Variability Aware Requirements Reuse Analysis

Muhammad Abbas
muhammad.abbas@{mdh.se,ri.se}
Mälardalen University & RISE Research Institutes of Sweden
Västerås, Sweden

ABSTRACT

Problem: The goal of a software product line is to aid quick and quality delivery of software products, sharing common features. Effectively achieving the above-mentioned goals requires reuse analysis of the product line features. Existing requirements reuse analysis approaches are not focused on recommending product line features, that can be reused to realize new customer requirements.
Hypothesis: Given that the customer requirements are linked to product line features' description satisfying them: then the customer requirements can be clustered based on patterns and similarities, preserving the historic reuse information. New customer requirements can be evaluated against existing customer requirements and reuse of product line features can be recommended.
Contributions: We treated the problem of feature reuse analysis as a text classification problem at the requirements-level. We use Natural Language Processing and clustering to recommend reuse of features based on similarities and historic reuse information. The recommendations can be used to realize new customer requirements.

KEYWORDS

software reuse, variability, product line, requirements, similarities

ACM Reference Format:
Muhammad Abbas. 2020. Variability Aware Requirements Reuse Analysis. In *42nd International Conference on Software Engineering Companion (ICSE '20 Companion), May 23–29, 2020, Seoul, Republic of Korea.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3377812.3381399

1 PROBLEM

In the manufacturing industry, particularly in the railway industry, the software has to work with many possible hardware configurations (varying motors, signals and actuators). In many cases varying regional safety and regulatory requirements are also to be satisfied for a specific group of customers. Software Product Line (SPL/PL) is one possible solution to tackle variable customer requirements [14]. New customer requirements are realized by deriving a suitable product from the product line. This requires a reuse analysis to identify features that can be configured, modified or reused to satisfy the new customer requirements. Thus engineers and architects have to spend a significant amount of time reasoning about which product line feature(s) can be used to realize the new customer requirement(s). This also allows in keeping the product line relevant by avoiding redundancy and aiding a high degree of reuse. The manual process of reuse analysis makes the process heavily dependent on the experience of the engineers and is time-consuming.

Figure 1 shows the reuse analysis process in the context of a complex system's development, supported by a product line. The process is initiated when a new product (for a new customer) is to be derived from the product line. The PL assets repository shown in the figure contains highly configurable features that can be reused/configured/modified to satisfy varying customer requirements. Due to the safety-critical nature of the products, the process of the development and product derivation has to comply with safety standards (railway safety standard [1], in case of our industrial partner, Bombardier Transportation AB). Compliance with safety standards also requires traceability between requirements and other artifacts. Thus traceability links between customer requirements and the product line feature descriptions, realizing them, are also created. In most cases, inter-dependencies between requirements are also documented and can be used to guide later stages of development and testing [1, 2]. Recommender systems can use the existing structured requirements traceability information to recommend features for reuse. This will make the process of reuse analysis less time consuming and less dependent on the engineer's experience.

Existing requirements reuse approaches in the area of SPLs are not focused on recommending PL features for reuse. Most of the approaches aim at enumerating requirements (based on similarity) to extract features and their relationships [9, 18]. Other approaches for reuse (e.g, [12] or [13]) are also not focused on recommending the reuse between two different levels of abstractions, sharing a less semantic similarity (such as customer requirements and PL features description). Traceability link recovery approaches [7] are more focused towards general trace recovery rather than reuse. In our case, the customer requirements and the PL features description share a very low semantic and syntactic similarity. Existing approaches are only applicable if both levels of abstraction share a syntactic or semantic similarity. To the best of our knowledge, there is no variability-aware reuse analysis approach that aims to recommend reuse PL features by bridging the gap between customer requirements and product line features' description.

2 RESEARCH OBJECTIVES

Our main research objective is to aid the quick and quality delivery of the product through a recommender system. We achieve this by providing support to engineers in the product derivation

ICSE '20 Companion, May 23–29, 2020, Seoul, Republic of Korea

ACM ISBN 978-1-4503-7122-3/20/05…$15.00
https://doi.org/10.1145/3377812.3381399

[1] 2011. CENELEC - EN 50128 - Railway applications - Communication, signallingand processing systems - Software for railway control and protection systems. ,132 pages.

