Mining Associations between Quality Concerns and Functional Requirements

Xiaoli Lian[1]
Beihang University
DePaul University
lxl_tuizi@hotmail.com

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

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 domain Positive Train Control and show that it significantly outperforms a basic information retrieval approach.

1 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 [1]. 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 [2]. However, quality requirements cannot be cleanly separated from functional concerns as they exhibit dependencies on each other [3]. In fact, quality concerns are often underspecified functional requirements and are interwoven throughout the functional components of a system [3].

Domain analysis techniques which rely upon mining requirements knowledge, could therefore benefit by focusing not only on extracting functional requirements, but also on discovering 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, and other associated literature.

[1] Xiaoli Lian is a visiting scholar in DePaul University now.

2 Our Approach: HRM

HRM consists of seven 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). 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 [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 [4]. In Step 5, the engineer identifies quality concerns of interest. In Step 6, the expanded term is used to search through the previously retrieved component-related text to identify quality concerns. In Step 7, the information is collated and be analyzed. The quality concern extraction (Steps 3, and 5-7) 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.

2.1 Query Expansion and Search

The initial set of search terms for each component is comprised of 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 [4]. 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 initial 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 in order 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.

2.2 Detecting Quality Concerns

To detect quality concerns, expanded queries using a previously created tool [15]. 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-repetitious 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 [16]. 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 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 relevant 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 to remove less useful terms such as object and tool while reserving the majority of meaningful terms.

2.3 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 naive approach would present all sentences that describe qualities for a component, even if they are near duplicates, or very similar to previously discovered ones. We therefore leverage Maximal Marginal Relevance (MMR)[13]. 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 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 Tq 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 $v_{q,t}$ is the weight of term t to quality q. Let Ts be the set of terms in sentence s. The cardinality of Ts is used as the length of sentence s. The similarity score of component c-related sentence s with quality q is defined formally as follows.

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

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_{sj} 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:

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

3 Evaluations

3.1 Case Systems

We 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

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 [4], 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

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 the other (who has several years of industrial experience in the EHR domain).

- Medical Infusion Pump

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 includes 100 files constituting 66.7MB of data. In addition, we utilized a requirements specification for the Medical Infusion Pump containing 126 requirements.

3.2 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. We therefore used the component Onboard unit 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, in 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 it if it described a quality concern.

3.3 Baseline

To evaluate our approach, we compared it against a basic baseline Vector Space Model (VSM). The Vector Space Model (VSM) [14] was selected as the baseline due to its common usage in the text similarity computation in information retrieval. It represents both the query and documents as vectors and then computes the Cosine of the angle between the two vectors to calculate their similarity. We applied the same 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 with term frequency-inverse document frequency (tf-idf). Expanded search terms for both functional components and quality related concerns were used for both our approach and the baseline.

Results were evaluated using Recall, Precision and F-Measure. Recall calculates the fraction of relevant sentences that are retrieved, and precision is the fraction of retrieved sentences that are relevant. F-Measure combines recall and precision with their harmonic mean.

3.4 Results

Results from this experiment are reported in Table 1 and show recall versus precision as well as F-Measure. Our HRM approach performed better for extracting sentences related to Security, Availability and Performance, but slightly weaker recall than VSM for Usability (i.e.,0.89 vs. 0.96) and significantly stronger Precision (i.e., 0.69 vs. 0.31). HRM outperforms VSM on F-Measure

values for all four quality concerns. Therefore we claim that 64% of quality-related sentences associated with a component are directly detected by HRM, which outperforms VSM.

