How to cite this paper:

Noor Hasrina Bakar, Zarinah M. Kasirun, Norsaremah Salleh, & Azni H. Halim. (2017). Extracting software features from online reviews to demonstrate requirements reuse in software engineering in Zulikha, J. & N. H. Zakaria (Eds.), Proceedings of the 6th International Conference of Computing & Informatics (pp 184-190). Sintok: School of Computing.

EXTRACTING SOFTWARE FEATURES FROM ONLINE REVIEWS TO DEMONSTRATE REQUIREMENTS REUSE IN SOFTWARE ENGINEERING

Noor Hasrina Bakar[1], Zarinah M. Kasirun[2], Norsaremah Salleh[3] and Azni H. Halim[4]

[1]*Universiti Kebangsaan Malaysia*, noorhasrina@ukm.edu.my,
[2]*University of Malaya*, zarinahmk@um.edu.my
[3]*International Islamic University of Malaysia*, norsaremah@iium.edu.my
[4] *University Sains Islam Malaysia*, ahazni@usim.edu.my

ABSTRACT. Software Product Lines Engineering is a systematic approach towards realizing software reuse. Among important artifacts to be reused includes requirements, architectures, source codes, designs or even test plans. Requirements reuse if done systematically can expedite the time to market, improve productivity and reduce laborious work. This paper presents the results from an experiment conducted on extracting features from online software reviews to demonstrate our proposed solution to requirements reuse problem. Fifty two software reviews from seven software categories are used as the test data. The automated process proposed is compared to the manual process and the results from experiment indicates significant improvements in terms of time efficiency and F-Measure.

Keywords: requirements reuse, software reviews, software product lines, software engineering

INTRODUCTION

Reuse of software artifacts such as requirements, architectures, designs, codes, and test plans can produce many benefits including reducing development costs, increasing developers' productivity, and expediting time to market [1] [2] [3] [4]. Software requirements can be reused either in an ad hoc basis such as in clone and own applications, software maintenance, or when systematically planned in Software Product Lines Engineering (SPLE). However, many problems exist when dealing with ad hoc reuse of natural language (NL) requirements. The problems with manual requirements reuse include arduous [5] costly [6], error-prone [7], and labour-intensive process [8], especially when dealing with large requirements. The majority of requirements are written in NL [9]. This is because text is commonly used to convey information to communicate stakeholders' needs [6]. Pohl et al. [10] emphasized that in SPLE, software requirements are documented either by using NL or model-based. As an example, NL requirements do not only appear in the form of Software Requirements Specification (SRS) format. In this research, we propose a Feature Extraction Approach from Natural Language Requirements, FENL to aid the requirements reuse process. We present the related works in Section 2. In section 3 we present the proposed FENL approach and we present the results in Section 4 of this paper and finally conclude the paper.

RELATED WORKS

Various works in the area of requirements reuse were published. For example, in [5], the authors described ArboCraft as a tool that can automatically process textual requirements into feature models, that can later be refined by the requirements engineers. This approach uses the Latent Semantic Analysis, LSA to group similar features. In-text variability was identified through a tool that detected uncommon words. Requirements were considered similar if they concerned similar matters. Thus, in ArboCraft, the subject matters of requirements were compared, resulting in similar subject matters to be clustered together. The GUI representation of ArboCraft was presented to illustrate the feature tree construction resulting from the feature extraction.

In [6], the functional requirements in each document were identified on the basis of lexical affinities and "verb-direct object" relations [6] and [12]. Fillmore's case theory was used to characterize each Functional Requirements Profile's (FRP) semantics. A verb followed by an object in a requirement sentence would be extracted as a FRP. The authors defined the FRP of a document to be the domain-aware Lexical Affinity, LA that has a high information value and bears a verb-direct object relation. Fillmore's case theory was applied to each FRP, by filling up the details for six semantic cases. Then, Orthogonal Variability Modelling, OVM was used to rigorously express the variability. Mu et al. in [13] improved Nan Niu's FRP by proposing ten semantic cases instead of just six, naming it as Extended Functional Requirements Framework (EFRF). The extractions were done based on the structure of EFRF. The extraction process came in two phases: NLP and rule-based converting process. OVM and SRS were also used in this work. Similarly, the text preprocessing technique was also highlighted in [7] to identify common features in product brochures from various vendors and also used when mining specifications for typical antivirus products in [14].

