Mining Associations between Quality Concerns and Functional Requirements

Xiaoli Lian
Beihang University
Beijing, China
lianxiaoli@buaa.edu.cn

Jane Cleland-Huang
University of Notre Dame
Notre Dame IN, USA
JaneClelandHuang@nd.edu

Li Zhang
Beihang University
Beijing, China
lily@buaa.edu.cn

***Abstract*—The cost and effort of developing software systems in a new technical area can be extensive. An organization must perform a domain analysis to discover competing products, analyze their architectures and features, and ultimately discover and specify product requirements. However, delivering high quality products, depends not only on gaining an understanding of functional requirements, but also of qualities such as performance, reliability, security, and usability. Discovering such concerns early in the requirements process drives architectural design decisions. This paper extends our prior work on mining functional requirements from large collections of domain documents, by proposing and evaluating a new technique for discovering and specifying quality concerns related to specific functional components. We evaluate our approach against three domains of Positive Train Control, Electronic Health Records, and Medical Infusion Pumps, and show that it significantly outperforms a basic information retrieval approach. Finally we classified the forms of retrieved information, discussed the utility of different types, and conducted a small study with an experienced engineer to investigate the quality of requirements produced using our approach.**

I. Introduction

When entering a new domain, an organization must invest significant time and effort exploring competitors products, analyzing publicly available product descriptions and reference architectures, and ultimately discovering and specifying requirements [15]. This task can either be performed manually by reviewing competitor's websites and brochures, or can utilize data mining and machine learning techniques to search through large collections of online materials in order to organize, cluster, and collate product descriptions in meaningful ways. Our prior work in this area [21] described an approach for mining requirements knowledge from large repositories of domain documents collected using standard web-searching techniques, and then extracting and ranking candidate requirements appearing in the documents. We showed that it was possible to discover meaningful requirements-related information as a basis for specifying functional requirements.

However, in most domains, product requirements are not only characterized by functional descriptions, but also by specific quality concerns. For example, products such as eBay or Amazon are driven by scalability, performance, usability, and security concerns [1], while domains such as avionics and medical devices are driven by dependability, security, response time, and safety concerns [4]. However, quality requirements cannot be cleanly separated from functional concerns as they exhibit dependencies on each other [26, 10]. In fact, quality concerns are often underspecified functional requirements [27, 5, 17, 16] and are interwoven throughout the functional components of a system [3, 23, 20].

Domain analysis techniques which rely upon mining requirements knowledge, could therefore benefit by focusing not only on extracting functional requirements, but also on discoving related quality concerns. In this paper we present our approach *Holistic Requirements Mining* (HRM) which searches domain documents with the aim of retrieving and organizing textual information describing interdependencies between quality concerns and functional components. The task is quite challenging given the diversity of domain documents – which include project proposals, requirement specifications, architectural designs, implementation plans, product brochures, domain surveys, presentation material, regulatory codes, and other associated literature. In addition to presenting HRM we use grounded theory to manually classify the types of information extracted by our approach and to evaluate the utility of different types of information. We address three specific research questions:

RQ1: To what extent can interdependencies between functional components and quality concerns be automatically extracted from a collection of domain documents?

RQ2: What kinds of knowledge can be gained from the interdependency descriptions between functional components and quality concerns extracted from domain documents?

RQ3: Can a software engineer without domain knowledge construct quality-related requirements for a functional component using HRM?

The first two research questions are evaluated against three different domains of Positive Train Control (PTC), Electronic Health Records (EHR), and Medical Infusion Pumps (MIP), while the last is evaluated against PTC only.

The remainder of the paper is laid out as follows. Section II provides an overview of our approach while Sections III and IV describe the technique for mining and classifying quality concerns and its evaluation. Section V describes the categories of information discovered by HRM, while Section VI describes a small case study conducted with an experienced engineer to evaluate HRM's support for specifying quality-

2332-6441/17 $31.00 © 2017 IEEE
DOI 10.1109/RE.2017.68

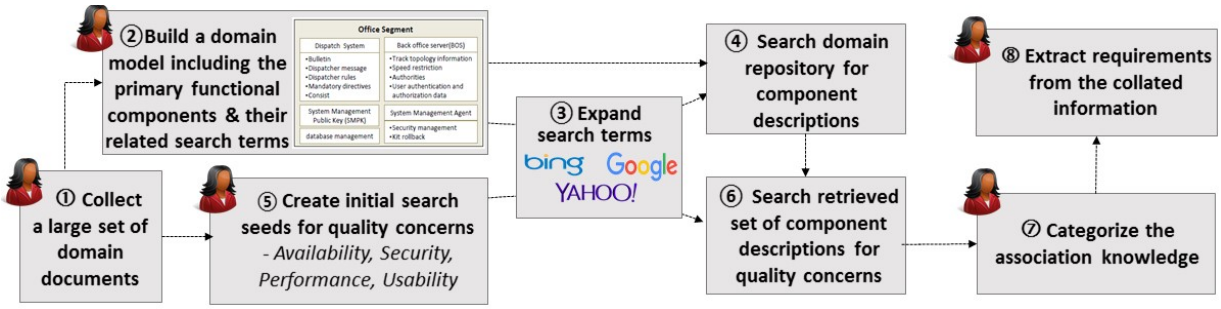

Fig. 1: An overview of the HRM process for mining quality related requirements information from large collections of domain documents and mapping it to functional components.

related requirements. Sections VII to IX describe related work, threats to validity, and conclusions.

II. Overview of Approach

As shown in Figure 1, HRM consists of eight steps. Engineers first collect a large and diverse set of domain documents (Step 1) from which they then construct an initial domain model which describes major functional components of the domain (Step 2). In our prior work [21] we compared the quality of a domain model produced manually by engineers against models generated using automated topic modeling techniques. Our experiments showed that the human constructed models were far superior to the generated ones in terms of their functional decomposition. Further, constructing a functional model from the domain documents took approximately 25 hours, which we, and our industrial collaborators, deemed acceptable given its superiority. The initial domain models presented in this paper, are therefore all manually constructed through an iterative process of reviewing the domain documents. Further, during the process of constructing a domain model, the HRM approach requires the engineer to specify an initial set of representative terms for each of the functional components. In Step 3, HRM performs a query expansion utilizing a technique previously used to expand HIPAA (USA's Health Care Information Portability and Accountability Act) regulations in order to identify HIPAA related security safeguards in sets of requirements [18]. Finally, in Step 4, the engineer searches the domain documents for information related to each of these components. Steps 1-4 are a replication of our prior work [21].

