IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received February 8, 2019, accepted February 28, 2019, date of publication March 5, 2019, date of current version April 2, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2903133

Annotation of Software Requirements Specification (SRS), Extractions of Nonfunctional Requirements, and Measurement of Their Tradeoff

MUHAMMAD ASIF[1], ISHFAQ ALI[1], MUHAMAD SHERAZ ARSHED MALIK[2], MUHAMMAD HASANAIN CHAUDARY[3], SHAHZADI TAYYABA[4], AND MUHAMMAD TARIQ MAHMOOD[5], (Senior Member, IEEE)

[1]Department of Computer Science, National Textile University, Faisalabad 37610 , Pakistan
[2]Department of Information Technology, Government College University Faisalabad, Faisalabad, Pakistan
[3]Department of Computer Science, COMSATS University Islamabad at Lahore Campus, Lahore 44000, Pakistan
[4]Department of Computer Engineering, The University of Lahore, Lahore 54000, Pakistan
[5]School of Computer Science and Engineering, Korea University of Technology and Education, Cheonan 31253, South Korea

Corresponding author: Muhammad Tariq Mahmood (tariq@koreatech.ac.kr)

This work was supported by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education under Grant 2016R1D1A1B03933860.

ABSTRACT Software requirement artifacts such as manuals request for proposals, and software requirements specification (SRS) are commonly focused on functional requirements. In most SRS files, nonfunctional requirements do not formally encoded or encoded as a whole, not for an individual design problem. Moreover, these nonfunctional requirements are intermingled with functional requirements. Therefore, these nonfunctional requirements need special attention to understand for successful project development. These nonfunctional requirements have an impact on each other and optimal tradeoff is required for balanced nonfunctional requirements set. NFRs have a negative and positive tradeoff with each other such as increase confidentiality, decrease the availability, and enhance authenticity. So, an optimum tradeoff among these design problem within a module is required to have better design decisions. Instead of considering all nonfunctional requirements, the NFRs that have mutual tradeoff is considered. In this paper, we devised a novel document annotation scheme for SRS and extracted nonfunctional requirements from these annotated artifacts. In the next step, we classified NFRs into two classes security triad and performance triad, and the cost is assumed constant for each NFR. From the design problem, the tradeoff ratio is calculated among NFRs associated with it. Then, the production possibility graph is plotted to estimate the optimum tradeoff ratio within the module. For estimation economic optimum from a set of NFR, iso-cost graphs by assuming the constant cost. Some hypothetical variations in cost are also examined using 3D iso-cost graph. The reason to measure these tradeoff is to make design decision more empirical and helpful for the selection of design patterns, especially secure design patterns.

INDEX TERMS Document annotations, nonfunctional requirements, SRS, requirement engineering, trade-off measurements, production possibility graphs, iso-cost graphs, 3D iso-cost graphs.

I. INTRODUCTION

A software requirements specification SRS describes all the requirements system which must have for success. These requirements are typically illustrating features of underdevelopment system. These features not only describe its functional requirements FR but also its nonfunctional requirements NFR. Functional requirements as well nonfunctional requirements are equally significant for successful development. Both FR and NFR have an impact on the overall success of any system. In contrast to FRs which are described deliberately in SRS, NFRs are not described explicitly. If some NFRs are described deliberately, there are multiple NFRs implicitly defined hidden in plain text [1]. Encoding software requirements from the earliest period of development are more focused on functional requirements.

The associate editor coordinating the review of this manuscript and approving it for publication was Jiafeng Xie.

Non-functional requirements either neglected or described as a whole i.e., not for individual design problem [2]. These requirements artifacts are encoded without any prescribed standards, it can be either structured, semi-structured or completely unstructured [3]. In order to learn these non-functional requirements, we have to understand the genre of requirements artifacts are organized such as Modules, submodule and finally atomic design problems. These design Problems are not defined in any structured way most of the time requirement encoders spent more time on functional requirements. These requirements in a textual form commonly describe functional requirements, however, implicitly define non-functional requirements as well [1]. The textual data for requirement extraction can be used for extraction of underlying functional requirements [4]. In order to learn these underlying functional requirements a standardized way of encoding, functional requirements must be developed. Requirements artifacts are encoded using the wording of end users i.e. interviews or the words of requirement from requirement inceptor i.e. owner, end users [5]. The objectives of this research are (a) Annotations of unstructured SRS documents in an automated fashion.

(b) Extractions and association of NFRs with individual functional requirement or design problem. (c) measurement of optimal tradeoff between them. Research questions are investigated as follows (i) What can be a generic structure of SRS documents and how it can be annotated. (ii) How can we extract NFRs with individual functional requirements or individual design problems? (iii) How to learn tradeoff between these NFRs to associate an optimal set of NFR with each design problem. All in all, the motivation of this research is *"Extractions of NFRs from textual artifacts and their association in accordance with their mutual tradeoff."*

The methodology employed to investigate above-mentioned research questions includes Information extraction using PDFBox tool [6] and preparation of manual generic structured annotated dataset as a training dataset for classification. A manually annotated dataset of eight SRS document is labeled by using the General Architecture for Text Engineering (GATE) tool [7]. In order to annotate these SRS documents automated way, we employed a two-phase scheme i.e. text extraction and text classification. The text extraction process along with compositional information is carried out by using PDFBox tool [8]. In order to automate the annotation process, multi-pass sieve framework is used. The output of this step is annotated set of design problems which will serve as input text for further steps. Next step is the extraction of nonfunctional requirements from individual design problems. NFR extractions from the text is a widely researched area. Ameller [9], Riaz *et al.* [1], Alkussayer *et al.* [10], Vlas and Robinson [11], and Lee [12] used multiple techniques for NFR extractions from the text. These techniques can be categorized into two types (i) Ontological extractions [13]–[16] (ii) Text classifications or tagging [17]. We used text classification scheme for NFR extractions, different machine learning classifiers are analyzed. For time being we consider seven nonfunctional requirements i.e. Confidentiality, integrity, accountability, availability, manageability, performance and cost. We classify these requirements into two classes i.e. security triad (confidentiality, integrity, accountability) and performance triad & cost. Different machine learning classifier used for text classification can be used for NFR extractions such SVM. KNN and Naïve Bayes. We examined the efficiency of these classifiers and found Multinomial Naïve Bayes (MNB) more suitable for NFRs extractions. In order to learn the tradeoff between performance and security requirements tradeoff, we examined multiple different tradeoff measures i.e. Production possibility graph (PPC), tradeoff ratio, 2D iso-cost graph, and 3D iso-cost graphs. The reason to learn tradeoff among them is to an optimum set of NFR for each module and the whole system.

The contribution of this research is providing a method to annotate SRS documents automatically, followed by extractions and associations of NFRs. Lastly, we classify NFRs into two classes i.e. Security triad and performance triad and a method is devised to calculate tradeoff among these NFRs classes. This tradeoff will aid to determine the optimum set of NFR to be associated with the design phase. Eventually, make design decision more empirical and accurate.

The rest of the paper is arranged as background, literature review, methodology followed by conclusion and future work.

II. BACKGROUND

A specification for software requirements (SRS) contains all system requirements. These are typically separated into functional requirements FR, which describe the features of the system under development, and non-functional requirements NFR, which include, among other things, quality attributes, design constraints. It is well known that NFRs have a significant impact on the overall cost and time of the process of system development, as they often describe cross-cutting concerns [18]. In order to improve support for software development, an automated analysis of SRS documents for various NFR types is necessary.

Software Requirements Specification (SRS) documents are probably the most significant artifacts in the software development process. The effectiveness of SRS directly impacts the success of the project. It contains all functional requirements FR and nonfunctional requirements NFR. Both FRs and NFRs are equally significant for successful development. Functional requirements are deliberately defined, higher attention of requirements encoding are spent on FRs. NFRs are either neglected or encoded as the whole for all functionalities, not for individual functionalities. In order to capture these nonfunctional requirements, NFRs implicitly defined in text, some natural language processing (NLP) techniques are required.

