	2	2	0	1	1.000	0.052	0.141	0.120
Encryption & decrypt (SED)	7	2	1	0	5	0	7	3	1	6	2	1.000	0.259	0.512	0.455
Integrity (I)	17	2	1	0	4	0	2	11	2	1	1	0.647	0.018	0.051	0.132
Personal authentication (PA)	42	31	7	0	4	2	10	6	39	7	23	0.929	0.188	0.401	0.291
Transmission security (TS)	7	4	4	0	5	1	0	5	0	7	3	1.000	0.079	0.205	0.239
Unique user ID (UUI)	11	2	3	0	2	1	0	4	3	1	7	0.636	0.033	0.090	0.058
Non-HIPAA		168	143	0	99	57	4	545	129	54	125				
Total retrieved		312	249	277	532	324	120	490	271	114	429				

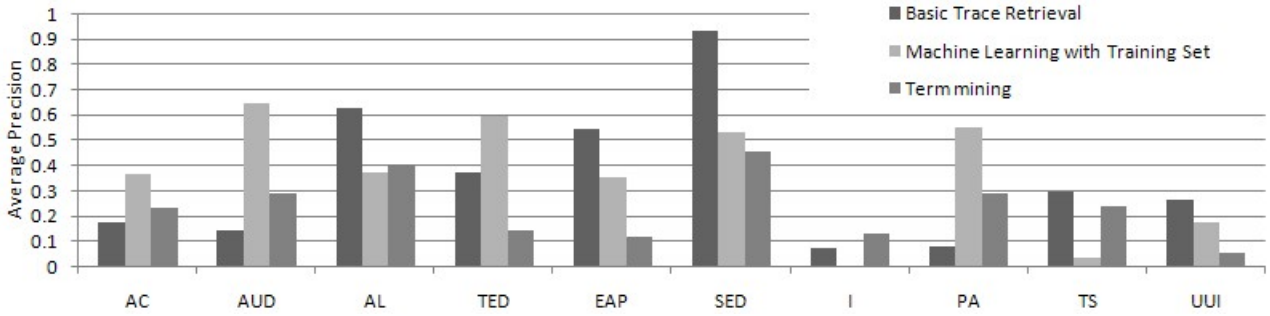


Figure 3. Average precision of three different traceability methods by query type.

Additionally the three regulations of *access control*, *audit controls* and *person or entity authentication* returned recall values of over 0.85. Although *integrity*, still did not perform well, with a recall of only 0.647; this was a significant improvement over both of the previous methods where maximum recall was only 0.529 for the basic approach and 0.176 for the machine learning method. The trace for *unique user ID* at recall of 0.63 performed worse than for both the previous methods. However this particular problem highlighted one of the risks of this non-deterministic tracing approach which is its dependence upon finding a suitable and expressive set of documents. In this case, the selection of domain documents failed to produce the term '*unique'* which occurred quite frequently across *UUI* requirements. Future experiments, in which the process of selecting domain specific documents is fully automated, and therefore can include a more extensive analysis, may be able to overcome this problem more satisfactorily.

The web mining approach outperformed the basic approach for three of the same HIPAA regulations that were improved by the machine learning method; these were *access control*, *audit control* and *personal authentication*. Although improvements in average precision were less than those achieved by the machine learning method, the human effort was also significantly less. The web mining approach also improved results for *integrity* which the machine learning approach had been unable to do. However the web-mining approach performed worse on four of the HIPAA regulations than either of the other methods. These were *transmission encryption and decryption, emergency access procedures, storage encryption, and unique user ID*. There were no cases in which the web-mining technique outperformed both of the other two methods.