In Step 5, the engineer identifies quality concerns of interest. Although HRM is designed to work with a broad set of quality concerns, we focus upon *Availability*, *Security*, *Performance* and *Usability* due to their importance to software architectural design according to a survey conducted by Ameller et al. [2]. As with the initial functional search terms, each quality concern is assigned some initial search seeds, which are then expanded (in Step 3). In Step 6, the expanded terms are used to search through the previously retrieved component-related text in order to identify quality concerns. In Step 7, the information is categorized according to its practical significance to the system requirements and then presented to the user in Step 8 so that they can use it to assess quality concerns and to specify them as requirements. The quality concern extraction (Steps 3, and 5-8) could be performed in conjunction with mining functional requirements (i.e. Steps 1-4), or could be applied against an existing set of requirements for a given domain. HRM is designed to support analysis by enabling an engineer to gain deeper understanding of existing system requirements, improve an existing set of system requirements, or simultaneously mine functional requirements and quality concerns from a domain repository.

A. Case Domains

We describe and evaluate HRM within the context of three systems engineering domains, namely: *Positive Train Control System (PTC)*, *Medical Infusion Pump (MIP)* and *Electronic Health Record (EHR)*. Each domain repository includes diverse documents including standards manuals, technical reports, practice experience reports, product booklets, system design and system development plan represented in .pdf and Microsoft Word format.

• Positive Train Control (PTC)

The PTC domain repository was constructed by our industrial collaborators. It included 365 files consisting of approximately 520MB of data. Files in the repository included a mixture of powerpoint, pdf, excel, html, and word. In addition, we were provided with 480 requirements for the Onboard Unit (OBU) subsystem, which is a critical PTC component for determining the train's current location, authorized limits and so on in PTC.

The domain model constructed following our previously described practices [21], included 27 components such as the *track warrant control*, *wayside status relay*, *onboard unit*, and the *dispatch system*. The model was verified by our industrial collaborators.

• Electronic Health Records (EHR)

We constructed the EHR repository ourselves using search terms such as *Electronic Health Records*, *patient record systems*, and *health care management systems*. The repository consisted of 100 files constituting 79.2MB of data. For cross-validation purposes, in order to evaluate our domain model and subsequent components, we utilized 1,182 requirements previously downloaded from the Certification Commission for Healthcare Information Technology (CCHIT) website. The

EHR domain model was constructed by one of the researchers on the team and independently verified by another (who has several years of industrial experience in the EHR domain).

• **Medical Infusion Pump (MIP)**
We also constructed the MIP repository ourselves for purposes of this work. Search terms included *Medical Infusion*, *infusion pump* and *smart pump*. The MIP repository included 100 files constituting 66.7MB of data. In addition, we utilized a requirements specification for the Medical Infusion Pump containing 126 requirements to further evaluate and improve the domain model.

III. Mining Functional and Quality Concerns

The HRM process involves searching repository documents for statements relevant to each component using a set of augmented search terms and then performing a second step to determine if any statement describes a quality concern. The identification of interdependencies is therefore a two-phase process. As previously explained we assume a starting point of a domain model in which each of the major functional components has been identified and represented by an initial set of *seed terms* used to perform a query-expansion. A selection of components in the three domain models is shown in Table I.

A. Query Expansion and Search

The initial set of search terms for each component is extracted from the functional sub-component descriptions in the manually constructed domain models. Descriptions of each component provide an initial overview of its functions, and were obtained through a manual survey of domain documents as described in our prior work [21]. In HRM, we further expand the descriptors for each query as follows:

1) Use existing descriptors to formulate a query, and search through the entire set of domain documents for sections of text (i.e. paragraphs, bullet points etc) containing at least one of the inital search terms.
2) Extract each section, and process the text to remove commonly occurring 'stop' words such as 'this' and 'that'.
3) Perform a part-of-speech analysis on the relevant text sections using QTag to identify nouns and noun phrases.
4) Compute the frequency at which each noun and noun phrase occurs across the entire set of relevant text sections.
5) Sort the nouns and noun phrases by their frequency, select the top ten, and add them to the list of search terms for the component.

As an illustrative example, the initial search terms for the *cluster controller* component were *ground network control*, *base station management*, and *turn-around*, and these were expanded to include *cluster controller*, *emergency broadcast*, *base station*, *corrective action*, *attached printer*, *alarm information*, *remote entity*, *alarm reporting*, *attached base station*, and *communications line*.

TABLE I: Examples of expanded Search Terms.

Positive Train Control (Selection Only)	
Components	**Representative Search Terms**
Wireless Net.	registration table, wifi, wireless gateway
Edge Message Protocol	edge message protocol, emp, class c, crc, message envelope, data integrity, delivery ack,...
Cluster Control	cluster control, cc, ground network control, base station management
Mobile	mobile communication package, frequency controller, mcp
Backbone Comm. Net.	backbone communication network, on-board ptc data communication package, wayside ptc data communication package
Brake	dynamic brake monitor, dbm to on-board computer,tractive braking effort,...
Event Recorder	crashworthy memory module,data preservation, data recovery, ...
Onboard Unit	locomotive control, lcon, cpu, braking algorithm, wheel tachometer, speed enforcement,...
Human Machine Interface	human machine interface, crew acknowledge, navigation, bulletin cancellation, establish track location, authority change request,...