A number of researchers used the technique to capture nonfunctional requirements from requirements artifacts [1], [8], [9]. In general, all above cited NLP techniques authors used multiple datasets and machine learning techniques to capture NFRs. Some datasets of requirements artifacts are described in a section below.

1) *PROMISE Dataset.*

The PROMISE corpus was compiled at DePaul University, a repository of 15 projects of MSc students. This dataset has labeled 326 NFRs and 358 FR. NFRs are labeled into 9 different categories. This dataset is available as a repository of sentences labeled with NFR impact. This dataset is considered generic enough to trace all NFR consideration. Initially, authors used machine learning techniques with higher recall up to 81% but low precision 0.124. The performance of classifier improved by using a naïve Bayes algorithm with higher recall and relatively low precision solution [20]. Another effort for NFR classification by using SVM with linear kernel [21]. Finally, Riaz *et al.* [1] used the same dataset with tool-assisted approach named as "NFR locator" which employs k-NN classifier. Along with PROMISE dataset they also used open source healthcare projects consist of seven requirement documents.

2) *John Slanks Dataset of open source project.*

As mentioned earlier Slankas *et al.* [22] compiled a new dataset of open source publically available healthcare projects requirements documents. One advantage of this repository is it consist of single domain healthcare projects. In contrast to PROMISE dataset in which NFRs are categorized into 9 different categories, this dataset is labeled against only five categories related to security and management. The same dataset used by Ferrari *et al.* [23] for extraction of security requirements from plain text. They have employed a tool based technique known as "NFR Discoverer". They produced high precision and recall results.

3) *Alesso Ferrari Dataset of publically available artifacts.*

Recently a new dataset was compiled, in which publically available requirements are compiled. It consist of 79 documents. This repository has both industry and academic projects developed from 1999 to 2011 [23]. There are structured, semi-structured and unstructured documents. Recently this dataset is used for pattern selection [17]. In their information retrieval approach authors manually annotate the text to reach individual functional requirements. In the proposed system we used above-mentioned dataset's chunk. The detailed dataset is explained in section A of methodology. In this project, we employed an automated way text extraction and annotation of text from PDF file format. The detail of text extraction and annotation is described in section B and C of methodology respectively.

A. DOCUMENT ANNOTATIONS

Most of the artifacts are in pdf, doc, RTF and HTML formats. However, PDF is most commonly used, which is based on the PostScript language, each document file encapsulates a complete description of a fixed-layout flat document, including the text, fonts, vector graphics, raster images and other information needed to display it. Information extraction from flat files is itself another field of research such as automated literature review and information extraction from medical reports etc. There are multiple ways used to extract information but most of them begin with annotations of documents. We have used PDFBox for text extractions and GATE tool for annotations of these documents.

B. ASSOCIATION OF NON-FUNCTIONAL REQUIREMENTS WITH DESIGN PROBLEMS

By applying a multi-pass sieve algorithm now we have classified text snippets. Text classified with the label of modules and further derived into paragraphs, these paragraphs are assumed as design problems. These design problems represent functional requirements explicitly and nonfunctional requirements implicitly. These design problem collectively represent a module and collection of module represents a system. By understanding the fact each design problem is disjoint in its functional requirement FR but NFRs associated with it have an impact on the overall system.

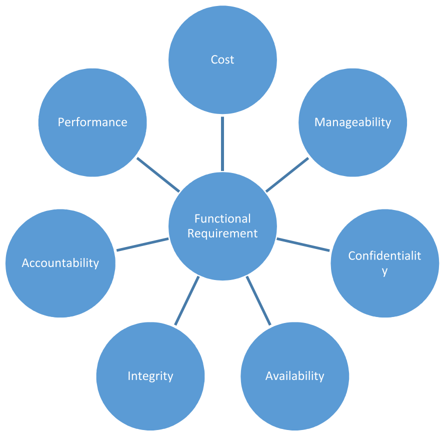


FIGURE 1. Nonfunctional requirements [26].

The nonfunctional requirements considered in this research are as follows, Accountability, Availability, Confidentiality, Integrity, Manageability, Usability, Performance, Cost. Their generic definition of these NFR is as follows. These requirements are not explicitly defined; these requirements are hidden in the plain text. The association of NFR for each FR is given in figure1. There is a tradeoff between these NFR i.e. if authorization is increased, there will eventually decrease the availability, cost, and manageability also. In an earlier study of secure design patterns, the tradeoff between these nonfunctional requirements tradeoff has been discussed [24]. A labeled corpus of these secure design patterns in which each design pattern is labeled against multiple security requirements tradeoffs [25]. However, these NFR varies FRs to FRs i.e. some FRs require less performance and high availability meanwhile it also requires security tradeoff too. A binary scheme for different tradeoff against each functional requirements is designed by us. Following figure 1 shows seven functional requirements are taken from a project PDF MERGE Sort in which their NFR are categorized required or not.

C. TRADEOFF MEASUREMENT

The performance triad-cost metric and security allow us to quantitatively determine how much security triad can be provided and how much performance will be reduced by implementing security metric. Hence, we can make a tradeoff between these two security and performance-cost metrics within the boundary of system's both functional requirements and nonfunctional requirements. The tradeoff between security triad and performance-cost triad as any measures taken to improve security triad will affect the performance triad. For example, a window is restricted from user's due to authenticity which eventually reduces the availability of the system. This tradeoff ratio varies module to module within a system. As some modules require more performance than security, on the other hand, some modules may require more security than performance metric [27]. This tradeoff can be calculated for formulating a utility function between both these two metrics as follows: Let us suppose PC represents Performance-cost triad and S represents security triad. The utility function between these metrics in equation 1.

$$U_{max} = w_1\,(PC) + w_2(S) \tag{1}$$

where w_1 and w_2 represent weights such as $w_1 + w_2 = 1$ and U_{max} represents utility function between these two metrics. With this utility function U_{max}, the tradeoff value can be calculated for individual design problem and the overall system. However, this tradeoff ratio decision between performance-cost has an impact on other design problems and multiple design problem resides within an interface and within a module. To calculate an optimal tradeoff ratio, we can use either a production possibility graph and iso-cost surface to calculate an economic optimum for tradeoff decision.

LINEAR PRODUCTION POSSIBILITY GRAPH

The production possibility curve is a hypothetical depiction of the amount of two different items that can be obtained by relocating resources from one to another. The slope depicts the choice between two different goods. This curve is commonly used for depiction of choice between two items such as consumption and investment. On the other hand, iso-cost or Isoquant curve is used for indications of various combinations of two or more factors with the same productions or output. The slope of such isoquant graphs represents the effectiveness of items. These graphs depict tradeoff and economic optimum. This optimum can be used further making decisions related to the right mix among multiple items.

III. LITERATURE REVIEW

The pioneer and notable effort to extract non-functional requirements from software artifacts are J. Slanks and L. Williams 2013 [28]. In their approach, they used available open source and publically available project documentation for extraction of nonfunctional requirements. In their work, they categorized nonfunctional requirements into 13 different categories. They used a tool based approach known as NFR locator for locating critical nonfunctional requirements. In their another work another dataset was used with labeled sentences, 10963 sentences are classified in 9 different categories with yes/no impact [1] However, their approach was restricted to health care domain and their approach classifies sentences rather than design problem [1], [17], [26]. Another approach using the same tool and same dataset to extract security requirement by Riaz *et al.* [29] in 2016. Another sentence classification technique to extract the nonfunctional requirement is developed by Mahmud and Williams [30] in 2016. The sentences used for classification are taken from different source code comments. They used thematic clustering technique to cluster nonfunctional requirements with 90% precision and 92% recall. Another approach for classifying sentence of comments is used by Azami Sheba in 2018 in her master's thesis. She employed an information retrieval model to extract nonfunctional requirements (citation pending). All the above-mentioned NFR extraction techniques extract at sentence level classification provides information regarding implications of NFR in the text. But the intent of these NFRs can't be learned from these techniques. In order to learn the intent/significance of certain NFR, requirements should be extracted from the individual design problem. In order to extract NFR from textual artifacts at the level of individual design problems text must be extracted with compositional detail.

