Journal of Computer Engineering & Information Technology

Review Article

A SCITECHNOL JOURNAL

Clustering-aided Framework Assessment of Requirement Engineering Approaches Applied to Software Product Lines

Naoufel Kraiem*

Department of Intelligent Systems and Decision Engineering, University of Tunis El Manar, Tunis, Tunisia

***Corresponding author:** Naoufel Kraiem, Department of Intelligent Systems and Decision Engineering, University of Tunis El Manar, Tunis, Tunisia; E-mail: naoufel.kraiem@gmail.com

Received date: 16 May, 2022, Manuscript No. JCEIT-22-63929;

Editor assigned date: 19 May, 2022, PreQC No. JCEIT-22-63929 (PQ);

Reviewed date: 02 June, 2022, QC No. JCEIT-22-63929;

Revised date: 15 July, 2022, Manuscript No. JCEIT-22-63929 (R);

Published date: 22 July, 2022, DOI:10.4172/2324-9307.1000253.

Abstract

Requirements Engineering (RE) is recognized as a critical stage in software development lifecycle. The cost of fixing a requirements flaw later in the development stage is much higher than the cost of identifying and fixing it in the early stages of development. In order to do this the system requirements must be properly identified, analyzed and reviewed early in the development process. Given the nature of Software Product Lines (SPLs), the importance of requirements engineering is more renounced as SPLs pose more complex challenges than development of a 'single' product. Several approaches have been proposed in the literature, which encompass activities for capturing requirements, their variability and commonality.

This thesis mainly aims to propose a framework that will guide system engineers to choose an adequate approach for their preferred goal. The proposed framework is expected to decrease the time required to search an effective approach from several approaches presented together. The framework assesses RE approaches for SPL based on a selected criteria set. It makes further contributions by implementing a machine learning algorithm (k-means) to cluster the quantitative data built from the assessment. Furthermore, it implements a website that helps achieve the initial objective of this thesis.

The result of the framework was validated and it showed that the classified data is practical. This framework will decrease the probability of being misled while choosing a suitable RE approach applied to SPL.

Keywords: Software product line; Requirement engineering; Ma chine learning; Data clustering

Introduction

Software Product Line Engineering (SPLE) refers to a paradigm that promotes modeling and development in which the objective is no longer obtaining a software system, but a combination of software systems with similar characteristics. Software Product Line Engineering (SPLE) has proven to expedite organizations to develop a variety of similar systems at lower cost, in shorter time, and with higher quality when compared with the development of single systems exclusively. Software Product Line Engineering (SPLE) extracts and clearly represents the concept of variability to comfort the development of complex systems. In SPLE, the architecture of a system not only describes one product, but many of the same nature with various variable parts and modules. A Software Product Line (SPL) is a set of software intensive systems that share a common, managed set of features satisfying the specific needs of a particular market segment or mission and that are developed from a common set of core assets in a prescribed way. The SPL approach distinguishes two processes namely, domain engineering, and application engineering.

The first process is domain engineering process which deals with developing and maintaining reusable core or domain assets that typically are reusable components of software. In this process, products are built from a set of artifacts that have been specifically designed as a reusable core asset base. These core assets comprise the software architecture, its documentation, specifications, tools and software component. The second process is application engineering which deals with the development of software products, or applications, using these core assets for quick and efficient composition of software products tailored to the customer's requirements. The transition between the two domains is made through a configuration that adapts a domain model to define an application product.

Product Line Engineering (PLE) is a worthwhile and significant reuse based development model that allows firms to understand the improvements in time to market, cost, productivity, quality, and flexibility.

PLE is an important method and is used to reduce time to market and TCO cost as well as it reducing the configuration time of new products. According to Voelter's research, there are some reasons for adopting SPL in enterprises like, it minimizes development time, effort, cost, and complexity by taking advantage of the commonality within a portfolio of same products. Beside obvious advantages of SPL, there are also several potential drawbacks and challenges. According to Schaefer's findings, reducing the time of software development affects the product's quality. This is especially occurring in the development of applications with high safety and security requirements. Moreover, maintenance of product lines is more difficult and expensive compared with single system maintenance.

Deelstra, et al. observed that the derivation of individual products from shared software assets is still a time consuming and expensive activity in many organizations. The authors stated that "there is a lack of methodological support for application engineering and, consequently, organizations fail to exploit the full benefits of software product families." "Guidance and support are needed to increase efficiency and to deal with the complexity of product derivation".