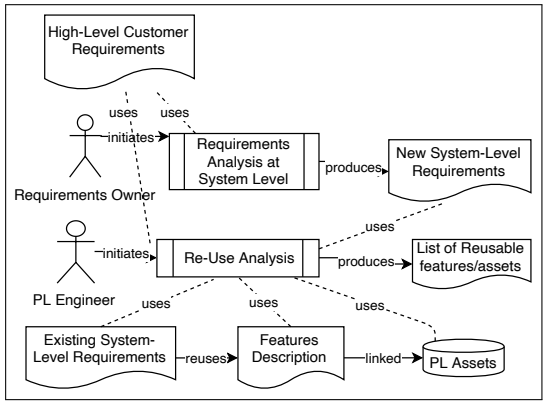

Figure 1: Reuse analysis process in context of SPL

from PL. Our approach is focused on recommending relevant PL features that can be reused to realize new and unseen customer requirement(s). Our main hypothesis is that the association rules between customer requirements and PL requirements can be learned, and the learned rules can be used to recommend the reuse of PL features. We propose to achieve this by vectorizing the customer requirements and then clustering the vectors based on feature similarities. Clustering requirements' vectors can help in finding the PL features that are common for reuse for a specific cluster. Historic reuse information can be used to recommend reuse of PL feature for unseen customer requirement(s). Association rules for other links can also be learned using the same approach. We also believe that the reuse analysis done with a recommender system can make the process less dependent on the experience of engineers.

On testing the hypothesis. We used one real project (an actual train) with historic reuse data. Due to confidentiality reasons, we cannot name the project or the region where it is deployed. A reuse analysis on the customer requirements was already performed and the PL features to be reused for each requirement were identified. As a first step, we applied a bag of words based Latent Dirichlet Allocation (LDA) [5] (topic modeling) to extract high-level topics from the customer requirements. We then verified if the requirements having common topics are realized by the same product line features. In many cases, the requirements sharing common topics were indeed realized by the same PL feature.

3 PROPOSED APPROACH

This section discusses each part of our approach (shown in Figure 2) in more detail. We also presented our preliminary results obtained from applying our approach to the requirements in Section 5.

1 Pre-Processing is done by first extracting requirements from the requirements management tool. We extracted customer requirements from the requirement specification document. We extracted requirements that had an outgoing link to at least one PL features' description. Approaches for identifying requirements in large documents do exist [3], but using this simple rule (should have an out-going link to PL features' description) we were able to correctly identify and extract customer requirements. We used spaCy [2] for cleaning the extracted requirements text. We removed all English stop words and besides, we removed a list of domain-specific stop words (e.g, "system"). We then rooted up the words to their lemmas. Lemmatization is important to avoid the treatment of similar terms differently. Note that spaCy also provides Named Entity Recognition (NER) and such information can also be very helpful for effective pre-processing and computation of behavioral similarity [4]. In the future, we aim to improve our pre-processing with the addition of NER.

2 Vectorizing (feature extraction) natural language requirements help in interpreting the text as numerical vectors. Vectorization is used as a way to extract numeric features from the raw text. The vectors can be obtained using two main approaches. The first class of approaches uses term frequencies to derive vectors and the second class of approaches uses deep neural networks for deriving a word or paragraph vector. The vectors obtained from term frequency-based approaches can have very high dimensions. Most of the features in the vectors from the frequency-based approaches can be redundant or co-related. Different approaches can be used to select a subset of the features from the vectors. We used Principal Component Analysis (PCA), which uses statistics to transform the values into uncorrelated vectors and thus helps in reducing the dimensions. We selected enough features to capture at least 95% of the variance in the data set. Currently, we have used Term-Frequency Inverse Document Frequency (TF-IDF) [16] followed by PCA and Google's Doc2Vec [11] for vectorizing the requirements. We will further discuss the results in more detail in this section. We aim to also focus on evaluating more vectorization methods for effective feature extraction and in reuse prediction. We aim to include Facebook's FastText [6] and Bidirectional Encoder Representations from Transformers (BERT) [8] in the dissertation.

3 Unsupervised Clustering can be used to classify the vectors based on the similarity of their features. Clustering is important to divide the prediction space into smaller chunks. Clusters can also be used to identify reuse outliers in a larger data set. We clustered the vectors in euclidean space preserving the historic reuse links using K-Means. We chose the number of clusters using the Elbow method (which uses explained variance for choosing the number of clusters). We also aim to include the evaluation of other clustering algorithms for our problem of reuse analysis. However, we believe that the clustering algorithm has no significant effect on the accuracy of the whole pipeline.