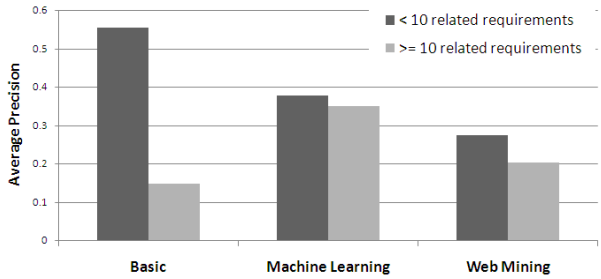
5. ANALYSIS OF THE RESULTS

This paper has described three methods for tracing regulatory codes to product level requirements, and has illustrated and evaluated them with respect to the HIPAA security rule. The results suggest that machine learning methods can be used to improve trace query results when sufficiently large sized training sets are available. A Pearson correlation analysis of the number of requirements associated with each HIPAA regulation (see Table 1) and the improvement in average precision achieved by utilizing the machine learning method instead of the basic trace retrieval approach, returned a coefficient of 0.783, indicating a strong positive correlation between these two factors. Surprisingly, a similar correlation was observed for the Web Mining approach. Further analysis suggests that in cases where there are a larger number of requirements fanning-out from one regulatory code, the terms in the raw regulatory code may provide insufficient query terms to adequately retrieve the diverse set of related requirements. In this case, query augmentation is essential. Our experiments show that in these cases, although the machine learning method was most effective, the web mining technique also improved results, but without the excessive effort required to manually construct a training set. These results are summarized in Figure 4

6. GENERALITY OF THE APPROACH

Establishing the training sets and the infrastructure to conduct these experiments was very time consuming, and so it was not feasible to repeat the entire study for a different set of regulatory codes within the scope of this paper. Nevertheless, a small and informal study was conducted to evaluate the generality of the second technique, in which indicator terms are mined from the web and used in the tracing process.

Figure 4. Mean Average Precision for HIPAA regulations categorized by number of related requirements in dataset of 10 Healthcare requirements specifications.



In the first informal experiment, a single regulation was traced from the American Railway Engineering and Maintenance of Way (AREMA) codes to 165 contractual requirements for a large systems engineering project. The selected code stated that "*When used, combustion burners should be provided with suitable safety controls including a combustion safety control (e.g. flame detector), high temperature limit safety control, and other safety controls that may be required to provide for safe operation of the device*." This particular code was selected for the experiment because it had the most individual traces (5) in the provided answer set. Applying the search-augmented trace method led to an improvement in precision from 22% to 27% at consistent recall levels of 100%. In a second informal experiment, a single regulation "*The total volume of gas required shall (a) be determined as the total volume for all appliances supplied…"* was traced from the Canadian Natural Gas and Propane Code (CNGPC) to 84 component level requirements; however for this particular trace, precision was quite poor with and without the search-augmented technique. The single target link, which originally appeared in the 17th ranked position moved only to the 14th position with the enhancement. However the search-enhanced method was able to learn the previously undetected phrase of *gas supply* (i.e. supply of gas) which was found in the targeted requirement. Although not incorporated into our current tracing tool, phrasing is known to improve trace results [31] and could therefore potentially be used to improve future traces.

An analysis of the AREMA and CNGPC codes suggests that future enhancements are feasible if the algorithm is fine-tuned to support a more focused search. Whereas, regulatory codes such as HIPAA tend to include relatively distinct topics, the CNGPC, and to some extent the AREMA codes, are much less distinct and individual codes contain overlapping ideas. Future work is clearly needed to fully understand the constraints of this approach and to fine-tune techniques for retrieving more focused domain relevant documents or sections of documents.

7. THREATS TO VALIDITY

There are three primary threats to validity in this work. The first relates to the generality of the healthcare related datasets. This threat was largely addressed through using 10 different requirements specifications taken from entirely different sources. In all cases, the requirements text was extracted directly from source documents, and only very common words were added during the specification process that would either be ignored as stopwords or assigned insignificant weightings by the tf-idf algorithm. The second threat relates to the fact that one of the datasets was much larger than the others which could have introduced terminology bias. Although not reported in detail in this paper, an informal analysis showed that the traces worked effectively across all of the different projects. Future work will examine this in greater detail.

The third major threat to validity relates to biases that could have been introduced in selecting domain specific documents for experimenting with the search-based approach to learning indicator terms. This threat was partially addressed through defining a set of selection rules. Nevertheless these rules were based on our observations of security related documents and may be biased towards HIPAA regulations or towards security related documents in general. As a result of this issue, we cannot claim generality of the document selection process for other types of regulations or for the tracing problem in general. As explained in section 5, this will be left primarily to future work.