The hardest single part of building a software system is deciding precisely what to build. No other part of the conceptual work is as difficult as establishing the detailed technical requirements. No other part of the work so cripples the resulting system if done wrong. No other part is as difficult to rectify later.

Requirements are statements of what the system must do, how it must behave, the properties it must exhibit, the qualities it must possess, and the constraints that the system and its development must satisfy. Among the other areas, Requirements Engineering (RE) is the most important area of Software engineering and possibly of the entire software life cycle, because errors produced at this stage, if undetected until a later stage of software development, can be expensive to resolve. The reliability, efficiency, usability and maintainability of a software system mainly depends on this phase.

Requirements engineering approaches applied to SPL has been an intensive research topic in the last years. Usually, RE approaches lists do not provide detailed and precise information about the new challenges that arises from the SPL adoption. In the last years, several approaches have been proposed to cover this limitation. But not much research has been done to compare how much coverage this approach has in Software Product Line Development (SPLD).

It is widely accepted that manual assessing of single approach is tedious and error prone. This is even worst when several approaches are represented altogether. Several methods have been proposed to assess requirement engineering approaches. But no tool has been proposed to automate the assessment of RE approaches. However, despite the relative success of these approaches, there are still a number of pending issues that have motivated the research presented in this thesis [1].

As there are several RE approaches available for SPL development, it is usually tough to evaluate and choose the required approach, thus resulting in choosing the wrong approach, causing problems, for example: delay in production of software, producing bad quality product, etc. The engineer must pragmatically go through each approach individually and find out which best fits the requirements, this wastes a lot of time, causing extending or running out of software deliver time. The current state of the art of RE for SPL shows some important features could be missed while searching for an adequate approach.

We aim to solve these issues using quantitative analysis of the approaches. The contributions of this research are the following:

- Significant differences in coverage of features among groups of RE approaches applied to SPL brought to light.
- A comparison framework to choose among different approaches that is adequate for a particular goal.
- An algorithm for the assessment of RE approaches using machine learning algorithm.
- A website for guiding users in selecting the appropriate approach.

We selected a large collection of RE approaches used in the field of SPLs that served as basis for our comparison. The criteria for selecting these approaches were, enough information available in the literature and some comparison done in other research papers on these approaches. In order to find requirements engineering approaches in software product lines that satisfy above mentioned criteria, we investigated existing research papers on the software product line requirements engineering.

Using these sources, we selected Feature Oriented Domain Analysis (FODA), Feature Oriented Reuse Method (FORM), Cardinality based Feature Modeling (CBFM), Goal and Scenario Based Domain Requirements Engineering, FeatuRSEB, Product Line Use case for Software and System engineering (PLUSS), AMPLE, Goal Driven Product Line engineering, AoURN-based Software Product Line, DREAM, Orthogonal Variability Model (OVM), I-GANDALF, SIRENspl and SREPPLine, based on the knowledge of authors on existing approaches on software product line requirements engineering. Since our work is still work-in-progress, this is not meant to be a comprehensive list.

The rest of the paper is structured as follows: Section 3 explains the evaluation criteria and the data collection, for evaluation of the approaches and section 4 deliberates the analysis and design of the proposed framework in detail and talks about its different phases. Section 5 discusses the evaluation for the recommended framework. It shows the different techniques used for evaluating the proposed framework. Finally, section 6 concludes and presents future work.

Literature Review

This review follows the systematic method proposed by Kitchenham and Webster and Watson. Several studies that proposed assessment of requirement engineering approaches for SPL were reviewed to answer the research questions. In the paper Blanes, et al. five criterions were defined to perform the comparative study. The first criterion analyzes the support to the SPL development by the RE approaches. The second criterion is the Requirements Engineering tasks that were covered by the RE approaches, which artifacts were employed, the type of requirements (functional or non-functional), and what type of traceability was supported. The third criterion is the model-driven coverage, the desired purpose of this adoption and, if followed up, which model and input models, language and transformation type is used. The fourth criterion is the automatic support to the approach with tool. Finally, the last criterion analyzed is the type of validation provided by the approach.

The paper Allen, et al. in order to analyze the selected approaches, chose twelve features to perform the comparative study. The features are evolvability, verification, trade-off-analysis, scalability, traceability, conceptual modeling, domain modeling, domain scoping, requirements modeling, commonality and variability modeling, aspect scoping and validation.