IV. METHODOLOGY

The methodology employed is based on three phases. It begins with corpus creation from annotated SRS files. The second phase is the extraction of nonfunctional requirements NFR. The third phase is the measurement of tradeoff among NFRs. These phases can be described in the following figure 2.

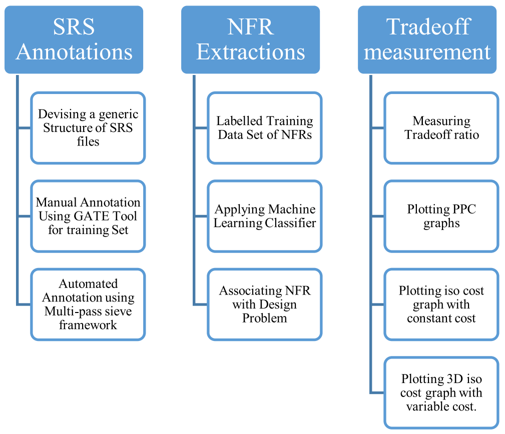

SRS Annotations
Devising a generic Structure of SRS files
Manual Annotation Using GATE Tool for training Set
Automated Annotation using Multi-pass sieve framework
NFR Extractions
Labelled Training Data Set of NFRs
Applying Machine Learning Classifier
Associating NFR with Design Problem
Tradeoff measurement
Measuring Tradeoff ratio
Plotting PPC graphs
Plotting iso cost graph with constant cost
Plotting 3D iso cost graph with variable cost.

FIGURE 2. Proposed methodology.

A. CORPUS OF SRS FILES

In order to extract nonfunctional requirements, we need a repository of software requirements specification. Compiling requirement artifact is tedious as most of the real-world

TABLE 1. SRS files in the repository.

Sr. No	Project Name	Explanation.	Structure	Year	Format
1	Gparted	partition editor GUI	Semi-structured	2010	.pdf
2	Opensg	Information Management framework	Structured	2011	.doc
3	split merge	PDF Manipulation	Structured	2010	.pdf
4	fishing	Fishing vessel logbook	Structured	2010	.pdf
5	Mashboot	website to monitor company presence in social network	Structured	2010	.pdf
6	model manager	research task management	Structured	2009	.pdf
7	Library	library system	Structured	2009	.pdf
8	PeaZip	Zip Tool	Structured	2009	.pdf

projects are not publically available [23]. A repository of fifteen software requirements specification was initially compiled at DePaul University [20]. This repository consists of only student's/ academics projects. Another repository of 79 requirements artifacts including both academic and industry projects [23]. However, these artifacts contain just increments or revision required and most of the projects SRS files are encoded at least 10 years earlier. Therefore, we developed criteria for choosing SRS files as follows. (i) Documents must be encoded between 2009 to 2011. (ii) SRS must be encoded for the first version. (iii) It must be in PDF format otherwise it should be converted into PDF format without losing compositional detail. (iv) SRS should be in a structured or semi-structured format. Unstructured documents are not included because we need composition detail to reach atomic design problem (Impossible for a document without any formatting rule.) Following the conditions of above-mentioned criteria documents shown in Table 1 are requirement used for further research experiments.

B. ANNOTATION OF PDF ARTIFACTS

1) TEXT EXTRACTION FROM PDF DOCUMENTS

There are multiple automated techniques for recognition of structure for a formal document in PDF or other formats. These techniques can be divided into two categories; machine learning techniques [13], [14] and the others are heuristics techniques [15], [16]. These methods use different ways to recognize the structure e.g. Title, Headings etc. An alternative and evaluated method used in for biomedical journal [33]. There are multiple formats of textual artifacts but for sake simplicity, we have taken PDF format only. The Portable Document Format (PDF) is a file format that is used to present data in a way that is independent of platform, hardware, and operating systems. Each PDF file holds textual data, compositional description and other information needed to display it. There are multiple examples of these libraries such as Jasper Reports, iText, Formatting Objects Processor, Adobe PDF Library and PDFBox Library. Apache PDFBox [8] is a commonly used open-source Java library which provides multiple options including development, conversion, and manipulations of PDF documents. With the help of this library, you can develop Java programs that create, convert and manipulate PDF documents. PDFBox library has multiple utilities for creation and manipulation textual data and graphics available in Jar format. It also provides multiple functionalities such as extract text, split and merge text, fill forms, print, save as image, create pdf, and signing. Software requirement specification is in different formats i.e. docs, HTML, and other formats. The very first step is the conversion of all SRS files into a single PDF format without losing its compositional structure and metadata. For the sake of extraction of text from these PDF documents, we have used this technique which employs the PDFBox tool for extraction of text from requirements artifacts. PDFBox is an open source tool widely used for multiple NLP Solution related to PDF. PDFBox performed well for extracting without losing any information and maintain the original order of the text. The initial extraction process is as follows: (i)Load document. (ii) Remove pages (iii)Extract text. (iv)Apply the multi-pass sieve algorithm.

2) LEARNING A LOGICAL STRUCTURE OF ARTIFACTS

Unfortunately, requirements specification documents don't follow any standards for encoding requirements [23]. However, a widely used scheme of encoding has the following structure Scope/Purpose, Introduction and description modules which include different functional requirements. Some of software requirements specification documents also include security requirements and legal implications. This generic structure commonly devised for the very first version of any system i.e. system inception. For the later versions, the SRS files are unstructured in nature or it does describe its modular structure. A generic solution of SRS documents is given in Table 2.

This structure can be visualized as in the figure 3, although this is not a standard structure of the document. But most of the SRS are encoded using this scheme or structure. However, there are multiple unstructured requirements documents are available which follow none scheme or structure to document them.

C. CORPUS CREATION

In order to build a corpus, a raw text which is extracted using PDFBox tool is further annotated using GATE annotation tool. The annotation of text scheme includes the following categories; scope, introduction, modules, security requirements. In order to learn atomic design problem, we leverage a technique in which modules portion is broken down into functional requirements by using the fact each requirements description either number, different font style or different case from rest of text as shown in figure 4. Division of modules portion of text can be trivial meanwhile the structure of

TABLE 2. Generic structure of SRS files.

Sr. No.	*Heading Name*	*Description.*
1	*Scope/Purpose*	*Usually a starting paragraph with slightly different heading names.*
2	*Introduction.*	*Introduction of SRS files*
3	*Modules*	*Usually, requirements are breakdown into modules. The idea of capturing module is heading along with a series of numbers. Each SRS artifact has a set of modules i.e. module1 to module n.*
4	*Functional Requirements.*	*Each module has a set of functional requirements can be captured from lists such as 3.1.. 3.2 etc.*
5	*Atomic design problem.*	*Each functional requirements may have single or multiple design problems for developers to meet.*
6	*NFR.*	*NFRs are sometimes encoded and mostly neglected*[22]

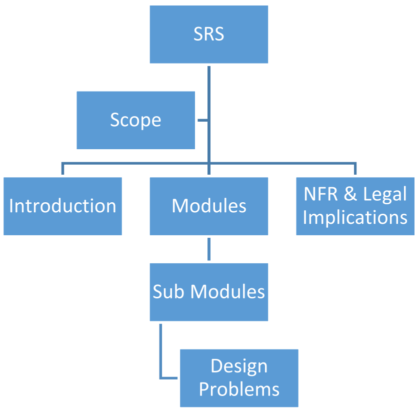


FIGURE 3. Generic structure of SRS files.

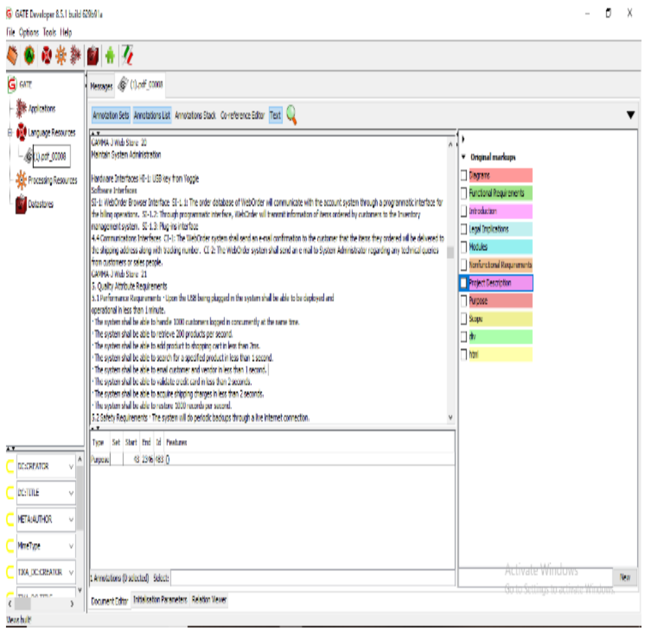
textual artifacts may vary. In order to automate this annotation process.

we employed a Multi-pass sieve text snippet classification scheme. The detail of this method is given in the following section.