As for grouping related requirements, work in [14] used data mining approach to find common features across products and also relationships among those features. An Incremental Diffusive Clustering, IDC algorithm, was used to extract features from online product listings. Association mining was applied together with k-nearest neighbour machine learning method to anlyse the relationships among features and make recommendations during the domain analysis process. The end results were a set of recommended features, which could be supplied to the requirements engineering process to help project stakeholders to define features for specific product lines. Chen et al. in [15] manually constructed requirements relationship graph from various requirements specification documents. Hierarchical clustering was also used in their work to merge requirements into feature trees. Unfortunately, the paper did not provide a detailed description on how this is obtained. Furthermore, their approach required heavy manual human involvement. We have published a detailed systematic literature review for features extraction from natural language in [16]. In this paper, we will firstly propose a process model for feature extraction approach and later demonstrate how it can be implemented.

FENL APPROACH

One of the main implications of the Systematic Literature Review conducted in [16] is the process model for an overall approach of Requirements Reuse, as illustrated in Figure 1:

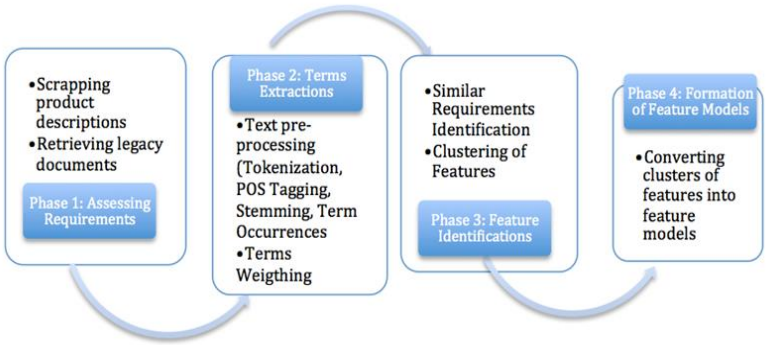

Figure 1. Feature Extraction Approach for Requirements Reuse.

FENL is separated into four main phases, with the first three phases to be automated. (Due to space limitation, this paper will only report the findings up until Phase 3). The FENL offers to extract software features from various forms of requirements, such as online software reviews, legacy requirements or product descriptions by using NL processing, and Information Retrieval techniques. However, for the experiment conducted in this research, only the freely available software reviews from the Internet are used. The reviews are selected from toptenreviews.com websites that provide a compilation of software reviews by experts. These reviews are also beneficial for developers (and domain analysts) who did not have access to SRS and can use these expert reviews as a source for identifying features for the product they want to build without having to initiate the RE process from scratch (reuse of requirements). In toptenreviews.com, software are reviewed by experts (more formal, less bias and with less noise) and compiled periodically as compared to the first-hand review data sets used in the related works [17], [18] and [19].

Phase 1 seeks for software reviews available on the Internet as an alternative to using SRS documents. To demonstrate this, 52 software reviews pertaining to various software products posted in toptenreviews.com are extracted. They came from seven categories as follows: PL1: Preschool Learning (10 compilations), PL2: Algebra Learning (10 compilations), PL3: Language and Reading Software (3 compilations), PL4: Creative Writing (9 compilations), PL5: Vacation Management Software (10 compilations), PL6: Social Networking Site (5 compilations) and PL7: Online Storage Service (5 compilations).

The documents being scraped in Phase 1 is now used as the input to the automated terms extraction process. Figure 2 lists out the process used for the terms extraction. Steps 1 until 4 in Fig. 2 are repeated for all selected reviews.

Step 1: Text pre-processing to remove the stop-words, punctuations, numbers, and special characters.
Step 2: Apply WordNet Lemmatization[1]
Step 3: Apply the Part of Speech Tagging from NLTK[2].
Step 4: Construct term-document-matrix.

Figure 2. Steps in Phase 2.

A final spreadsheet contains ***n***-terms by ***m***-documents (terms-document matrix, where ***n*** represents number of unique terms and ***m*** represents the number of documents). Based on the terms collected, the term weights are calculated by using the term frequency inverse document frequency. This is the weight used in IR and text mining to evaluate how important a word is to a document in a collection. For this case, the spreadsheets comprising collection of terms from various documents are merged and the terms occurring within each document can be clearly seen. The main outputs for Phase 2 are important terms (verbs and nouns) in each document and their occurrences.