Electronic Health Records	
Components	**Representative Search Terms**
Clinical Decision Support	cdss,clinical reminder, mental health survey, trigger warnings, diagnostic decision support, clinician work list,...
Administrative System	administrative system, registration, admissions, discharge, transfer, medical record number, master patient index,chief complaint,...
Electronic Communication	order communication, picture archiving and communications system,clinical messaging, e-mail,communication hub,remote access,asp,...
Report. & Pop. Health Mngmt.	public health,reportable disease,syndromic surveillance,disease registries,clinical dashboard,external accountability report,...
Clinical Documentation	electronic document, medication list, living wills, physician notes, wrist band,...
Comp. Provider Order Entry	electronic prescribing, prescription routing, patient drug allergies,drug-drug interaction,...

Medical Infusion Pump	
Components	**Representative Search Terms**
Drug Library	dose error reduction system, disease acuity,icu drugs,patient weight,programmable limits,...
Communication	wireless technology,wpa2,emc, bluetooth, zigbee, wmts, wi-fi, remote monitoring, encryption,...
Barcode System	barcode scanning, barcode scanner, wristband,barcode verification,medical record number,mrn,id band,medication label,...
Human Factors Engineering	human performance,operational error,operator stress,training requirement,user fatigue,...
Pump Operation	programming, manual mode, graphical user interfaces,guis,flow rate,volume control,duration, volume to be infused,...
Security Features	alarm, alerts and warnings, air-in-line sensor, human error, verify drug and concentration,...

This expanded set of search terms was then used to search through all of the documents in the domain repository in order to retrieve all relevant documents and to extract relevant chunks of text (i.e. paragraphs, sentences, or bullet points). We retrieved any chunk of text which contained at least one of the

TABLE II: The search terms for each of the four target quality concerns, displayed in descending order of their weights

Quality	search terms
Availability	redund, failur point, manpow, downtim, self, replic, hazard analysi, fault toler, mission, error detect, diagnost, assess, recoveri, percent, maximum, avail, instrument, repair, tree, bit, busi, rate, mode
Security	encrypt, overflow, address resolut, resolut protocol, redund, confidenti, principl, admiss, host, access control, envelop, attack, dictionari, kei, authent, privileg, integr, buffer, author, arp, call, audit, activ, check, content, error, standard
Performance	ratio, throughput, effici, wait time, cach, propag, compress, stree, latenc, workload, bandwidth, path length, power consumpt, delai, util, space, speed, capac, channel, complet, respons time, resourc, sensor, storag, transmiss, usag, rate, instruct, process, case
Usability	wizard, effici, skill, percept, reliabl, wheel, repres, adapt, usabl, experi, interact, color scheme, life, energi, visual, menu, auto, save, analysi, attent, notif, navig, design, input, district, error, respons, field, control, access, view, displai

search terms from the expanded set of terms. Our prior work presented a ranking algorithm for ordering these documents by relevance [10]. Such a ranking is important if a user is to evaluate the retrieved documents with the aim of specifying requirements; however, it is not the focus of the work in this paper. The end result of this phase of the analysis is a set of textual chunks of information, which are potentially relevant to each of the components.

B. Detecting Quality Concerns

To detect quality concerns, the queries were expanded using a previously created tool [18]. We created an initial set of 30 representative search terms for each quality. These seeds terms were easily obtained, using simple searches e.g. searching for "security terms" in Google. Given a list of candidate search terms returned by the query-expansion algorithm, we accepted non-redundant terms. For example, we retained "access control" but not "access control list" or "access control service".

Search terms were passed to a standard search engine (e.g. Google) and a set of relevant documents were retrieved. Each of these documents was partitioned into chunks using a sliding window, and the Vector Space Model (VSM) was then used to identify the most relevant chunk in each document. The chunk was parsed to extract nouns and noun phrases [35] [31]. Finally, domain specific nouns and noun phrases were extracted. Our general domain corpus was constructed from a collection of over 50 e-books and other online documents that covered topics as broad as science fiction, business, nature, self-help, and even romance. *Domain Term Frequency (DTF)* and *Domain Specificity (DS)* metrics [18] were then used to identify and rank candidate terms and phrases for inclusion in the expanded set of search terms. They are computed as follows:

Domain Term Frequency (DTF) is the normalized term frequency for term t across multiple chunks of text as follows: $DTF(t) = \sum_{d \in D}(freq(t,d)/|d|)$ where $freq(t,d)$ is the total number of occurrences of term t in a relevent chunk of text d, and $|d|$ is the length of that chunk expressed as the total number of terms in d. Such normalized occurrences from all retrieved domain-specific chunks D are added together.

Domain Specificity (DS) measures the extent to which a term or term phrase is specific to text related to each quality concern, versus other general text. The domain specificity $DS(d,t)$ of term t in document d, is computed as follows:

$$DS(t,d) = ln[\frac{freq(t,d)}{\sum_{u \in d} freq(u,d)} / \frac{freq(t,G)}{\sum_{v \in G} freq(v,G)}] \quad (1)$$

where the first element $(freq(t,d))/(\sum_{u \in d} freq(u,d))$ is the normalized number of occurrences of term t in the domain-specific document d, and the second element is the normalized number of occurrences of t in the general corpus of documents. Domain specificity (DS) is then calculated as the average value of all domain specificities from each document from the collection:

$$DS(t) = \frac{1}{|D|} \sum_{d \in D} DS(t,d) \quad (2)$$

Terms not found in the general corpus, are assigned a high DS value of 10000.

We removed terms and phrases with overly high DS values (i.e. 10000) as they were too specific for general use (for example *finger* as related to *response time*). We also excluded any terms with $DTF < 1.0$ in order to remove less useful terms such as *object* and *tool* while reserving the majority of meaningful terms. As a result of this process, we ended up with 24 terms for *Availability*, 27 for *Security*, 30 for *Performance* and 32 for *Usability*. These are listed in Table II. DS values of each term gained from the all seeds and meeting our "inclusion criteria" (i.e., DS and DTF threshold) are accumulated, and were used as term weights in the next phase.

When a term serves as an indicator of a quality (e.g. *speed* for *performance*) and is also a natural part of the project's domain vocabulary (e.g. *train speed*) its inclusion can cause large numbers of false positives. We therefore remove any terms which appear as both quality and functional indicators. Taking *OBU* component of *PTC* as example, 24 terms are left for *Availability*, 24 for *Security*, 28 for *Performance* and 30 for *Usability*.