MULTI-PASS SIEVE TEXT CLASSIFICATION

For automated classification, a multi-pass sieve framework is used. This framework categorizes text into Purpose, Introduction, Modules, and NFR classes. This framework provides a solution which is widely used for cross solve co-reference resolution problems. This framework performed well for learning text snippets for medical reports and research papers [6].

The advantage of using this framework based on multiple independent sieves applying on documents several times to perform the classification meanwhile other machine learning techniques classify in a single pass. The multi-pass sieve model's flexibility of breaking down the complex tasks into



GAMMA-J Web Store 20
Maintain System Administration

Hardware Interfaces HI-1: USB key from Yoggie
Software Interfaces
SI-1: WebOrder Browser Interface SI-1.1: The order database of WebOrder will communicate with the account system through a programmatic interface for the billing operations. SI-1.2: Through programmatic interface, WebOrder will transmit information of items ordered by customers to the Inventory management system. SI-1.3: Plug-ins interface
4.4 Communications Interfaces CI-1: The WebOrder system shall send an e-mail confirmation to the customer that the items they ordered will be delivered to the shipping address along with tracking number. CI-2: The WebOrder system shall send an e mail to System Administrator regarding any technical queries from customers or sales people.
GAMMA-J Web Store 21
5. Quality Attribute Requirements
5.1 Performance Requirements · Upon the USB being plugged in the system shall be able to be deployed and operational in less than 1 minute.
· The system shall be able to handle 1000 customers logged in concurrently at the same time.
· The system shall be able to retrieve 200 products per second.
· The system shall be able to add product to shopping cart in less than 2ms.
· The system shall be able to search for a specified product in less than 1 second.
· The system shall be able to email customer and vendor in less than 1 second.
· The system shall be able to validate credit card in less than 2 seconds.
· The system shall be able to acquire shipping charges in less than 2 seconds.
· The system shall be able to restore 1000 records per second.
5.2 Safety Requirements · The system will do periodic backups through a live Internet connection.

Type	Set	Start	End	Id	Features
Purpose		43	2396	483	{}

1 Annotations (0 selected) Select:

FIGURE 4. Manual annotations using GATE tool.

multiple sieves. Each sieve is specific to the special task, which can be conveniently tested, debugged and precision can be controlled. In order to adapt the Multi-pass sieve algorithm for classification of text snippets we used four configurations as follows, +Begin Condition, +Pass Condition, +Stop Condition, +Directionality Condition, and +Repetition Condition. These compositional rules are used to extract not only text but also compositional information from extracted text. The compositional rules are given in Table 3.

TABLE 3. Compositional rules for sieves.

Sieve Conditions.	*Explanation.*
+Begin Condition	*If one composing rule met.*
+Pass Condition	*Pass condition exceeds the size of a sieve.*
+Stop Condition	*Stop condition limits the size of a sieve.*
+Directionality Condition	*This condition begins guides the sieve to move up or down.*
+Repetition Condition	*This condition determines the number of repetitions of sieve* *A sieve can repeat any numbers.*

The detailed descriptions of these sieve condition in accordance to SRS documents is given in Table 4.

Now, we have at least six text snippets i.e. Scope, Purpose, Introduction, Modules, NFR and Legal Implications suitable for extractions of nonfunctional requirements. In order to move forward for NFR extraction, we classify these snippets classify them into two classes Intent and Functional requirement class. Modules snippets should be annotated in order to reach individual atomic design problems. These vectors modules contain multiple design problems, these design problem extractions from module snippet are discussed. The intuition to extract atomic design problems is to associate NFRs with each design problem to find out the optimal

TABLE 4. Compositional rules (CR) for annotating sieves.

Snippet Heading	*Sieve*	*Compositional Rules*
Purpose	*Begin Condition*	***RI*** *Begin with heading Purpose.* ***R2*** *Repeat Unlimited.*
	Pass Condition	***R3*** *Same Paragraph with the previous line.* ***R4*** *Next Paragraph without heading*
	Stop Condition	***R5*** *Fail to pass condition.* ***R6*** *Next Heading starts*
	Directional/ Repeat Condition.	***R6*** *Repeat in both directions* ***R2*** *Repeat Unlimited.*
Scope	*Begin Condition*	***R7*** *Begin with heading Scope* ***R2*** *Repeat Unlimited.*
	Pass Condition	***R3*** *Same Paragraph with the previous line.* ***R4*** *Next Paragraph without heading*
	Stop Condition	***R5*** *Fail to pass condition.* ***R6 Next*** *Heading starts*
	Directional/ Repeat Condition.	***R6*** *Repeat in both directions* ***R2*** *Repeat Unlimited.*
Introduction	*Begin Condition*	***R8*** *Begin with heading Scope* ***R2*** *Repeat Unlimited.*
	Pass Condition	***R3*** *Same Paragraph with the previous line.* ***R4*** *Next Paragraph without heading*
	Stop Condition	***R5*** *Fail to pass condition.* ***R6 Next*** *Heading starts*
	Directional/ Repeat Condition.	***R6*** *Repeat in both directions* ***R2*** *Repeat Unlimited.*
Module	*Begin Condition*	***R9*** *Begin of heading with large font* ***AND*** ***R10*** *Begin similar with introduction heading in composition information.* ***R2*** *Repeat Unlimited*
	Pass Condition	***R3*** *Same Paragraph with the previous line.* ***R4*** *Next Paragraph without heading*
	Stop Condition	***R5*** *Fail to pass condition.* ***R6 Next*** *Heading starts*
	Directional/ Repeat Condition.	***R6*** *Repeat in both directions* ***R2*** *Repeat Unlimited.*
NFR	*Begin Condition*	***R11*** *Begin of heading Nonfunctional Requirements* ***AND*** ***R12*** *Both R9 and R10 condition already met.* ***R2*** *Repeat Unlimited*

tradeoffs among them. The association is of nonfunctional requirements described in the following section. Variables used to automatically classify text snippets are given in Table 5.

We have implemented and tested a true representative set of algorithms for machine learning. algorithm for the multi-pass sieve. Multiple machine learning algorithms implemented This study used and integrated in the WEKA data mining software (39). The support for this Vector Machine algorithm with sequential minimal optimization (SMO) The kernel radial base function (RBF) has been tested on different exponents, gamma values and Values c hyper-parameter. We tested Naïve Bayes, J48 and logistic regression as well as Algorithms with default settings. We have developed a model for machine learning in Table 2. In addition to the word

TABLE 5. Variables used and their description.

Variable	*Type*	*Description*
Snippet_length.	*Numeric*	*Length of text snippet in number of characters*
Paragraph_no.	*Numeric*	*Paragraph position number*
Page_no.	*Numeric*	*Page number*
IsPrimaryFont	*Boolean*	*Whether the text snippet uses the most prominent font in the document*
font_size	*Numeric*	*A font size of the text snippet*
containInIntroduction	*Boolean*	*Whether the snippet is contained in the title*
containInScope	*Boolean*	*Whether the snippet is contained in the abstract*
containInModules	*Boolean*	*Whether the snippet is before section breaks*
containInNFR	*Boolean*	*Whether the snippet is had NFR keyword*
containInLegal	*Boolean*	*Whether the snippet is had legal keyword*
numModule	*Numeric*	*Module Number*
NumDP	*Numeric*	*Number of DESIGN PROBLEM entities*
numNFR	*Numeric*	*Number of NFR paragraphs entities*
Bag-Of-Words features	*Numeric*	*Frequencies of each word DTM*

bag characteristics, length of text snippets are large which generates a scalability issue. A 16 GB RAM is not sufficient for these variables however, we used each document a single classifier and treated each document individually.