8. RELATED WORK

As referenced in the introductory section of this paper, numerous researchers have investigated the use of semi-automatic and automatic approaches to dynamically generate traceability links [28] based on information retrieval (IR) methods including Latent Semantic Analysis (LSA), the Vector Space Model, and probabilistic approaches [2, 7, 9, 11, 12, 19, 23]. Extensive empirical studies have shown that in order to recall at least 90% of the correct links, precision levels of 10% to 50% can be achieved in most datasets[18]. Despite these precision problems, IR methods have been shown to save significant time and effort when used in place of brute force tracing methods. Associated tools such as RETRO [20], Poirot [7], and ADAMS[11] are currently being used on a variety of government and industrial projects. Other semi-automated tracing techniques include rule-based approaches[29], scenario and test case-based methods [13], event-based approaches[8], and policy-based methods [24]. However much of this related work has focused on generating traces between documentation and code [2, 23] or across artifacts such as requirements, design, code, and test cases within a project [7, 9, 12, 19, 20]. Furthermore, techniques such as scenario or policy-based approaches are more appropriate for tracing within a project than for tracing external documents such as regulatory codes.

With respect to automated learning of indicator terms, Hu et al, used Wikipedia to augment users' web-based queries [17], while several other researchers have used more general web knowledge to augment queries [5, 27]; however their work assumes a typical web-based query of only 2-3 words. Their intent is therefore to augment the query with additional and potentially disambiguating information. In contrast, regulatory codes are much longer in nature, typically involving from 10-20 terms, and so our approach is designed to augment a standard regulatory code with alternate terms that might be found in the search space, while removing non-critical peripheral terms that cause precision problems.

9. CONCLUSIONS AND FUTURE WORK

The techniques described in this paper offer a promising new approach for supporting the task of tracing from regulatory codes such as the HIPAA security rule to contractual and product level requirements. Experimental results have shown significant improvements over existing trace-retrieval methods, suggesting that the reported techniques could significantly reduce the human effort and increase the accuracy of tracing regulatory codes. We are currently engaged in an industry pilot study [4] and plan to repeat the traces for this study using the term mining technique

described in this paper. Other areas for future work include tracing requirements to lower level refinements of the HIPAA regulations. In prior work, Breaux and Anton [5] identified six types of data access constraints, to handle complex cross-references, resolve ambiguities, and assign priorities between access rights and obligations. As part of our future work we will therefore apply the data mining techniques described in this paper to a more fine-grained model of the HIPAA regulatory codes showing specific rights and obligations with respect to specific patient healthcare information. This approach is expected to produce a more precise traceability report that highlights regulatory risks and failures with respect to specific health care information.

10. ACKNOWLEDGMENTS

This work is funded in part by the National Science Foundation under grant #CCF-0810924. We also thank Travis Breaux and Arlene Yetnikoff for their invaluable input.