4 Recommendations are generated by predicting a cluster for vectors of unseen customer requirements. We compute the distance between the existing vectors (in the predicted cluster) and the new vectors. We retrieve at most five unique reused PL features from the closest neighbors of the new vectors (customer requirements). We then rank the list (of PL features) based on similarity in the vectors at the customer requirements level. The ranked list is produced as a final output to the end-users. Note that we are not considering the frequency of reuse of a PL feature at the moment but we are planning to include the reuse frequency in our ranking function.

4 EVALUATION PLAN

Evaluation is planned as a two-fold activity. First, we want to focus on the evaluation of the feature extraction methods in the context

[2] https://spacy.io/

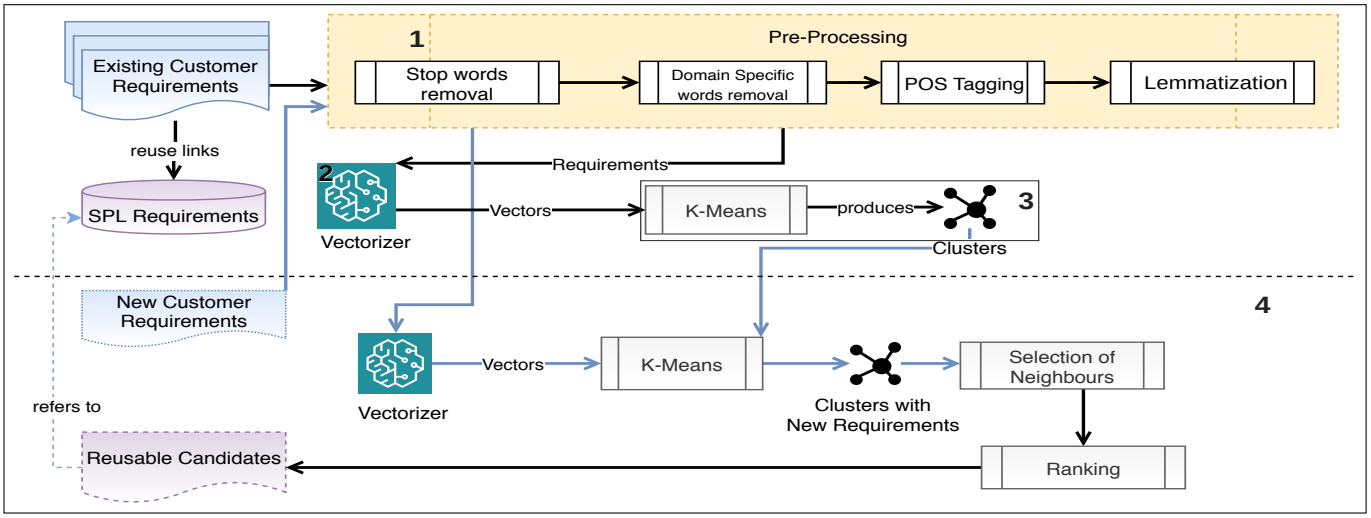

Figure 2: Proposed Product Line features' reuse recommender

Table 1: Pipelines considered for this study

Name	Pipeline
TF-IDF	`PreProcess()->TfidfVectorizer(min_df=6, max_df=0.5, ngram_range=(1,8))->PCA()-> KMeans()`
Doc2VecTraining	`PreProcess()->Doc2Vec(vector_size=300, min_count=2, epochs=10)->infer()-> KMeans()`
Doc2VecWiki300	`PreProcess()->Doc2VecWiki300.infer()-> KMeans()`

Table 2: Accuracy results for pipelines

Pipeline	Run 1	Run 2	Run 3	Average
TF-IDF	74.35	66.66	71.79	**70.93**
Doc2VecTraining	10.63	17.02	8.51	**12.05**
Doc2VecWiki300	61.70	63.82	59.57	**61.69**

of requirements-level reuse prediction. We aim to compare different vectorization methods for efficiency (in terms of time taken for vector generation), and effectiveness (precision and recall of the pipeline). This side of the evaluation will be using a controlled experiment, following the experiment design and reporting guidelines [10]. To prepare data for our experiment, we will use already derived products and their requirements with recorded reuse. The existing reuse links will be used as ground truth for the evaluation of the different pipelines.

The second side of the evaluation will be focused on reporting a case study (following guidelines [17]) on performing reuse analysis with recommender systems in the railway industry. The use of a case-study research method would help in studying the reuse analysis in real and in a natural environment.