In Mohsen, et al. the evaluation criteria were developed in top-down (investigating existing evaluation frameworks) and bottom-up (investigating feature-oriented approaches) manners. In order to ensure the quality of the criteria set, they applied a set of meta-criteria, criteria for evaluating the criteria set. They defined three meta-criteria: coverage of requirements engineering, coverage of variability and commonality analysis, and coverage of tooling support. The first meta-criterion investigates if the criteria-set contains the required aspects for evaluating techniques with respect to requirements engineering principles and processes. Variability and commonality analysis the key principles in the success of software product line engineering forms the second meta-criterion. Adoption of technique by software developers in addition to detailed guidelines (processes) and explicit representations requires tooling support. The third meta-criterion investigates whether the criteria-set contains required criteria to evaluate this aspect.

Juan, et al. compared current RE tools in the quest to answer the following research question: What level of variation, in terms of functionality, is observable in state-of-practice RE tools? The technical report of Blanes, et al. presents an analysis of specific approaches used in the development of SPL to provide solutions to model variability and to deal with the requirements engineering activities.

Evaluation Criteria and Data Collection

In this section, requirements engineering approaches applied to product line engineering were evaluated according to a set of proposed criteria. These criteria set was taken from the research work of Mohsen, et al., had a well-defined feature set for comparison of the approaches, that covered most of the RE and SPL fields, and therefore it was selected for this paper.

Criteria description

In order to ensure about the quality of the criteria set, meta-criteria notion was applied, criteria for evaluating a criteria set. Three meta-criteria were defined as: Coverage of requirements engineering, coverage of variability and commonality analysis, and coverage of tooling support.

The first meta-criterion investigates if the criteria-set contains the required aspects for evaluating techniques with respect to requirements engineering principles and processes. Variability and commonality analysis the key principles in the success of software product line engineering forms the second meta-criterion. Adoption of technique by software developers in addition to detailed guidelines (processes) and explicit representations requires tooling support. The third meta-criterion investigates whether the criteria-set contains required criteria to evaluate this aspect.

For the first meta-criterion, three subcategories were further defined based on the knowledge on software development methods formulated in Software Process Engineering Meta-Model (SPEM) and principles related to the context of meta criterion (*i.e.* requirements engineering). According to SPEM, two important aspects of software development methods are process and artifact. Therefore, we considered both process and artifact aspects for evaluating the proposed approaches. With respect to process aspect, the generic requirements engineering process needs to encompass requirements elicitation, requirements modeling, requirements validation and verification, and requirements management. Hence, these steps form criteria for evaluating process aspects of requirements engineering in the SPL. Several artifacts have been used in the requirements engineering literature to represent requirements.

Among the existing artifacts, more commonly used artifacts were selected *i.e.* goal model, use-case model, scenario based model, and non-functional model as criteria. Also, according to requirements engineering papers, the needs of stakeholders include functional and non-functional requirements as well as preferences over the functional and non-functional requirements.

For the second meta-criterion, similarly both process and artifacts aspects were defined. Due to the nature of software product lines, a variability and commonality management process is distributed in both the domain engineering and the application engineering lifecycles. In the domain engineering lifecycle, the purpose mainly is to capture and model the variability and commonality while in the application engineering the purpose is to reuse the variability and commonality artifacts. According to the variability management literature, the variability process includes identifying, analyzing, modeling, and binding (configuring) the variability. Hence, requirements engineering approaches should cover these steps to manage variability in requirements models. Managing delta requirements (*i.e.* requirements that are not covered by the product line) is an important step in the application engineering lifecycle. With respect to artifacts, variability can be represented in a variability dimension (*i.e.* feature models) or in software development artifacts (e.g. use-cases) or combination of the variability dimension and the development artifacts. Regarding principles in variability management, types of variability and a strategy adopted for developing a product line were considered [2].

For the third meta-criterion, modeling support, support for traceability, and automatic validation criteria were selected based on criteria defined in for evaluating tooling support.

Table 1 shows the criteria set for evaluating requirements engineering in the software product line domain. I do not claim that the criteria set is complete, but it provides a proper set of criteria to highlight some challenges in the existing requirements modeling techniques in product lines.