Table 1 HRM and VSM work on the PTC documents to extract the sentences describing about the interdependencies between OBU and four quality concerns

Quality	Algo.	Recall	Precision	F-Measure
Usability	HRM	0.89	0.69	0.78
	VSM	0.96	0.31	0.47
Performance	HRM	0.79	0.28	0.41
	VSM	0.64	0.08	0.14
Security	HRM	1.0	0.35	0.52
	VSM	1.0	0.16	0.28
Availability	HRM	0.64	0.28	0.39
	VSM	0.54	0.27	0.36

		Ava.	Perf.	Sec.	Usa.		Ava.	Perf.	Sec.	Usa.
PTC	wireless net.	1	12	2	0	Brake	59	15	6	1
	antenna	1	0	0	2	Remote Ctl Loco	1	7	0	0
	edge msg protocol	4	1	31	4	Event Recorder	1	0	0	4
	Base Commu. Pkg.	12	2	0	0	Locomotive Net.	0	0	0	0
	Cluster Control	6	1	0	0	Loco. Inter. Gt.	3	0	2	0
	Commu Mgmt Unit	6	1	0	1	Onboard Unit	5	3	0	6
	Mobile Commu.	1	2	4	0	Comp. Display Unit	0	1	0	5
	Backbone commu. net.	0	0	0	0	Human Machine Inter.	6	0	1	18
	Front End Proc.	6	1	0	0	Dispatch System	6	2	6	3
	Wayside Inter. Unit	40	25	43	10	Back Off. Server	14	4	6	3
	Track Warrant Ctrl	2	0	0	1	Sys Mgmt. Agent	19	13	6	1
	Wayside Detector	0	0	0	0	Comm. in EIC	57	51	28	8
	Wayside Stat. Relay	0	0	0	0	EIC interface	0	0	0	0
	Loco. Control	2	2	0	4					
EHR	Clinical decision support	2	0	1	0	Report. & pop. Health mngmt.	0	0	0	0
	Admin.system	0	0	1	1	Clinical docu.	0	0	0	1
	Electronic commu.	1	0	5	0	CPOE	0	0	1	3
MIP	Drug library	0	0	0	2	HFE	0	0	0	0
	Communication	3	1	3	3	Pump operation	1	0	0	25
	Barcode system	0	0	0	1	Safety features	2	0	0	8

Figure 1 The distribution of quality concerns across the functional components of the three domains

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 suggest that HRM identifies appropriate quality concerns and does not assign quality concerns homogeneously across all

components. For example, as expected, the PTC Brake system has 59 availability-related sentences identified.

4 Related Works

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. [8] 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. [9] 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. [10] extracted and visualized quality concerns from a set of requirement specifications. Mirakhorli et al.[12] proposed an architectural tactic detector to discover quality concerns from source code or text documents. Lian et al. [11] detected architectural tactic code with the classifier [9] and provided three visualization techniques to show the traceability between source code and architectural tactics. However, these works focus on quality concerns only. Furthermore, our HRM approach works on large repositories of unstructured domain documents, and is not limited to searching through relatively small requirements specifications.

2.7 Conclusions

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.

References

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

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

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

[4] X.Lian, M.Rahimi, J.Cleland-Huang, L.Zhang, R.Ferrai and M.Smith, “Mining requirements knowledge from collections of domain documents”. In 24th IEEE International Requirements Engineering Conference, RE2016, Beijing, China, September 12-16,2016, pages 156-165, 2016.

[5] 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.

[6] 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 Requirements

Engineering, 2001. Proceedings. Fifth IEEE International Symposium on, pages 84–91. IEEE, 2001.

[7] 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.

[8] 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.

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

[10] M. Rahimi, M. Mirakhorli, and J. Cleland-Huang. “Automated extraction and visualization of quality concerns from requirements specifications”. In IEEE 22nd International Requirements Engineering Conference, RE 2014, Karlskrona, Sweden, August 25-29, 2014, pages 253–262, 2014.

[11] 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.

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

[13] J. Carbonell and J. Goldstein. “The use of mmr, diversity based reranking for reordering documents and producing summaries”. In Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval, pages 335–336. ACM, 1998.

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

[15] M. Gibiec, A. Czauderna, and J. Cleland-Huang. “Towards mining replacement queries for hard-to-retrieve traces”. In Proceedings of the IEEE/ACM international conference on Automated software engineering, pages 245–254. ACM, 2010.

[16] 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.