In Phase 3, we identify the similar documents, followed by extraction and grouping of similar software features. We have used K-Means and latent Semantic Analysis to group similar documents together (similar product categories). When dealing with unstructured documents such as software reviews, measuring sentence similarities is not easily achieved. This is because reviews were written freely and did not follow sentence structure such as sentences

[1] http://textanalysisonline.com/nltk-wordnet-lemmatizer
[2] http://www.nltk.org/

that exist in SRS. To cater to this, we have applied the natural language processing (NLTK from Python) to extract the features from software reviews. Combinations for parts of speech are used with three different kind of configurations as follows:

```
cfg = {} #Simple Tagging(Verb-direct object)
cfg["VB+NNP"] = "NNP"

cfg = {} #NP Only (Noun Phrase Extraction)
cfg["NN+NN"] = "NNI"
cfg["JJ+NN"] = "NNI"

cfg = {} #FENL (Hybrid)
cfg["VB+NN"] = "NNP"
cfg["NN+JJ+VB"] = "NNP"
cfg["NNP+NN"] = "NNI"
cfg["JJ+NN"] = "NNI"
```

**NN = Nouns, JJ = Adjectives, VB = Verb Base form*

The first configuration, labelled as Simple Tagging, extract the verbs and nouns only, similar to the related work [6] and [13] that focusing on Verb + Direct Object in the extraction of functional requirements profile of a software system. Meanwhile the second configuration, labelled as Noun Phrase Extraction (NP Only) applies the extraction approach similar to the work by [7] that uses nouns and adjectives which is believed to bring out the components of a software system. FENL takes the hybrid form of both approaches.

EXPERIMENTAL RESULT

The number of software reviews that has been used as input is 52, subdivided into 7 categories. Each document length ranges from 91 to 440 sentences, while the total word lists extracted from all 52 software reviews is 7451. (Note: same reviews were used by both Software Practitioner and teachers). Figure 3 indicates the comparison between manual and the three automated extraction approaches in terms of number of features extracted for all seven categories of reviews.

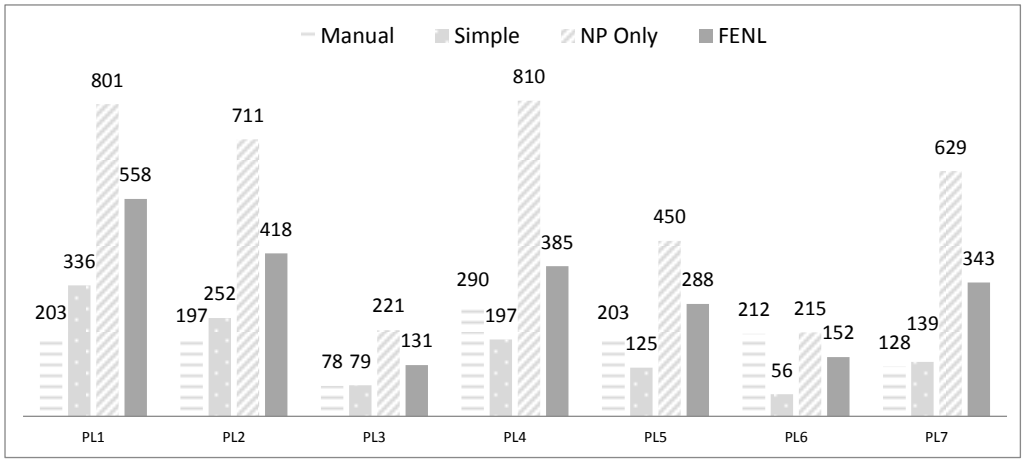

Figure 3. Number of features extracted by manual approach as compared to the automated approach.