5 PRELIMINARY RESULTS

We used one project for conducting our preliminary evaluation. The project had 159 customer requirements (already identified) reusing 44 distinct PL features. We randomly selected 75% of the requirements (120 out of 159) as a training set. The remaining 25% (39 in total) customer requirements and their reuse links were selected as a test set. In evaluation, the reuse links for the test set were used as ground truth. We prepossessed the customer requirements using the pre-process pipeline shown in 2. The pre-processed text obtained for the training set was then vectorized using TF-IDF, self-trained Doc2Vec, and Gensim Pre-Trained Doc2Vec. Vectors were obtained and clustered using K-Means. Table 1 shows all the three pipelines used to obtain the preliminary results. We used Gensim implementation of the Doc2Vec [15] and the pre-trained model is also part of the Gensim API. The trained/fitted models were then evaluated for accuracy on the test set. The test set was pre-processed using the same pre-process pipeline. The pre-processed text obtained from the test set was then vectorized using the resultant model, fitted/trained on the training set. The test set vectors were then clustered in the existing clusters and recommendations for PL features were generated. Recommendations were counted correct: if the recommendation produced by the pipeline contained the ground truth. We ran each pipeline three times with randomly selected training and testing sets and computed the accuracy for each run. We presented the accuracy results from each run in Table 2, shown in percentages.

Our preliminary results show that the term frequency based TF-IDF approach performed better on this dataset. TD-IDF pipeline was able to recommend PL features with an average of 70.93% accuracy. Because the examples fed into the learning process were not enough, we cannot comment on the power of our `Doc2VecTraining` pipeline.

The pre-trained neural network-based model performed significantly better than the self-trained model. We believe that using transfer learning on the pre-trained model can also improve the accuracy of the `Doc2Vec300Wiki` pipeline.

We also validated our preliminary results from experts in the industry. We showed the results obtained from the TD-IDF pipeline to three engineers in an informal interview. The engineers commented that our results are useful and insightful, as the first step to automated reuse analysis. Engineers also commented that such approaches could be very useful if deployed as a plugin in the requirements management tool, DOORS [3].

6 TIMELINE

We are planning to complete the work proposed in this study in one and a half years. As a first step, we started with evaluating requirements vectorization techniques. We developed the pipeline for reuse analysis based on term-document approach and also based on neural network based word embedding. We aim to extend our requirements level reuse analysis to more vectorizers and also build the basis for diversity based test selection during the first half-year. We also aim to train the neural network-based word embedding algorithms on domain-specific data.

In the last year, we will focus on building a tool for selecting the best policy for the PL feature's reuse prediction, based on the requirements repository. The approach will compute the precision and recall for each available vectorizer (on a subset of the dataset) and select the best vectorizer suited for the data. We also aim to evaluate the efficiency of our approach using a controlled experiment. We will ask a group of engineers to identify the reusable PL features for a set of new customer requirements and then we will ask another group to do the same with support from our tool. We will evaluate the time taken by each group and present our results.

7 CONCLUSIONS

We propose a reuse analysis approach for PL features. We formulated the reuse problem as text clustering and classification problem. Our approach is a first step towards bridging the gap between two different requirements abstraction levels. This is done by extracting meaningful vectors from the requirements text and clustering them in euclidean space, so that feature reuse recommendations can be generated. Feature reuse recommendations for unseen customer requirements are generated using the historic reuse of neighbors in the cluster. The same approach can also be used to recommend other dependencies among requirements, sharing a less semantic or syntactic similarity. Preliminary results show that recommending PL assets/features using similarities in customer requirements is possible. Our research will add to the body of knowledge in Software Engineering by reporting our approach along with an extensive evaluation of natural language similarities computation approaches for reuse analysis. Our research will also report an experiment on the use of a decision support system for reuse analysis.

We aim to include model-level similarity analysis to our variability-aware reuse approach. Given two models (one for tested product and other for a newly derived product) and one set of test cases, our future work will aim to perform reuse analysis for test cases. We aim to classify the test cases as reusable, tailor-able and not reusable classes. We then aim to automatically repair the tailor-able test cases for the newly derived product.

[3]https://www.ibm.com/se-en/marketplace/requirements-management

ACKNOWLEDGMENTS

This work has been supported by and received funding from the ITEA3 European XIVT, and ARRAY projects. The author would like to thank his advisors and people at Bombardier Transportation AB for their continued support.