C. Identifying and Extracting Interdependencies

The goal is to identify requirements and other related textual information which describes a quality concern for a specific functional component, while attempting to minimize redundancy. A more naïve approach would present all sentences that describe qualities for a component, however as there is some degree of redundancy, this approach will return numerous close duplicates. We address this program by computing *Maximal Marginal Relevance (MMR)* scores [6]. In short, each sentence s in paragraphs related to component c is assigned a score indicating its relevance to quality concern q and the dissimilarity to other sentences that are already in the

selected sentences set *S'*. The score *R(s,q,c)* can be defined formally as following:

$$R(s,q,c) = 0.6 * Sim(s,q,c) + 0.4 * \max_{s_j \in S'} disSim(s,s_j) \quad (3)$$

where *Sim* and *disSim* are the similarity of sentence s with quality query q and dissimilarity between s with those selected sentences, respectively. The dissimilarity between s and each sentence in *S'* is computed and the maximum value is chosen to represent their dissimilarity. Sentences with the highest R score are iteratively added to *S'* until a predefined number of sentences is achieved or all elements in the candidate sentence set S have been checked.

The similarity score *Sim(s,q,c)* is primarily based on the frequency of search terms of quality q in component relevant sentence s and the term weights; however, the length of sentence s is also considered. Let T_q be the set of search terms of quality q. For each term t, let *freq(t,s)* be the frequency of term t appearing in sentence s and let $v_{q,t}$ be the weight of term t for quality q. Let T_s be the set of terms in sentence s. The cardinality of T_s is the length of sentence s.

The similarity score of component c-related sentence s with quality q is defined formally as follows:

$$Sim(s,q,c) = \frac{\sum_{t \in T_q} freq(t,s) * v_{q,t}}{|T_s|} \quad (4)$$

The dissimilarity *disSim* between sentences s and s_j, based on the maximal lexical diversity between two sentences, is the ratio of the number of terms in sentence s which are not contained by s_j to the total number of terms in sentence s. Let T_{s_j} and T_s be the set of terms in sentences s_j and s. $|T_s|$ is the cardinality of terms in sentence s. The *disSim* can be depicted as:

$$disSim(s,s_j) = \frac{|\{t \notin T_{s_j} | t \in T_s\}|}{|T_s|} \quad (5)$$

IV. Evaluation

We conducted an experiment to evaluate the first research question **RQ1:To what extent can interdependencies between functional components and quality concerns be automatically extracted from a collection of domain documents?** The experiment involved two steps: first, developing a reference set containing all quality related sentences associated with each functional component, and then evaluating HRM by comparing its results to the reference set.

A. Building a Reference Set

Given the nature of our problem i.e. mining requirements-related information from large repositories of domain documents, identifying all relevant sentences for each component and each of the four quality concerns is overly time-consuming and would take many hundreds of hours of work. Therefore we constructed a reference set for the *Onboard unit* from *PTC*. *Onboard unit* was selected due to data availability from our collaborators.

The process involved using a snow-balling technique to collect as many candidate sentences as possible. First, all previously identified search terms were used to retrieve sentences containing at least one search term from the domain documents. Then all neighboring sentences were reviewed to check if they described the same functional components. If so, they were added to the candidate set and their core phrases were extracted and added to the list of search terms to be incrementally used to retrieve additional candidate sentences. We then manually reviewed each candidate sentence and tagged those that described quality concerns. Ultimately we identified 551 sentences, including 11 *Availability*, 13 *Security*, 14 *Performance* and 73 *Usability* ones. The identified sentences included duplicates and near duplicates. However, the goal of the automatic selection algorithm is to select only one representative sentence from each group of these closely related sentences.

B. Baseline

To evaluate our approach we compared it against a basic baseline constructed using the Vector Space Model (VSM) [29]. VSM was selected as the baseline because it has been shown to perform consistently well in comparison to other techniques for computing similarity across multiple software related data sets [22]. It represents all documents as vectors of terms and computes similarity as the *cosine* of the angle between two vectors. We applied the same automatic two-phrase extraction used in our approach, meaning that OBU-related sentences were first extracted and then quality concerns identified. Term weights for the vectors were calculated using *term frequency-inverse document frequency* (tf-idf) [29]. Expanded search terms for both functional components and quality concerns were used for both our approach and the baseline. The only difference in the two approaches was the use of query expansion after removing the domain functional terms for HRM as described in Section III.

Results were evaluated using *Recall*, *Precision*, *F-Measure* and *F-2*. *Recall* calculates the fraction of relevant sentences that are retrieved, *precision* measures the fraction of retrieved sentences that are relevant, while *F-Measure* computes the harmonic mean of *recall* and *precision*. Finally, we also report the *F-2* measure which assigns recall twice the weight of precision. This reflects the fact that returning a more complete, albeit less precise set of candidate sentences, can be beneficial for engineers as filtering out irrelevant sentences requires less work than searching for missing ones.

C. Results

Results from this experiment are reported in Table III. Our HRM approach achieved higher recall than VSM for extracting sentences related to *Security*, *Availability* and *Performance*, but slightly lower recall than VSM for *Usability* (i.e.,0.89 vs. 0.96). However, HRM returned higher precision than VSM for all quality types. These results are reflected in both the F-Measure and F2-Measure where we observe that HRM outperforms VSM on *F-Measure* and *F-2* values for all four quality concerns. Overall, on average 83% of quality-related sentences associated with a component are directly detected by

TABLE III: Results achieved using VSM (baseline) versus HRM (our approach) for extracting sentences describing the interdependencies between OBU and four quality concerns.

Alg.	Qua.	Recall	Precision	F-Measure	F-2
VSM	U.	0.96	0.31	0.46	0.67
	P.	0.64	0.08	0.14	0.27
	S.	1.0	0.16	0.28	0.49
	A.	0.54	0.27	0.36	0.45
HRM	U.	0.89	0.69	0.78	0.84
	P.	0.79	0.29	0.41	0.58
	S.	1.0	0.35	0.52	0.73
	A.	0.64	0.28	0.39	0.51

U.:Usability P.:Performance S.:Security A.:Availability

HRM compared to 78% for VSM. We can therefore address **RQ1**, and report that in this experiment HRM outperformed VSM.