D. EVALUATIONS OF THE RESULTS OF ANNOTATIONS

A hypothesis "Classifications of PDF documents having no predefined format for encoding using a multi-pass sieve framework would be accurate and efficient. Another hypothesis is either metadata is helpful for classification of snippets or not. In order to evaluate, we randomly split data into 50-50%. But half of the snippets are annotated and using machine learning classifier, and multi-pass sieve framework. Since it's a novelty in the field of requirement engineering there is no such work exists in the literature. However, the performance of this work can be compared in another discipline such as annotation of medical reports and research papers. We compared the results with similar techniques used for systematic literature review (SLR) in the field of medical research.

Results shown in the Table 6 are slightly lower than results shown by [6] due to a number of reasons as follows (i) Research papers are well structured and rich in compositional information. (ii) Compositional rules for research papers are more concrete as they have to follow rigid

TABLE 6. Comparison of performance with medical reports.

	Medical Reports		SRS files	
No of documents	42		31	
No of classes for text snippets	5		8	
File format considered	PDF only		PDF along with doc and HTML version converted into PDF.	
Nature	Well Structured		Both Structured and unstructured.	
	Logistic Regression	Multi-pass sieve	Logistic Regression	Multi-pass sieve
Accuracy	82.9	92.9	78.7	89.1

structured documents. (iii) Software development companies don't follow any generic structure for documenting SRS files. Therefore, most of SRS vary in their structure. (iv) Lack of compositional information as online SRS files are available in multiple file formats. Some formats don't save metadata of text files.

E. NONFUNCTIONAL REQUIREMENTS EXTRACTION FROM PLAIN TEXT USING MACHINE LEARNING

NFR extractions begin with corpus creation in which annotated snippets are considered as individual document i.e. corpus consists of sentences related to design problem. Next step is data cleaning since data is captured unstructured and there is a universally accepted format of SRS files a lot of unwanted noise is included in text snippets. In order to remove this noise, following steps are taken which includes a change to lower case, removal of numbers, removal of whitespace, removal of stop words, removal of symbols and special character removal. The next step is stemming; it is a process of reducing words to their base or root. We used wordStem from R library which uses Dr. Martin Porter's stemming algorithm for reduction of dimensions. Although stemming reduces dimensions, due to a number of sparsity terms dimensions are further reduced using singular value decomposition (SVD). SVD provides terms aggregations widely used for input with large word count. It takes a vector $m \times n$ where m represents documents n represents terms and produced an orthogonal matrix of eigenvalues of order $\times r$. After removal of noise and dimensions reductions, the next step is feature extraction by using most common term frequency-inverse document frequency (TF-IDF) for further classification. TF-IDF is a way used for determination of the significance of a word in a document, widely used for feature extractions and information retrieval systems such as a Vector space model (VSM).

Preparation of training set. These design problems are not labeled and it is quite difficult to label these design problems against different NFR. In order to cope with the requirement of the training set, we used dataset publically available for used for NFR extractions. The dataset has 11 software requirements documents related to the healthcare domain This dataset is used for both training and testing of the model and further employed for design problems NFR extractions. The training set is subjected for the same procedure of preprocessing and feature extractions except for SVD. Classifications. Finally, for classification different classifiers can be used for classifying NFRs such as Support vector machine (SVM) [cite], K-Nearest Neighbors (kNN) and Naïve Bayes[cite]. According to [1], [2], [21], and [26] Naïve Bayes perform better for nonfunctional requirements extractions we use Naïve Bayes and SVM and kNN and found Naïve Bayes suitable for nonfunctional requirements extractions. The NFR extraction procedure is given in figure 5.

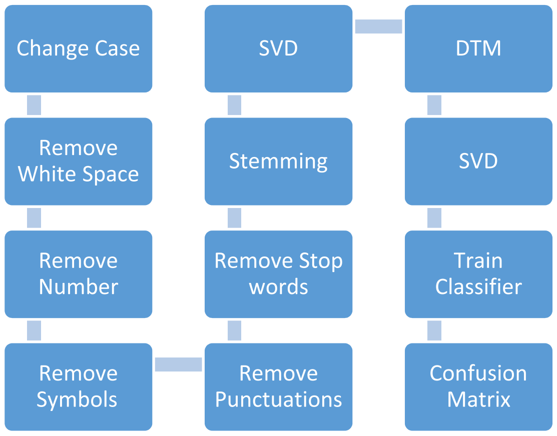


FIGURE 5. Text classification scheme.

Different algorithms are used for further classifications of design problems as are as follows

(a) Multinomial Naïve Bayes (MNB)

From the family Naïve Bayes, Multinomial Naïve Bayes (MNB) [35] is commonly used for text classifications. The main advantage of using Naïve Bayes it performs well when tagged data is a small i.e. a couple of thousands labeled sentences, and it also requires less computational resources. Naïve Bayes classifier is based on Naïve Bayes theorem. It computes the conditional probability of occurrence of two events based on their individual probability of occurrence. In the context of text classification, each vector of text contains a probability of occurrence of an individual word. Which will eventually compute the likelihood of a vector class?

(b) Support vector machine SVM

Support vector machine SVM one of many algorithms for classification can be chosen. It is suitable where training data is scarce. On the other hand, it requires more computational resource. It achieves accurate results in different machine learning problem. A linear SVM draws a line or hyperplane which divides space into two spaces [36]. This line classifies vectors in accordance with the tagging of the training set. It is widely used for ham or spam classification examples.

(c) K- Nearest Neighbor Classifier.

kNN is a classifier which measures the similarity between vectors to classify vectors. kNN can be used for nonfunctional requirements extraction from plain text [9]. The kNN classifier is based on the idea that an instance categorization is quite similar to the classification of other instances in the vicinity of the vector. Especially, when compared to other text classification techniques, such as the Bayesian classifier, kNN does not depend on previous probabilities and is computationally efficient. The main calculation is the category of training documents to find the closest neighbors to the test document.

(b)Evaluations of performance.

In order to evaluate results, confusion matrix are used in which four parameters are described as true positive (TP), false positive (FP), true negative (TN) and false negative (FN). The confusion matrix is used to describe Accuracy, precision, Recall or sensitivity, Error rate, Prevalence, Null Error Rate, Cohen's Kappa, F-Score and ROC curve parameters for evaluations of a classifier.

The accuracy of classification or correctness of a classification can be defined as

$$\textit{Accuracy} = \frac{\textit{True Positive} + \textit{True Negative}}{\textit{Total} + \textit{Predictions}} \quad (2)$$

Does accuracy describe how often classifier correctly? **Error rateorsensitivity describes how often classifier predict falsely as**

$$\textit{error rate} = \frac{\textit{False Position} + \textit{False Negative}}{\textit{Total Prediction}}. \quad (3)$$

Another parameter is specificity, also known as True positive rate can be defined as the ratio between actual false and predicted false.

$$\textbf{Specificity} = \frac{\textit{Predicted false}}{\textit{actual false}}. \quad (4)$$

Another significant parameter is precision which can be defined as how often it is "True" when it is predicted "True" i.e.

$$\textit{precision} = \frac{\textit{True Position}}{\textit{True}} \quad (5)$$

Prevalence defines how often True occurs in a dataset i.e.

$$\textit{precision} = \frac{\textit{True Position}}{\textit{Total observations}} \quad (6)$$

Null Error Rate determines how often would be wrong if it predicted the majority class. This parameter serves in calculations of Cohen's Kappa another parameter for evaluation which describes accuracy and null error rate. F-Score and ROC curve are others evaluation parameters widely used for information retrieval systems. The performance of all three classifiers is discussed in Table 6.

Association schema for NFR based on classifier predictions for each design problem in terms of yes or no.

TABLE 7. Performance of classifier.