Meta -criteria	Criteria		Description
Coverage of requirements engineering	Requirement Types	Functional Requirements	Ability of the technique to manage functional requirements of stakeholders
		Non-functional requirements	Ability of the method to manage non-functional requirements of stakeholders
		Preferences	Ability of the method to manage preferences of stakeholder in terms of prioritization of both functional and non-functional requirements.
	Process aspect	Requirements elicitation	Method explicitly defined an activity or mentioned reusing traditional requirements elicitation techniques for requirements elicitation in SPL
		Requirements modeling and analysis	Method explicitly defined an activity or mentioned reusing traditional requirements modeling and analysis

			techniques for requirements modeling in SPL
		Requirements validation and verification	Method explicitly defined an activity or mentioned reusing traditional requirements validation and verification techniques for requirements validation in SPL
		Requirements management	Method explicitly defined an activity or mentioned reusing traditional requirements management techniques for requirements management in SPL
	Artifact aspect	Goal-model	A representation for objectives of stakeholders
		Use-case model	Representation for functional requirements in the family
		Scenario based models	Representation for behavior part of the systems in family
		Non-functional representation	Representation for non-functional requirements in family
	Variability types	Optional Variability	Special Case of single variant when only available variant set consist of one variant
Coverage of variability and commonality		Alternative variability	From a set of variants in the binding time a single variant is picked
		Multi-parallel variability	From set of variants in the binding time one or more variants can be picked
	Process domain engineering	Identification	A method is explicitly defined for identifying variability and commonality in the approach
		Analyzing	A method explicitly is defined for identifying types of variation and dependency between them
		Modeling	A method is explicitly defined for representing the variability
	Process application engineering	Configuring	A method is explicitly defined about how to bind the variation points and when to bind them
		Reusing	A method is explicitly defined about how to instantiate reusable requirements
		Identify deltas	A method is explicitly defined about how to manage application requirements not covered in the family.
	Product	Variability dimension	If there is a separate variability dimension for visualizing variability in the family
		Artifact dimension	If the existing artifacts were extended to incorporate variability into them
	Adoption Strategy	Proactive strategy	Whether method considers developing the product line from scratch

		Extractive strategy	Whether method considers developing the product line from set of existing products
		Reactive strategy	Whether method considers evolving product line for new requirements
	Tooling Support	Automatic validation	Whether method describes a validation approach for checking inconsistency between variability in variability dimension and other requirements artifacts
Coverage of tooling Support		Support for traceability	If method provides vertical and horizontal traceability
		Modeling support	If the method explicitly develop a tooling support

Table 1: A set of criteria for evaluating requirements engineering me thods in software product line.

The results of analysis are shown in Table 2. The results are achieved by reading publications related to each method and exploring for coverage of criteria in the publications.

Approaches / Criteria		Foda	Pluss	Dream	OVM	Alférez et al.	Bragança and Machado	Coelho and Batista	Corriveau et al.	Dhungana et al.	Guelfi and Perrouin	In gandalf	Siren spl	Sreppline	From	Cbfm	Goal-Driven sple	Goal and Scenario	Aourn	FeaturSEB	Ample
Require-ment types	Functional requirement	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F
	Non-functional requirement	P	F	N	F	N	N	F	F	N	N	F	F	F	N	N	P	N	F	F	N
	Preference	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
Process aspect	Requirement elicitation	F	F	F	F	F	F	N	N	F	N	F	F	F	F	F	F	F	F	F	F
	Requirement modeling and analysis	F	F	P	P	P	P	P	P	P	P	P	P	P	F	F	F	F	F	F	F
	Requirement validation and verification	N	N	N	F	N	N	N	P	N	N	N	P	P	N	F	N	N	N	N	N
	Requirement management	N	F	F	F	F	N	N	N	N	N	N	N	N	N	N	N	N	N	F	F
Artifact aspect	Goal model	N	N	N	N	N	N	N	N	N	N	F	N	N	N	N	F	F	F	N	N
	Use-case model	P	F	P	N	F	F	N	N	N	P	N	F	P	N	F	N	F	N	F	F
	Scenario based model	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	F	F	F	N
	Non-functional model	N	F	N	N	N	N	N	N	N	N	N	F	N	N	N	F	N	F	N	N