11. REFERENCES

1. Health Insurance Portability and Accountability Act of 1996 *HIPAA*, 1996.
2. Antoniol, G., Canfora, G., Casazza, G. and De Lucia, A., Information Retrieval Models for Recovering Traceability Links between Code and Documentation. in *IEEE Intn'l Conf on Software Maintenance*, (San Jose, CA, 2000), 40-51.
3. Bennett, K.H., Rajlich, V. and 73-87, I.-F.o.S.T., Software maintenance and evolution: a roadmap. in *International Conference on Software Engineering - The Future of Software Engineering Track*, (2000), 73-87.
4. Berenbach, B., Gruseman, D., and Cleland-Huang, J., "Application of Just In Time Tracing to Regulatory Codes", Systems Engineering Research, Hoboken, NJ, March, 2010.
5. Breaux, T.D. and Anton, A.I. Analyzing Regulatory Rules for Privacy and Security Requirements *IEEE Transactions on Software Engineering*, 2008, 5-20.
6. Broder, A., Fontoura, M., Gabrilovich, E., Joshi, A., Josifovski, V. and Zhang, T. Robust Classification of Rare Queries using Web Knowledge. *30th Intn'l ACM SIGIR Conf on Research and Development in Inf. Retrieval*, July, 2007
7. Cleland-Huang, J., Berenbach, B., Clark, S., Settimi, R. and Romanova, E. Best Practices of Automated Traceability. *IEEE Computer*, *40* (6). 27-35
8. Cleland-Huang, J., Chang, C.K. and Christensen, M. Event-Based Traceability for Managing Evolutionary Change. *IEEE Trans.s on Software Engineering*, *29* (9). 796-810.
9. Cleland-Huang, J., Settimi, R., Duan, C. and Zou, X. Utilizing Supporting Evidence to Improve Dynamic Requirements Traceability *International Requirements Eng. Conf.*, IEEE, Paris, France, 2005, 135-144.
10. Cleland-Huang, J., Settimi, R., Zou, X. and P., S. Automated Detection and Classification of Quality Requirements. *Reqs. Eng. Jrnl, Springer Verlag*, *12* (2), 103-220.
11. DeLucia, A., Fasano, F., Oliveto, R. and Tortora, G. Enhancing an Artefact Management System with Traceability Recovery Features. *Proc. of the 20th Intn'l Conf on Software Maintenance*, *Chicago, IL* (Sept). 306-315.
12. Duan, C. and Cleland-Huang, J. Clustering Support for Automated Tracing *Conference on Automated Software Engineering*, IEEE, Atlanta, GA, 2007, 244-253.
13. Egyed, A. Scenario-Driven Approach to Trace Dependency Analysis. *IEEE Trans. on Software Eng.*, *29* (2) 116-132.
14. Fawcett, T. ROC Graphs: Notes and Practical Considerations for Researchers *HP Labs Technical Report*, 2003.
15. Gotel, O. and Finkelstein, A. An Analysis of the Requirements Traceability Problem, *Intn'l Conf on Requirements Eng.*, Colorado Springs, CO, USA, 1994.
16. Gotel, O. and Finkelstein, A., Extended Requirements Traceability: Results of an Industrial Case Study. *Intn'l Symposium on Requirements Engineering*, (1997), 169-178.
17. Hu, J., Wang, G., Lochovsky, F., Sun, J. and Chen, Z. Understanding user's query intent with wikipedia. *18th International Conference on World Wide Web*, *Madrid, Spain, April 20-24, 2009* (WWW'09). 471-480.
18. Huffman Hayes, J. and Dekhtyar, A. A Framework for Comparing Requirements Tracing Experiments. *International Journal of Software Engineering and Knowledge Engineering*, *15* (5). 751-782.
19. Huffman Hayes, J., Dekhtyar, A. and Karthikeyan, S. Advancing Candidate Link Generation for Requirements Tracing: The Study of Methods. *IEEE Transactions on Software Engineering*, *32* (1). 4-19.
20. Huffman Hayes, J., Dekhtyar, A., Sundaram, S. and Howard, S. Helping Analysts Trace Requirements: An Objective Look *Reqs. Eng. Conference*, Kyoto, Japan, 2004, 249-259.
21. Jalaji, A., Goff, R., Jackson, M., Jones, N. and Menzies, T. Making Sense of Text: Identifying Non Functional Requirements Early. *W.Virginia Univ. CSEE Tech.report.*
22. Maletic, J.I. and Marcus, A., Using Latent Semantic Analysis to Identify Similarities in Source Code to Support Program Understanding. in *12th IEEE Intn'l Conf on Tools with Artificial Intelligence*, Vancouver, BC, 2000, 46-53.
23. Marcus, A. and Maletic, J.I., Recovering Documentation-to-Source-Code Traceability Links using Latent Semantic Indexing. in *25th IEEE/ACM Intn'l Conf on Software Engineering (ICSE'03)*, (Portland, OR, 2003), 125-137.
24. Murta, L.G.P., Andre, v.d.H. and Werner, C.M.L. ArchTrace: Policy-Based Support for Managing Evolving Architecture-to-Implementation Traceability Links. *21st IEEE Intn'l Conf on Automated Software Eng.*, 135-144.
25. Ramesh, B. and Jarke, M. Towards Reference Models for Requirements Traceability. *IEEE Trans. on Software Engineering*, *27* (1). 58-93.
26. Salton, G. *Automatic Text Processing: The Transformation, Analysis and Retrieval of Information by Computer*. Addison-Wesley, 1989.
27. Shen, D., Sun, J., Yang, Q. and Chen, Z. Building bridges for Web Query Classification. *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, *SIGIR-06* (2006).
28. Spanoudakis, G. and Zisman, A. Software Traceability: A Roadmap. *Handbook of Software Eng. and Knowledge Eng., S.K. Chang, Ed.*, *World Scientific Publishing Co.* 395-428.
29. Spanoudakis, G., Zisman, A., Perez-Minana, E. and Krause, P. Rule-based generation of requirements traceability relations. *The Jrnl of Systems and Software*, *72* (2004). 105-127.
30. Zou, X. Evaluating the Use of Project Glossaries in Automated Trace Retreival *Software Engineering Research and Practice*, CSREA Press 2008, Las Vegas, USA, 2008.
31. Zou, X., Settimi, R. and Cleland-Huang, J. Improving Automated Requirements Trace Retrieval: A Study of Term-based Enhancement Methods. Empirical Software Engineering, Online First.