To provide additional insights we also report the distribution of identified quality concerns against randomly selected components of PTC, EHR, and MIP. We highlight cells representing higher concentrations of quality-related sentences. These results indicate that HRM identifies appropriate quality concerns and does not superflously detect quality concerns across all components. For example, the PTC Brake system has a predominance of availability-related sentences identified, while the Edge Message Protocol has primarily security ones.

V. Categorize the Interdependencies

In this section we concentrate on the categories of knowledge contained in the interdependencies extracted from domain documents. We define an interdependency as a statement describing the component which is also classified as a quality concern. For example, the statement that *The PCA pump shall switch to KVO infusion rate upon receiving an authenticated command through its ICE interface* is associated with the *pump operation* component as well as the *security* quality concern. It therefore represents an interdependency.

We adopted a grounded theory approach [9] and followed six steps to perform a category analysis and validity evaluation with the goal of identifying categories of practical significance to system requirements.

- The first author, who had constructed all three domain models, carefully reviewed each statement associated with components from the three domains, and manually tagged those which she considered relevant to each of the four quality concerns. These were labeled as *interdependencies*.
- The same author then reviewed each of the extracted interdependencies in order to categorize their possible role in the requirements analysis process. While performing this categorization, the researcher created a codebook describing the rules for identifying each of the categories.
- The second author, who has prior industrial experience in the EHR domain, a long-term research partnership in the PTC domain, and is familiar with literature from the MIP domain, reviewed the codebook and the mappings between the categories and the extracted interdependencies built by the first author.

		Avai.	Perf.	Sec.	Usa.
PTC	wireless network	1	12	2	0
	antenna	1	0	0	2
	Edge Message Protocol	4	1	31	4
	Base Commu. Pkg.	12	2	0	0
	Cluster Control	6	1	0	0
	Commu. Mgmt Unit	6	1	0	1
	Mobile Commu.	1	2	4	0
	Backbone commu. net.	0	0	0	0
	Front End Proc.	6	1	0	0
	Wayside Interface Unit	40	25	43	10
	Track Warrant Control	2	0	0	1
	Wayside Detector	0	0	0	0
	Wayside Stat. Relay	0	0	0	0
	Locomotive Control	2	2	0	4
	Brake	59	15	6	1
	Remote Control Locomotive	1	7	0	0
	Event Recorder	1	0	0	4
	Locomotive Net.	0	0	0	0
	Loco. Interface Gt.	3	0	2	0
	Onboard Unit	5	3	0	6
	Comp. Display Unit	0	1	0	5
	Human Machine Interface	6	0	1	18
	Dispatch System	6	2	6	3
	Back Office Server	14	4	6	3
	System Mgmt. Agent	19	13	6	1
	Comm. in EIC	57	51	28	8
	EIC interface	0	0	0	0
EHR	Clinical Decision Support	2	0	1	0
	Admini. System	0	0	1	1
	Electronic Commu.	1	0	5	0
	Report. & pop. Health mngmt.	0	0	0	0
	Clinical docu.	0	0	0	1
	Comp. Provider Order Entry	0	0	1	3
MIP	Drug library	0	0	0	2
	Communication	3	1	3	3
	Barcode System	0	0	0	1
	Human Factors Engineering	0	0	0	0
	Pump Operation	1	0	0	25
	Safety features	2	0	0	8

Fig. 2: The distribution of quality concerns across the functional components of the three domains.

- These two researchers discussed disagreements until they reached agreement on the final codebook.
- The external evaluator (i.e., a blind coder), who was not a member of the research group, not familar with the three domains but with solid understanding of usability, performance, security, and availability requirements, used the codebook to categorize each interdependency.
- The inter-rater agreement between researchers and the external evaluator's results was computed using Cohen's kappa [14].

The following 10 categories were identified:

1) **New Requirement**: The text serves as a requirement specifying system functionality, properties, and constraints [25]. No semantically equivalent requirement currently exists in the specification. For example in the EHR

domain: *the risk analysis implementation specification requires covered entities to "Conduct an accurate and thorough assessment of the potential risks and vulnerabilities to the confidentiality, integrity, and availability of electronic protected health information held by the covered entity"*.

2) **Reinforcing Requirement**: The text represents a requirement, but a semantically similar requirement already exists in the specification. For example, *this key activity (conducting an analysis of existing physical security vulnerabilities) may be performed as part of the risk analysis implementation specification* is similar to the previously identified *New Requirement*, while providing additional information that *an analysis of existing physical security vulnerabilities* should be contained in the *accurate and thorough assessment*.
3) **Principle**: The text provides an explanation about inputs, outputs, processing technology etc. An example is *"this functionality would display when a referral order was executed, when an appointment was made with the patient (or when a cancellation occurred)..."*
4) **Experience or Suggestion**: The text documents experiences and/or suggestions related to a specific problem. One example from the EHR domain is *"consider asking the business associate to conduct a risk assessment that addresses administrative, technical, and physical risks..."*.
5) **Standards or Policies**: The text describes rules that must be followed in order to comply with specific standards or policies in the domain, for example *"this guidance also places significant emphasis on the importance of risk analysis and risk management strategies, policies and procedures, ...during its remote access, storage, and transmission"*.
6) **Technology Tradeoffs**: The text discussess advantages or disadvantages of a technology. For example, *"if a single sign-on solution is used, then generally the user is authenticated to the storage encryption technology during os login, so the files are not protected against these threats once os login occurs"*.
7) **Requirements Elicitation Prompts (sample question)**: The text poses a question which could trigger requirement discovery. For instance, *"is the process implemented in such a way that it does not compromise the authentication information (password file encryption, etc)?"*.
8) **Rationales (hazards)**: The text describes a hazard or a technology for mitigating the hazard. An example from the MIP domain is *"these modular iv pumps are designed to reduce medication errors at the bedside through the use of a dose error reduction system ..."*.
9) **Industry solution**: The text references a specific product's name and describes a specific solution. For example *"...NYP is currently using pdf automated transmission to improve .. capabilities "*.
10) **General knowledge**: Describes common knowledge that engineers in the domain should know. For example *".. the top 10 security risks about which organizations should be concerned: malicious insider misuse, external attack,...etc"*.