Evaluation metric	SVM	Naïve Bayes	kNN
Accuracy	0.93641	0.983	0.965
Specificity	0.983	0.92	0.91
Precision	0.017	0.22	0.33
Prevalence	0.78	0.44	0.42
Null Error Rate	0.022	0.65	0.62
Cohen's Kappa	0.32	0.55	0.63

These NFR association can be described as an input of the design problem, the output will be yes or no for each NFR parameter. The percentage of NFR associations with design problems is shown in figure 6.

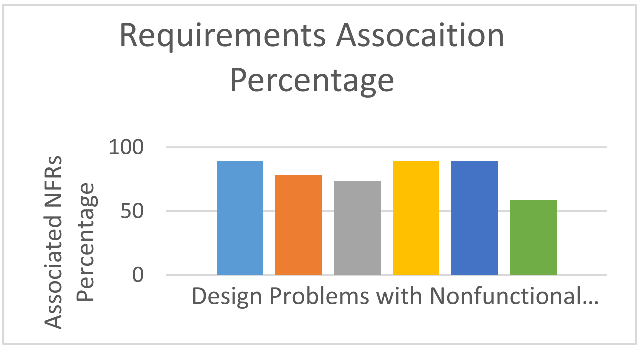


FIGURE 6. Percentage of associations.

Percentage of different NFRs describes by above histograms for the percentage of design problems associated with NFRs.

In order to measure tradeoff between them, these requirements are categorized into two classes i.e. Security related requirements Confidentiality, Integrity and Accountability (CIA) and performance related Availability, Cost, Performance. Then these classes production possibility curve is used commonly in economics for optimal productivity, is used for measurement of optimal tradeoff among them.

F. MEASUREMENT OF TRADEOFFS BETWEEN SECURITY AND PERFORMANCE

We classified extracted nonfunctional requirements from each design problem in a problem. Further, these requirements are associated with each design problem in binary format i.e. from PDF_Merge_Sort project Merge_PDF functional requirements have no confidentiality, integrity, and accountability are not required but availability and performance are required. The association scheme "0" for not required and "1" for required. For understanding the structure of the project can be visualized as the relation of FR and their associated NFR as $\textit{Module} = \{(a, b) | a \in \textit{set of FRs} \wedge b \in \textit{set of NFRs}\}$. The impact of each FR on any other FR can be determined associated NFR with it. The impact of nonfunctional requirements NFR on the other can be described by classifying into two classes Security and Performance& Manageability. Security classes contain

CIA tirades and performance & manageability class includes availability, performance, manageability, and accountability.

Return of NFR not chosen and return of NFR chosen cannot be calculated but we can assume it in binary format i.e. by assuming it each security requirement will definitely affect performance class NFRS i.e. cost, availability or performance. This opportunity cost for each functional requirements can be used for plotting production possibility graph for identifying an optimal set of nonfunctional requirements suitable to be employed for each module of the system. Which eventually aid to choose appropriate design patterns, architecture design. The calculations of opportunity cost graph for the different project are given below.

G. PPC GRAPH FOR PROJECT PDFsam

It is an online available pdf utility along with publically available documentation [37]. It has six modules and 34 functional requirements. We extracted nonfunctional requirements associated with each functional requirements and PPC graph is calculated as given below Measurement of PPC tradeoff among CIA triad and performance measurements. CIA Tirades Performance Measure Confidentiality, Integrity, Accountability, Availability Performance Manageability Cost association as shown in Table 8.

TABLE 8. Classes of NFRs.

FR Id	*CIA Triads*			*Performance Triad & Cost*			
0	C	I	Acc.	Av.	P	M	C
1	0	1	1	1	1	0	0
2	1	1	1	0	0	0	0
3	0	1	1	0	1	1	1
4	0	0	0	1	0	0	0
5	1	1	1	0	1	1	0
6	1	1	0	0	0	0	0
7	0	1	1	1	1	1	1

Consider a measure taken for confidentiality of information, it will decrease the availability of the functional requirement, integrity will decrease performances and manageability. All CIA triad will affect cost. There is a tradeoff between these nonfunctional requirements classified CIA and Performance NFRs. In order to measure the tradeoff ratio among a module to choose optimal NFR for a module is given below. These PPC graphs reflect optimal tradeoff among these NFRs. Another approach for representing NFR tradeoff is by using iso-cost graph. Iso-cost graphs depict the tradeoff ratio especially suitable when one variable remains constant during a change. There is a tradeoff between these two classes can be calculated as a microeconomics concept "Opportunity cost". Opportunity cost in context to these NFRs can be defined as what will be gained in terms of security at which cost in performance. The formula for gain in NFR can be calculated as given in equation 6.

$$\begin{aligned}&\textbf{\textit{opportunity cost}}\\&\quad = \textbf{\textit{return of nfr not chosen}} - \textbf{\textit{return of nfr chosen}} \qquad (7)\end{aligned}$$

In order to learn opportunity-cost between the tradeoff these NFR classes, we have examined all designs problems within a module. These design problems serve as data points for plotting PPC graph for each module of the system. PPC graph for each module of the system provides the optimum tradeoff value between NFRs associated with each design problems and eventually choose appropriate design patterns and more empirical design decisions. These PPC curves help the designer for making a better design decision. As these curves provide information about tradeoff among design problem within a module. This tradeoff measure is plotted as production possibility PPC graph for reflecting optimal set NFRs for the modules as shown in figure 7 and 8.

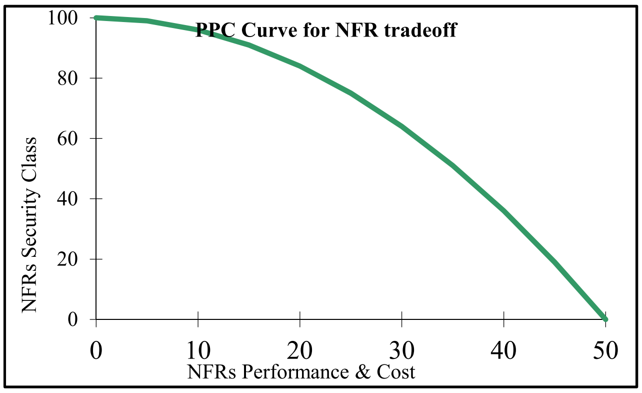


FIGURE 7. PPC Curve for PDFsam project.

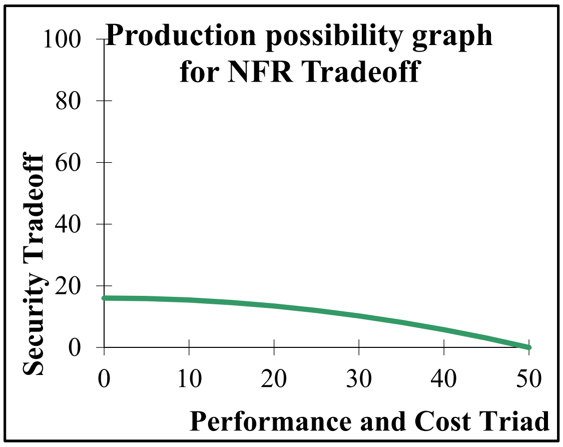


FIGURE 8. PPC Curve for PDF SamProject.

ISO-COST GRAPH

This iso-cost graph hypothetically defines optimal tradeoff ratio between two classes i.e. ΔNFR_S *and* ΔNFR_P. This ratio provides information regarding change in the NFRs set and their impact on other NFRs within a module. This change can be described in the equation 8.

$$\frac{\Delta NFR_S}{\Delta NFR_P} = \frac{\mathit{Changein\ NFR_s\ for\ security\ if\ choosen}}{\mathit{Change\ in\ NFR_p\ for\ performance\ if\ chosen}}$$

where NFR_s *represents Nonfunctional requirements from CIA Triad and* NFR_p *represents from Performance triad and cost*. In the limit, $NFR_s - NFR_p$ tradeoff ratio represents derivative ratio $d(NFR_s)/d(NFR_p)$, further calculations are based on assumptions that all NFRs have similar cost and their production cost is also assumed identical. Their optimum value can be calculated as

$$\frac{\Delta NFR_S}{\Delta NFR_P} = \frac{P_{NFR_s}}{P_{NFR_p}} \tag{8}$$

a tradeoff finally the value of Productivity of NFR is represented as productivity of security tradeoffs over other nonfunctional requirements can be derived from the equation as in Equation 9.

$$P_{NFR_s} = \frac{\Delta NFR_S}{\Delta NFR_P} \times P_{NFR_p} \tag{9}$$

The this pa tradeoff is described by using 2D iso-cost graph.