NP Only produces the biggest number of features across all product lines. FENL performs steadily across all product lines (produces higher number of features if compared to manual and Simple tagging). Overall, it is observed that NP Only extract the highest number of features from the data set. From observing the NP Only results, not all the features are actually

relevant, as it contains some noises, for example, terms such as "small enterprise" and "possible customer" are noises and they did not represent software features. To compare the accuracy of the all extraction approach (manual and all three automated), the Recall, Precision and F-Measure are calculated based on the total features exist in the truth data set versus total features extracted by all of the approach. Figure 4 illustrates the average performance reported for the accuracy evaluation results. From F-Measure results obtained, FENL performs higher when compared to Simple and NP approach for all product lines except PL6 and PL7, thus making the average F-Measure for FENL approach to be at 66.61%, about 25% lower to manual approach. Benchmarking on the manual approach, the performance for FENL is superior when compared to the other two automated approaches.

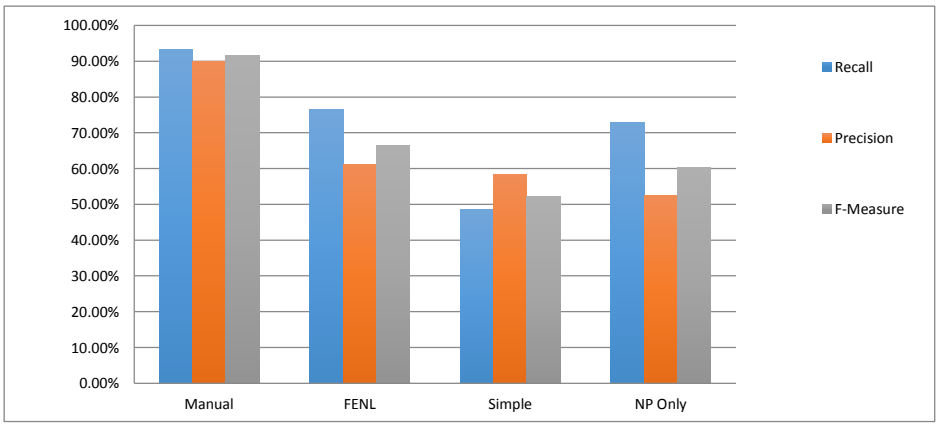

Figure 4. Average performance result.

To determine whether there was statistically significant difference between the means produced by different extraction methods, One Way ANOVA test has been conducted. Based on the sample data, there was statistically significant difference between groups as determined at $F(3,24)=13.873$, $p=0.000$ for Recall, at $F(3,24)=8.226$, $p=0.001$ for Precision and at $F(3,24)=12.987$, $p=0.000$ for F-Measure. Performance in terms of time efficiency, the time taken to complete the extraction process is also recorded, in which far better than time needed if done manually. In the experiment conducted, FENL recorded higher recall values, which indicates the relevant features that are finally selected. Although FENL extracted some noises (irrelevant items that is indicated by lower precision values), we note that there is an average of 76.59% of the relevant items which consists of actual features (recall). The average recall, precision and F-measure results obtained by the FENL in comparison with related works that uses similar evaluation measure is presented in Table 1.

There are three recent studies that reported similar evaluation results [19], [17] and [18]. Other related works were not included in this comparison either because their approach did not present the evaluation results in terms of Precision, Recall and F-Measure or they did not use the data set of similar nature, i.e. user reviews, thus comparison cannot be made). From Table 1, FENL reported a lower F-Measure when compared with [18] & [17] but performed slightly better if compared to Guzzman's work. This result tells us that FENL approach performed comparably with related works.

Table 1. FENL Versus Related Works. (Average Precision, Recall, and F-Measure)

Results (related works)	Precision	Recall	F-Measure
Guzzman (2014)	0.582	0.520	0.549
Carreno and Windbladh (2013)	0.941	0.67	0.782
Khan et al. (2014)	0.79	0.717	0.752
FENL	0.61	0.77	0.67

CONCLUSIONS

This paper describes the application of information retrieval technique and k-Means clustering for a requirements reuse problem in software engineering, demonstrated through FENL approach. The results obtained from FENL approach is validated by measuring the precision, recall, and F-measure. One Way ANOVA test via SPSS was applied to the average precision, recall and F-Measure to test for their significance. FENL when compared to manual approach indicates a significant improvement in terms of time efficiency and F-Measure.

ACKNOWLEDGMENTS

This work was supported in part by UMRG by University Malaya and USIM/FRGS/FST/32/51414.

REFERENCES

M. Eriksson, J. Borstler, and K. Borg, "Sotware Product Line Modeling Made Practical: An Example from Swedish Defense Industry," *Commun. ACM*, vol. 49, no. 12, pp. 49 – 53, 2006.