TABLE IV: The number of sentences in each category for the associated sentences about the functional component(s) and each quality concern in the three domains.

	PTC-OBU				EHR				MIP			
	A.	P.	S.	U.	A.	P.	S.	U.	A.	P.	S.	U.
New Req.	3	8	8	18	20	6	41	2	4	3	1	5
Reinforcing Req.	0	2	1	36	3	0	16	0	0	0	1	0
Principle	5	1	3	8	18	7	46	2	2	3	3	7
Exp./Suggest	0	0	0	0	7	7	55	1	14	9	1	23
Standards/Pol	0	1	0	0	4	0	6	0	0	1	0	0
Tech Tradeoffs	2	1	0	0	5	2	16	0	0	2	0	1
Req Elicit.Prompts.	0	0	0	0	4	1	2	1	0	1	0	0
Rationales (hazards)	2	0	2	0	0	0	0	0	0	0	0	2
Industry sol.	1	0	0	2	1	2	7	0	2	4	0	1
Gen. knowledge	0	0	0	1	8	2	15	0	9	2	1	4

A.: Availability, P.:Performance, S.:Security, U.:Usability

A. Analysis of Categories

In order to measure the inner-agreement between the categories made by our two researchers and the blind coder, we utilized *Cohen's kappa* [14] which is a quatitative metric. The *kappa* values on *OBU*, *EHR* and *MIP* are 0.715, 0.664 and 0.733 respectively. According to Viera et al. [33], 'Good' agreement has been achieved on all these three domains (i.e., *kappa*: 0.61-0.80). We therefore can say that our category guidance is valid and clear.

While only certain pieces of information are marked specifically as *requirements*, other types of information can provide support for requirements discovery. By analyzing requirements assigned to each category, and the distribution of categories across the four quality concerns, as depicted in Table IV, we were able to observe the following:

- The interdependencies represented diverse knowledge types across all three domains.
- Given redundancy of domain documents, reinforcing requirements, can provide additional insights about a requirement. Approximately 50% of requirements fall into this category.
- Quality concerns are often described in sentences marked as design decisions. This makes sense as quality concerns impact architectural design. The requirements engineer may need to search through design decisions in order to extract and specify quality concerns for a component.

With respect to the second research question **RQ2**, we conclude that at least 10 types of knowledge can be obtained from the interdependency descriptions extracted from domain documents. All of these 10 categories can potentially provide assistance to the engineers in understanding the domain and the existing system requirements. Some contribute requirements directly, while others such as *Experience or Suggestions*, could help the engineer to improve system requirements.

VI. Case Study

In this paper we have proposed, implemented, and evaluated our HRM approach for mining associations between quality concerns and functional requirements. In this section we report on a small case study we conducted to discover security requirements for PTCs *Onboard Unit*. The case study, while relatively small, is designed as an initial way to address **RQ3 "Can a software engineer without domain knowledge construct quality-related requirements for a functional component using HRM?"**. As an exploratory study, we recruited a single developer who had four years of prior experience working in a large Systems Engineering Group on CyberPhysical Systems. He had not previously worked in the Positive Train Control domain.

We applied HRM and extracted security related information. Further, each piece of information generated by HRM has a link back to its original source document and page number, and the extracted text was highlighted in yellow. 14 textual descriptions were retrieved including five classified as *new* requirements, four classified as *reinforcing* requirements, two as *rationales*, and three as *principles*. The engineer participating in our study was tasked with reviewing each of the ranked pieces of information, selectively following traceability links back to the original documents, reviewing the descriptions, and specifying requirements. We set no constraints on the time he spent performing the task or on what information he viewed in any documents he choose to open.

The engineer took 80 minutes to perform the task and produced 11 security-related OBU requirements as depicted in Table V. The engineer chose to conduct the task by (1) *reading* a simple overview of PTC which provided a one page pictorial summary including 2-3 descriptive sentences for each of PTC's five major components (5 minutes), (2) *reviewing* the HRM extracted sentences, retrieving related documents, and reading 1-2 paragraphs surrounding the highlighted statement, (3) *identifying* two 'anchor' documents through performing step #2, and using these to check for similarities between emergent requirements in other documents. The engineer commented that the sentences which linked to table cells in documents were difficult to use because they tended not to provide sufficient context to understand the requirement. He therefore made most use of documents with richer textual descriptions.

The requirements produced by the engineer were complete with respect to the security requirements provided by our industrial collaborators. Further, they were clearly specified as observable in Table V. It was interesting that the engineer identified more individual requirements than HRM had retrieved and labeled; however, upon inspection we determined that the additional requirements were derived from text surrounding the HRM highlighted sentences in the collection of relevant source documents. This confirmed the importance of providing links back to the original documents rather than only showing the extracted text. Further, we noted that the engineer used all types of source information – not just sentences marked as potential *requirements* in the HRM generated documents.

TABLE V: Security Requirements for the PTC On board unit written by a Systems Engineer using HRM without any prior knowledge of the PTC Domain.

1	To ensure that permission is granted to the correct train, proper form of authentication, in form of session authentication and a Message authentication code is required between the Train Unit and the BackOffice.
2	The OBC braking algorithm must calculate the speed reduction using the provided external signals (wayside signal, track integrity circuits, speed restrictions). In case the required stopping distance can not be guaranteed, a full stop must be performed.
3	To prevent software or memory errors PRT application shall use Error Detection Codes to verify software and memory integrity.
4	Regular software integrity checks also need to be performed on OBC startup.
5	The alpha-numeric instruction string entered into the OBC by the train crew need to be encryped and can be decrypted by the MWP passcode.
6	Before any automatic actions are performed by the OBC a user interface must provide information to the train crews including audio and visual warnings.
7	In case of telemetry signal loss, after the predefined amount of time, the train functions under reduced capability allowing the it only to operate within the last limit of authority
8	If the control equipment or the gps system malfunctions the system must be shutdown completely or reduced capability depending on the type of failure.
9	Both audible and visual signals must inform operators and dispatchers in such cases.
10	In case of connection loss between the OBC and the EIC PRT a fall back strategy must be provided and the train must stop.
11	The Train management system must display alerts for all occurring exceptions. No actions performed unless the exception corresponds to predefined actions

VII. Related Work

Related work primarily falls under two areas associated with the interplay of functional requirements and quality concerns, and techniques for detecting and extracting quality-related requirements.