This productivity measure provides a competitive advantage of security requirements over other nonfunctional requirements. The productivity of all individual requirements is considered identical for all NFRs so the value of P_{NFR_p} will be constant. So the productivity is depending upon the ratio of change in NFRs i.e. ΔNFR_S and ΔNFR_P. These changes are computable only if we extract all these NFRs from individual design problems and measure relative change among all design problems. This relative change is computed using by the sum of Security and Performance & Cost class triads. As shown in 2D iso-cost Figure 9.

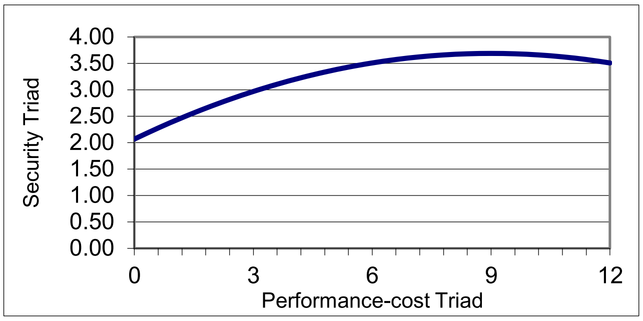


FIGURE 9. 2D iso-cost graph.

This hypothetical while assuming cost is constant. However, taking cost as a variable then the iso-cost surface would be third dimensions' graphs as shown in figure 10.

This three-dimensional iso-cost plane describes the economic optimum for a design problem. However, the accuracy of such plots highly depends upon the estimated cost for each NFR which is practically impossible. However, this plot provides information related to Two classes and variations in cost. As can be seen the peak of the 3D curve in term of security have a higher cost.

Although these curves provide information regarding the trends in a tradeoff of design patterns. But still, improvements are required in the field of NFR extractions and annotations

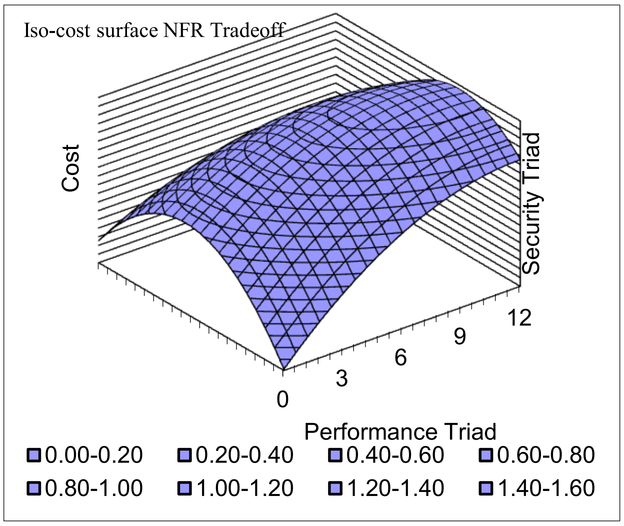


FIGURE 10. 3D iso-cost curve for NFR tradeoff.

V. CONCLUSION AND FUTURE WORK

In this study, a gold standard for the annotation of software requirement specifications SRS has been developed. The annotation is further evaluated for with an accuracy of up to 78% for unstructured documents and for structured documents, it is 89%. Further, we have extracted nonfunctional requirements from these requirements and provide a method for calculating optimal tradeoff between different NFRs which aid in design patterns selections and improve the success factor of a system. A limitation of this work the tradeoff is measured as a discrete figure but it fluctuates indifferently. In future work, we will improve the accuracy of the annotation scheme for SRS and evolution of tradeoffs calculation among different nonfunctional requirements, the tradeoff measurements provide more consideration related to NFRs and their impact. It provides earlier information regarding apply a security metric impact on other NFRs such as availability, cost, and manageability. This research also provides information to achieve an optimum regarding the tradeoff among NFRs. Although it is difficult to predict the relative impact of any NFR metric to others. In this research, we consider equal impact i.e. binary format Yes or No. However, measurement of NFR impact on other NFR metrics at an earlier stage will ensure design decision more empirical and accurate. As this research provides a basic idea to calculate the impact of NFR between two classes i.e. Security triad and performance triad. Although the impact of each NFR over other NFR might be different, the impact with security triad is positive i.e. a metric provides accountability also increase confidentiality similar is the case within performance triad. But the impact among both classes is negative. Increase in metric decrease the other i.e. a metric for confidentiality will decrease availability. Therefore, the tradeoff can only be calculated between these two classes. In the future, we will design an interface which accepts SRS files and provides the user economic optimum set of associated

NFRs for making better design decisions. This optimum set of NFR will be based on calculations of tradeoffs among these NFRS. Although this research includes only a single open source application with the least security requirements. This interface will serve as an information retrieval system for design patterns selections especially in the field of selection of secure design patterns.

REFERENCES

[1] M. Riaz, J. King, J. Slankas, L. Williams, and N. Carolina, "Hidden in plain sight: Automatically identifying security requirements from natural language artifacts," in *Proc. IEEE 22nd Int. Requirements Eng. Conf. (RE)*, Aug. 2014, pp. 183–192.

[2] M. Rahimi, M. Mirakhorli, and J. Cleland-Huang, "Automated extraction and visualization of quality concerns from requirements specifications," in *Proc. IEEE 22nd Int. Requirements Eng. Conf. (RE)*, Aug. 2014, pp. 253–262, Aug. 2014.

[3] A. Ferrari, G. O. Spagnolo, and S. Gnesi, "Towards a dataset for natural language requirements processing," in *Proc. CEUR Workshop*, 2017, p. 1796.

[4] X. Xiao, A. Paradkar, S. Thummalapenta, and T. Xie, "Automated extraction of security policies from natural-language software documents," *Proc. ACM SIGSOFT 20th Int. Symp. Found. Softw. Eng.*, Nov. 2012, p. 12.

[5] I. Hussain, L. Kosseim, and O. Ormandjieva, "Using linguistic knowledge to classify non-functional requirements in SRS documents," in *Proc. Int. Conf. Appl. Natural Lang. Inf. Syst.*, Jun. 2008, pp. 287–298.

[6] D. Duc, A. Bui, G. Del Fiol, and S. Jonnalagadda, "PDF text classification to leverage information extraction from publication reports," *J. Biomed. Inform.*, vol. 61, pp. 141–148, Jun. 2016.

[7] *GATE.ac.uk—Index.html*. Accessed: Jan. 11, 2019. [Online]. Available: https://gate.ac.uk/

[8] *Apache PDFBox | A Java PDF Library*. Accessed: Jan. 4, 2019]. [Online]. Available: https://pdfbox.apache.org/

[9] D. Ameller, C. Ayala, J. Cabot, and X. Franch, "Non-functional requirements in architectural decision making," *IEEE Softw.*, vol. 30, no. 2, pp. 61–67, Mar. 2013.

[10] A. Alkussayer, "Towards a secure software development framework based on an integrated engineering process," *Florida Inst. Technol.*, vol. 5, Aug. 2011, Art. no. 3441945.

[11] R. E. Vlas and W. N. Robinson, "Two rule-based natural language strategies for requirements discovery and classification in open source software development projects," *J. Manag. Inf. Syst.*, vol. 28, no. 4, pp. 11–38, Apr. 2012.

[12] J. Lee, "Recommendation systems," *Big Data Comput. Intell. Netw.*, to be published.

[13] A. Souag, R. Mazo, C. Salinesi, and I. Comyn-Wattiau, "Reusable knowledge in security requirements engineering: A systematic mapping study," *Requir. Eng.*, vol. 21, no. 2, pp. 251–283, Jun. 2016.