A. Monzon, "A Practical Approach to Requirements Reuse in Product Families of On-Board Systems," in *16th IEEE International Requirements Engineering Conference*, 2008, pp. 223–228.

B. Moros, A. Toval, F. Rosique, and P. Sánchez, "Transforming and Tracing Reused Requirements Models to Home Automation Models," *Inf. Softw. Technol.*, Dec. 2012.

A. Von Knethen, B. Paech, F. Kiedaisch, F. Houdek, D.- Kaiserslautern, D.- Ulm, and D. Ag, "Systematic Requirements Recycling through Abstraction and Traceability," in *Requirements Engineering*, 2002, pp. 273–281.

N. Weston, R. Chitchyan, and A. Rashid, "A Framework for Constructing Semantically Composable Feature Models from Natural Language Requirements," in *Proceeding of the 13th Software Product Lines Conference*, 2009, pp. 211–220.

N. Niu and S. Easterbrook, "Extracting and Modeling Product Line Functional Requirements," *2008 16th IEEE Int. Requir. Eng. Conf.*, pp. 155–164, Sep. 2008.

A. Ferrari, G. O. Spagnolo, and F. Dell'Orletta, "Mining commonalities and variabilities from natural language documents," in *Proceedings of the 17th International Software Product Line Conference on - SPLC '13*, 2013, p. 116.

E. Boutkova and F. Houdek, "Semi-automatic identification of features in requirement specifications," in *2011 IEEE 19th International Requirements Engineering Conference*, 2011, pp. 313–318.

C. Denger, D. M. Berry, and E. Kamsties, "Higher Quality Requirements Specifications through Natural Language Patterns," in *Proceedings of the IEEE International Conference on Software—Science, Technology & Engineering (SwSTE'03)*, 2003, pp. 1–11.

V. Alves, C. Schwanninger, L. Barbosa, A. Rashid, P. Sawyer, P. Rayson, C. Pohl, and A. Rummler, "An Exploratory Study of Information Retrieval Techniques in Domain Analysis," *2008 12th Int. Softw. Prod. Line Conf.*, pp. 67–76, Sep. 2008.

J. Nicolás and A. Toval, "On the generation of requirements specifications from software engineering models: A systematic literature review," *Inf. Softw. Technol.*, vol. 51, no. 9, pp. 1291–1307, Sep. 2009.

N. Niu, J. Savolainen, Z. Niu, M. Jin, and J.-R. C. Cheng, "A Systems Approach to Product Line Requirements Reuse," *IEEE Syst. J.*, pp. 1–10, 2013.

Y. Mu, Y. Wang, and J. Guo, "Extracting Software Functional Requirements from Free Text Documents," in *2009 International Conference on Information and Multimedia Technology*, 2009, pp. 194–198.

J.-M. Davril, E. Delfosse, N. Hariri, M. Acher, J. Cleland-Huang, and P. Heymans, "Feature model extraction from large collections of informal product descriptions," *Proc. 2013 9th Jt. Meet. Found. Softw. Eng. - ESEC/FSE 2013*, p. 290, 2013.

K. Chen, W. Zhang, H. Zhao, and H. Mei, "An approach to constructing feature models based on requirements clustering," in *13th IEEE International Conference on Requirements Engineering (RE'05)*, 2005, pp. 31–40.

N. H. Bakar, Z. M. Kasirun, and N. Salleh, "Feature extraction approaches from natural language requirements for reuse in software product lines: A systematic literature review," *J. Syst. Softw.*, vol. 106, pp. 132–149, Aug. 2015.

K. Khan, B. Baharudin, and A. Khan, "Identifying Product Features from Customer Reviews Using Hybrid Patterns," *Int. Arab J. IT*, vol. 11, no. 3, pp. 281–286, 2014.

L. V. G. Carreno and K. Windbladh, "Analysis of User Comments : An Approach for Software Requirements Evolution," in *International Conference of Software Engineering, ICSE 2013*, 2013, pp. 582–591.

E. Guzman and W. Maalej, "How Do Users Like This Feature ? A Fine Grained Sentiment Analysis of App Reviews," in *Requirement Engineering Conference 2014*, 2014, pp. 153–162.