Eckhardt et al. [17] analyzed 530 quality related requirements extracted from 11 industrial requirements specifications and claimed that most "non-functional" requirements are not non-functional as they describe behavior of a system. Karlsson et al. [19] found that interdependencies between requirements are a major problem in market-driven software development, which highlights the importance of discovering them early in the requirements process. Carlshamre et al. [7] identified six different interdependency types of requirements in industry based on a survey of five companies. Svensson et al. [30] provided further proof of these six requirement interdependency types by interviewing product managers and project leaders from five software companies. They also showed that these interdependencies lack sufficient emphasis in requirements elicitation, analysis and documentation in industry. Anish et al. [3] claimed that quality requirements associated with functional components were underspecified. They developed a workflow and sets of "probing questions" to assist business analysts in the elicitation of quality-related requirements for several types of functional component. Finally, goal-oriented methods, such as the NFR framework [11], KAOS [32], and i* family [8, 34], depict quality concerns (or non-functional

requirements), functional components, and their interdependencies in a single model.

Several researchers have described techniques for automatically extracting quality concerns from requirements specifications. While they did not focus on the interplay of functional components and quality concerns, their work is highly relevant to our approach. Cleland-Huang et al. [12] presented a Goal Centric approach for managing and maintaining the system quality concerns. The quality concerns, non-functional requirements and their interdependency, and the related operationalizations (i.e., functional points) were modeled in a Softgoal Interdependency Graph. However the goal structures were created manually. Cleland-Huang et al. [13] proposed a technique to automatically detect and classify non-functional requirements describing quality concerns from a set of free-form requirement documents. Rahimi et al. [28] extracted and visualized quality concerns from a set of requirement specifications. Mirakhorli et al. [24, 23] proposed an architectural tactic detector to discover quality concerns from source code or text documents. Lian et al. [20] used the same tactic detector to identify architectural tactics and then visualized associations between architectural tactics in source code and quality concerns distributed across the code base. However, all of these techniques focus primarily on quality concerns without exploring their crosscutting impact upon functional components. In addition, existing approaches focus on retrieving information from source code and requirements, whereas HRM retrieves information from both source code and large repositories of unstructured domain documents.

VIII. Threats to Validity

There are several threats to validity related to our study. The first threat is associated with the reference set used to evaluate our approach. While we followed a rigorous and systematic approach, some statements may have been miscategorized, especially if they make only brief reference to a quality and a judgment was required. However, to mitigate this bias we compare our approach against VSM using the same datasets and reference sets. Furthermore, in our quantitative evaluation of HRM, we compared only against the *OBU* components of the *PTC* domain, for which a complete traceability set had been provided by our industrial collaborators.

The second important threat is the fact that we addressed RQ3 only through a single study with a single developer. However, the developer was an engineer with four years of industry experience but not prior PTC domain knowledge. He was therefore well representative of our target group.

One important threat to external validity is the process of searching for and retrieving documents to place in the domain repositories, and collecting initial search terms for functional and quality concerns. The final outcome of our approach is dependent upon the effort and quality of these manual steps. While we did not conduct multi-user studies to investigate the impact of the user of HRM's effectiveness, all user-dependent steps represent common domain analysis practices and utilize publicly available search tools.

IX. Conclusion

The work described in this paper is designed to aid requirements engineers in discovering quality related domain knowledge so that they can specify both functional and quality concerns early in the project. We have presented HRM which is capable of mining quality concerns associated with previously mined functional components. Our final case study revealed that the engineer used multiple information sources from the identified knowledge types, and that he leveraged embedded traceability links to inspect the context of information retrieved by HRM. Ultimately, the engineer was able to develop well-written, and relatively complete quality-related requirements for the component he was assigned.

Future work will involve additional user studies. Given results from our initial study we will also pay attention to the type of document and context in which the retrieved information is extracted in order to return high quality, more useful information, to the engineer.

Acknowledgment

Funding for this work has been provided by the US National Science Foundation grant US National Science Foundation Grant CCF:1441791 and by the National Science Foundation of China Grant No.61672078.

References

[1] M. U. Ahmed. ebay-ecommerce platform, a case study in scalability. *McGill University*, pages 1–13.

[2] D. Ameller, C. Ayala, J. Cabot, and X. Franch. How do software architects consider non-functional requirements: An exploratory study. In *2012 20th IEEE International Requirements Engineering Conference (RE)*, pages 41–50. IEEE, 2012.

[3] P. R. Anish, B. Balasubramaniam, A. Sainani, J. Cleland-Huang, M. Daneva, R. J. Wieringa, and S. Ghaisas. Probing for requirements knowledge to stimulate architectural thinking. In *Proceedings of the 38th International Conference on Software Engineering, ICSE 2016, Austin, TX, USA, May 14-22, 2016*, pages 843–854, 2016.

[4] A. Banerjee, K. K. Venkatasubramanian, T. Mukherjee, and S. K. S. Gupta. Ensuring safety, security, and sustainability of mission-critical cyber–physical systems. *Proceedings of the IEEE*, 100(1):283–299, 2012.

[5] M. Broy. Rethinking nonfunctional software requirements: A novel approach categorizing system and software requirements. *Software Technology*, 10.

[6] J. Carbonell and J. Goldstein. The use of mmr, diversity-based reranking for reordering documents and producing summaries. In *Conference on Research and development in information retrieval*, pages 335–336. ACM, 1998.