[14] P. Velasco-Elizondo, R. Marín-Piña, S. Vazquez-Reyes, A. Mora-Soto, and J. Mejia, "Knowledge representation and information extraction for analysing architectural patterns," *Sci. Comput. Program.*, vol. 121, pp. 176–189, Jun. 2016.

[15] A. Souag, C. Salinesi, I. Wattiau, and S. Requirements, "Ontologies for security requirements: A literature survey and classification," in *Proc. Int. Conf. Adv. Inf. Syst. Eng.*, 2012, pp. 61–69.

[16] L. Tóth and L. Vidács, "Study of various classifiers for identification and classification of non-functional requirements," in *Proc. Int. Conf. Comput. Sci. Its Appl.*, 2018, pp. 492–503.

[17] I. Ali, M. Asif, M. Shahbaz, A. Khalid, M. Rehman, and A. Guergachi, "text categorization approach for secure design pattern selection using software requirement specification," *IEEE Access*, vol. 6, pp. 73928–73939, 2018.

[18] A. Rashwan, O. Ormandjieva, and R. Witte, "Ontology-based classification of non-functional requirements in software specifications: A new corpus and SVM-based classifier," in *Proc. Int. Comput. Softw. Appl. Conf.*, Jul. 2013, pp. 381–386.

[19] A. Patra and D. Singh, "A survey report on text classification with different term weighing methods and comparison between classification algorithms," *Int. J. Comput. Appl.*, vol. 73, p. 7, Jan. 2013.

[20] J. Cleland-Huang, R. Settimi, X. Zou, and P. Solc, "Automated classification of non-functional requirements," *Requir. Eng.*, vol. 12, no. 2, pp. 103–120, Apr. 2007.

[21] A. Motii, B. Hamid, A. Lanusse, and J.-M. Bruel, "Guiding the selection of security patterns based on security requirements and pattern classification," in *Proc. 20th Eur. Conf. Pattern Lang. Programs*, vol. 15, Jul. 2015, pp. 1–17.

[22] J. Slankas, L. Williams, and N. Carolina, "Automated extraction of non-functional requirements in available documentation," in *Proc. 1st Int. Workshop Natural Lang. Anal. Softw. Eng. (NaturaLiSE)*, May 2013, pp. 9–16.

[23] A. Ferrari, G. O. Spagnolo, and S. Gnesi, "Towards a dataset for natural language requirements processing," in *Proc. REFSQ Workshops*, 2017, pp. 52–62.

[24] M. Schumacher, E. Fernandez-Buglioni, D. Hybertson, F. Buschmann, and P. Sommerlad, *Security Patterns: Integrating Security and Systems Engineering*. Washington, DC, USA: DARPA, 2006.

[25] D. M. Kienzle, M. C. Elder, D. Tyree, and J. Edwards-Hewitt, *Security Patterns Repository Version 1.0*. Washington, DC, USA: DARPA, 2002.

[26] M. Riaz and L. Williams, "Security requirements patterns: Understanding the science behind the art of pattern writing," in *Proc. 2nd IEEE Int. Workshop Requirements Patterns (RePa)*, Sep. 2012, pp. 29–34.

[27] W. Zeng and M. Y. Chow, "A trade-off model for performance and security in secured networked control systems," in *Proc. ISIE*, Jun. 2011, pp. 1997–2002.

[28] J. Slankas and L. Williams, "Automated extraction of non-functional requirements in available documentation," in *Proc. 1st Int. Work. Nat. Lang. Anal. Softw. Eng. Nat.*, May 2013, pp. 9–16.

[29] M. Riaz, J. Stallings, M. P. Singh, J. Slankas, and L. Williams, "DIGS - A Framework for Discovering Goals for Security Requirements Engineering," in *Proc. 10th ACM/IEEE Int. Symp. Empirical Softw. Eng. Meas.*, Sep. 2016, p. 35.

[30] A. Mahmoud and G. Williams, "Detecting, classifying, and tracing non-functional software requirements," *Requir. Eng.*, vol. 21, no. 3, pp. 357–381, May 2016.

[31] R. Kern, K. Jack, and M. Hristakeva, "TeamBeam—Meta-data extraction from scientific literature," *D-Lib Mag.*, vol. 18, nos. 7–8, Jul. 2012.

[32] S. Klampfl, M. Granitzer, K. Jack, and R. Kern, "Unsupervised document structure analysis of digital scientific articles," *Int. J. Digit. Libr.*, vol. 14, nos. 3–4, pp. 83–99, Aug. 2014.

[33] S. M. Meystre, G. K. Savova, K. C. Kipper-Schuler, and J. F. Hurdle, "Extracting information from textual documents in the electronic health record: A review of recent research," *IMIA Yearb. Med. Informat. Methods Inf. Med.*, vol. 47, no. 1, pp. 44–128, 2008.

[34] A. K. Uysal and S. Gunal, "The impact of preprocessing on text classification," *Inf. Process. Manag.*, vol. 50, no. 1, pp. 104–112, Jan. 2014.

[35] A. M. Kibriya, E. Frank, B. Pfahringer, and G. Holmes, *Multinomial Naive Bayes for Text Categorization Revisited*. Berlin, Germany: Springer, 2004, pp. 488–499.

[36] R. Setiawan, "Performance comparison comparison and and optimization optimization of of text text document document classification using nave bayes classification class," *Procedia Comput. Sci.*, vol. 116, pp. 107–112, Aug. 2017.

[37] (2018). *PDF SAM*. [Online]. Available: https://pdfsam.org/documentation/

MUHAMMAD ASIF received the M.S. and Ph.D. degrees from the Asian Institute of Technology (AIT), Thailand, in 2009 and 2012, respectively, through the HEC foreign Scholarship, where he was a Research Scholar with the Computer Science and Information Management Department. He is currently the Chair of the Department of Computer Science, National Textile University, Faisalabad. During the course of time, he was a Visiting Researcher with the National Institute of Information Tokyo, Japan. He was involved in some projects including the Air Traffic Control System of Pakistan Air force. He is also a permanent member of Punjab Public Service Commission (PPSC), as an Advisor, and a Program Evaluator at the National Computing Education Accreditation Council (NCEAC), Islamabad. He is serving as a Reviewer for a number of reputed journals. He has authored a number of research papers in reputed journals and conferences. He is serving as Associate Editor for the IEEE Access, the prestigious journal of the IEEE.

ISHFAQ ALI is a Research Scholar with National Textile University, Faisalabad. He has authored many top journal and conference papers. He is currently involved in the domain of security patterns and their usage in different kinds of applications.

MUHAMAD SHERAZ ARSHED MALIK received the Ph.D. degree in information technology from Universiti Teknologi PETRONAS Malaysia. He is currently an Assistant Professor with the Department of Information Technology, Government College University Faisalabad, Pakistan. His research interests include information visualization, big data, data analytics, and the IoT future trends.

MUHAMMAD HASANAIN CHAUDARY received the Ph.D. degree from the Asian Institute of Technology, Thailand, in 2012. He has several years of teaching and research experience. He is currently an Assistant Professor with the COMSATS University Islamabad at Lahore Campus, Pakistan.

SHAHZADI TAYYABA received the bachelor's degree in computer engineering and the master's degree in computer science and engineering from UET Lahore, Pakistan, in 2006 and 2008, respectively, and the Ph.D. degree in engineering in microelectronics and embedded systems from the School of Engineering and Technology, Asian Institute of Technology (AIT), Bangkok, Thailand, in 2013. She is currently the Head of the Department of Computer Engineering, The University of Lahore, Pakistan. She has published over 100 research articles in reputed international journals and peer reviewed conferences. Her research interests include simulation and modeling, MEMS, NEMS, biomedical engineering, nanotechnology, material processing, and fuzzy logic and control systems.

MUHAMMAD TARIQ MAHMOOD (M'12–SM'16) received the M.S. degree in computer science from the Blekinge Institute of Technology, Sweden, in 2006, and the Ph.D. degree in informatics and mechatronics from the Gwangju Institute of Science and Technology, South Korea, in 2011. He is currently an Assistant Professor with the School of Computer Science and Engineering, Korea University of Technology and Education. His research interests include image processing, machine learning, and computer vision.