REFERENCES

[1] Muhammad Abbas, Irum Inayat, Naila Jan, Mehrdad Saadatmand, Eduard Paul Enoiu, and Daniel Sundmark. 2019. MBRP: Model-Based Requirements Prioritization Using PageRank Algorithm. In *2019 26th Asia-Pacific Software Engineering Conference (APSEC)*. IEEE, 31–38.

[2] Muhammad Abbas, Irum Inayat, Mehrdad Saadatmand, and Naila Jan. 2019. Requirements dependencies-based test case prioritization for extra-functional properties. In *Proceedings - 2019 IEEE 12th International Conference on Software Testing, Verification and Validation Workshops, ICSTW 2019*. IEEE, 159–163.

[3] Sallam Abualhaija, Chetan Arora, Mehrdad Sabetzadeh, Lionel C Briand, and Eduardo Vaz. 2019. A Machine Learning-Based Approach for Demarcating Requirements in Textual Specifications. In *27th IEEE International Requirements Engineering Conference (RE'19)*. IEEE, 51–62. https://doi.org/10.1109/RE.2019.00017

[4] Maximiliano Arias, Agustina Buccella, and Alejandra Cechich. 2018. A Framework for Managing Requirements of Software Product Lines. *Electronic Notes in Theoretical Computer Science* 339 (2018), 5–20. https://doi.org/10.1016/j.entcs.2018.06.002

[5] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2002. Latent Dirichlet Allocation. In *Advances in Neural Information Processing Systems 14*, T. G. Dietterich, S. Becker, and Z. Ghahramani (Eds.). MIT Press, 601–608. http://papers.nips.cc/paper/2070-latent-dirichlet-allocation.pdf

[6] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching Word Vectors with Subword Information. arXiv:cs.CL/1607.04606

[7] Markus Borg, Per Runeson, and Anders Ardö. 2014. Recovering from a decade: a systematic mapping of information retrieval approaches to software traceability. *Empirical Software Engineering* 19, 6 (2014), 1565–1616. https://doi.org/10.1007/s10664-013-9255-y

[8] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. arXiv:cs.CL/1810.04805

[9] Nili Itzik, Iris Reinhartz-Berger, and Yair Wand. 2016. Variability Analysis of Requirements: Considering Behavioral Differences and Reflecting Stakeholders' Perspectives. *IEEE Transactions on Software Engineering* 42, 7, 687–706. https://doi.org/10.1109/TSE.2015.2512599

[10] Andreas Jedlitschka, Marcus Ciolkowski, and Dietmar Pfahl. 2008. *Reporting Experiments in Software Engineering*. Springer London, London, 201–228. https://doi.org/10.1007/978-1-84800-044-5_8

[11] Quoc V. Le and Tomas Mikolov. 2014. Distributed Representations of Sentences and Documents. arXiv:1405.4053 http://arxiv.org/abs/1405.4053

[12] Yan Li, Tao Yue, Shaukat Ali, and Li Zhang. 2019. Enabling automated requirements reuse and configuration. *Software and Systems Modeling* 18, 3 (2019), 2177–2211. https://doi.org/10.1007/s10270-017-0641-6

[13] Nan Niu, Juha Savolainen, Zhendong Niu, Mingzhou Jin, and Jing-ru C Cheng. 2014. A Systems Approach to Product Line Requirements Reuse. *IEEE Systems Journal* 8 (2014), 827–836. https://doi.org/10.1109/JSYST.2013.2260092

[14] Klaus Pohl, Günter Böckle, and Frank J van Der Linden. 2005. *Software product line engineering: foundations, principles and techniques*. Springer Science & Business Media.

[15] Radim Řehůřek and Petr Sojka. 2010. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. ELRA, Valletta, Malta, 45–50. http://is.muni.cz/publication/884893/en.

[16] Stephen Robertson. 2004. Understanding inverse document frequency: On theoretical arguments for IDF. *Journal of Documentation* 60, 5 (2004), 503–520. https://doi.org/10.1108/00220410410560582

[17] Per Runeson and Martin Höst. 2009. Guidelines for conducting and reporting case study research in software engineering. *Empirical Software Engineering* 14, 2 (2009), 131–164. https://doi.org/10.1007/s10664-008-9102-8

[18] Nathan Weston, Ruzanna Chitchyan, and Awais Rashid. 2009. A framework for constructing semantically composable feature models from natural language requirements. In *Proceedings of the 13th International Software Product Line Conference*. 211–220. https://doi.org/10.1145/1753235.1753265