[7] P. Carlshamre, K. Sandahl, M. Lindvall, B. Regnell, and J. N. och Dag. An industrial survey of requirements interdependencies in software product release planning. In *Symposium on Requirements Engineering*, pages 84–91. IEEE, 2001.

[8] J. Castro, M. Kolp, and J. Mylopoulos. Towards requirements-driven information systems engineering: the tropos project. *Info. systems*, 27(6):365–389, 2002.

[9] K. Charmaz. *Constructing grounded theory*. Sage, 2014.

[10] L. Chen, M. A. Babar, and B. Nuseibeh. Characterizing architecturally significant requirements. *IEEE Software*, 30(2):38–45, 2013.

[11] L. Chung, B. A. Nixon, E. Yu, and J. Mylopoulos. *Non-functional requirements in software engineering*, volume 5. Springer Science & Business Media, 2012.

[12] J. Cleland-Huang, R. Settimi, O. BenKhadra, E. Berezhanskaya, and S. Christina. Goal-centric traceability for managing non-functional requirements. In *Proceedings of the 27th international conference on Software engineering*, pages 362–371. ACM, 2005.

[13] J. Cleland-Huang, R. Settimi, X. Zou, and P. Solc. Automated classification of non-functional requirements. *Requirements Engineering*, 12(2):103–120, 2007.

[14] J. Cohen. A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20:37–46, 1960.

[15] D. de Champeaux, D. Lea, and P. Faure. *Chapter 13, Domain Analysis, Object-oriented system development*. Addison-Wesley, 1993.

[16] J. Eckhardt, D. M. Fernandez, and A. Vogelsang. How to specify non-functional requirements to support seamless modeling? a study design and preliminary results. In *Empirical Software Engineering and Measurement (ESEM)*, pages 1–4. IEEE, 2015.

[17] J. Eckhardt, A. Vogelsang, and D. M. Fernández. Are non-functional requirements really non-functional?: an investigation of non-functional requirements in practice. In *Proceedings of the 38th International Conference on Software Engineering*, pages 832–842. ACM, 2016.

[18] M. Gibiec, A. Czauderna, and J. Cleland-Huang. Towards mining replacement queries for hard-to-retrieve traces. In *International conference on Automated software engineering*, pages 245–254. ACM, 2010.

[19] L. Karlsson, Å. G. Dahlstedt, B. Regnell, J. N. och Dag, and A. Persson. Requirements engineering challenges in market-driven software development–an interview study with practitioners. *Information and Software technology*, 49(6):588–604, 2007.

[20] X. Lian, A. Fakhry, L. Zhang, and J. Cleland-Huang. Leveraging traceability to reveal the tapestry of quality concerns in source code. In *8th IEEE/ACM International Symposium on Software and Systems Traceability, SST 2015, Florence, Italy, May 17, 2015*, pages 50–56, 2015.

[21] X. Lian, M. Rahimi, J. Cleland-Huang, L. Zhang, R. Ferrai, and M. Smith. Mining requirements knowledge from collections of domain documents. In *Requirements Engineering Conference, RE Beijing, China*, pages 156–165, 2016.

[22] S. Lohar, S. Amornborvornwong, A. Zisman, and J. Cleland-Huang. Improving trace accuracy through data-driven configuration and composition of tracing features. In *Foundations of Software Engineering, Saint Petersburg, Russia*, pages 378–388, 2013.

[23] M. Mirakhorli and J. Cleland-Huang. Detecting, tracing, and monitoring architectural tactics in code. *IEEE Trans. Software Eng.*, 42(3):205–220, 2016.

[24] M. Mirakhorli, Y. Shin, J. Cleland-Huang, and M. Çinar. A tactic-centric approach for automating traceability of quality concerns. In *34th International Conference on Software Engineering, ICSE 2012, June 2-9, 2012, Zurich, Switzerland*, pages 639–649, 2012.

[25] L. Northrop, P. Clements, F. Bachmann, J. Bergey, G. Chastek, S. Cohen, P. Donohoe, L. Jones, R. Krut, R. Little, et al. A framework for software product line practice, version 5.0. *SEI.–2007–http://www. sei. cmu. edu/productlines/index. html*, 2007.

[26] B. Nuseibeh. Weaving together requirements and architectures. *IEEE Computer*, 34(3):115–117, 2001.

[27] K. Pohl. *Requirements engineering: fundamentals, principles, and techniques*. Springer Publishing Company, Incorporated, 2010.

[28] M. Rahimi, M. Mirakhorli, and J. Cleland-Huang. Automated extraction and visualization of quality concerns from requirements specifications. In *Requirements Engineering Conference, RE Karlskrona, Sweden*, pages 253–262, 2014.

[29] G. Salton, A. Wong, and C. S. Yang. A vector space model for automatic indexing. *Commun. ACM*, 18(11):613–620, Nov. 1975.

[30] R. B. Svensson, T. Gorschek, and B. Regnell. Quality requirements in practice: An interview study in requirements engineering for embedded systems. In *International Working Conference on Requirements Engineering: Foundation for Software Quality*, pages 218–232. Springer, 2009.

[31] D. Tufis and O. Mason. Tagging romanian texts: a case study for qtag, a language independent probabilistic tagger. In *Proceedings of the First International Conference on Language Resources and Evaluation (LREC*, pages 589–596, 1998.

[32] A. Van Lamsweerde. Goal-oriented requirements engineering: A guided tour. In *Symposium on Requirements Engineering*, pages 249–262. IEEE, 2001.

[33] A. J. Viera, J. M. Garrett, et al. Understanding interobserver agreement: the kappa statistic. *Fam Med*, 37(5):360–363, 2005.

[34] E. S. Yu. Towards modelling and reasoning support for early-phase requirements engineering. In *Requirements Engineering, 1997., Proceedings of the Third IEEE International Symposium on*, pages 226–235. IEEE, 1997.

[35] X. Zou, R. Settimi, and J. Cleland-Huang. Improving automated requirements trace retrieval: a study of term-based enhancement methods. *Empirical Software Engineering*, 15(2):119–146, 2010.