Variability types	Optional variability	F	F	N	N	N	N	N	N	N	N	N	N	N	F	F	F	F	F	F	F
	Alternative variability	F	F	N	N	N	N	N	N	N	N	N	N	N	F	F	F	F	F	F	F
	Multi-parallel variability	F	F	N	N	N	N	N	N	N	N	N	N	N	F	F	F	F	F	F	F
Process domain engineering	Identification	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F
	Analyzing	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F
	Modeling	F	F	F	N	F	F	N	N	N	F	F	F	F	F	F	F	F	F	F	F
	Configuring	N	P	N	N	N	N	N	N	N	N	N	N	N	F	F	F	N	F	F	F
Process application engineering	Reusing	N	P	F	N	F	N	N	N	N	F	N	N	N	F	F	F	F	F	F	N
	Identify deltas	N	F	N	F	N	N	N	N	F	N	N	N	F	N	N	N	N	N	N	N
Product	Variability dimension	F	F	N	N	N	N	N	N	N	N	N	N	N	F	F	F	N	F	F	F
	Artifact dimension	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	F	F	N	F	F
Adoption strategy	Proactive strategy	F	F	F	F	F	F	F	N	N	N	F	F	F	F	F	F	F	F	F	F
	Extractive strategy	P	N	F	F	F	N	N	N	N	N	N	N	F	N	F	N	N	N	F	F
	Reactive Strategy	P	F	N	F	N	F	F	N	F	N	F	F	F	N	N	N	N	N	F	N
Tooling support	Automatic validation	P	N	N	N	N	N	N	N	N	N	N	N	N	N	F	P	N	F	N	N
	Support for traceability	F	F	P	F	P	F	P	N	N	P	P	P	F	F	F	F	N	F	F	F
	Modeling support	F	F	N	N	N	N	N	N	N	N	N	N	N	F	F	F	F	F	F	F

Table 2: The Evaluation Result of Re Approaches for Spl -- F (Fully Supported), N (Not Supported), P (Partially Supported).

Comparative study

In this section, a comparative analysis of the most important requirements engineering proposals that have been published to support the development of software products following the software product line approaches has been presented. The results were achieved by reading publications related to each approach and exploring for coverage of criteria in the publications.

The results revealed that validation and verification, preferences, artifact aspect and product artifact dimension have been neglected by researchers. Additionally, delta requirements should be handed during the application engineering process [3].

Proposed framework

This section demonstrates the suggested framework in detail. There are three phases in the framework that are clearly shown in Figure 1. The first phase is for gathering of data. The second phase is for clustering data using machine learning. And the last phase is for classification of data using mean value. The main purpose of the framework is to help users in selecting the most adequate approach that is applicable to their SPLD.

The approach contribution is summarized in three phases.

First phase (Gathering of data): In this phase, a number of approaches were selected based on the amount of information available on them in the existing papers. Also the evaluation criteria are selected in this phase and detailed. Qualitative data was then collected from expert research papers, which was later converted to quantitative data following the work of Juan, et al. (2014).

Second phase (Clustering using machine learning): The score table derived from the quantitative raw data was used as the dataset in machine learning algorithms. The algorithms helped to visualize the data into clusters.

Third phase (Classification using mean): The third phase is responsible for taking user input from a user friendly website, calculating the scores and adding it to the database. The site then sorts out the data from lowest score to highest score and shows as a comparison table (Figure 1).

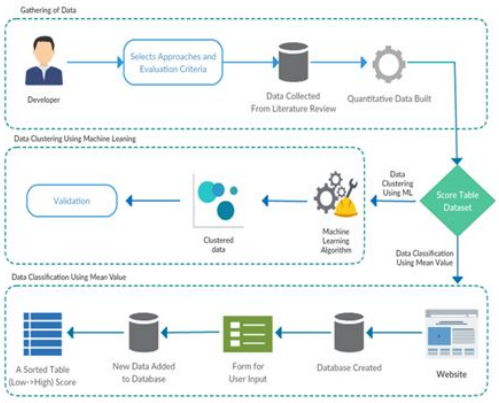

Figure 1: Framework overview.

Descriptive Analysis

As shown previously, the data that was collected is qualitative as there are three possible answers to each feature: "Fully Supported", "Partially Supported" and "Not Supported". But the proposed framework to assess the approaches is based on quantitative data analysis. Which leaded to consider an efficient way to change the qualitative data to quantitative data. The research paper by Juan et al., suggested a quantitative approach to compare RE tools, which was used for further work in the proposed framework of this paper. The data in Table 1 was converted to quantitative data, by assigning a point 1 for fully supported, point 0.5 for partially supported and point 0 for not supported.

The features of the evaluation criteria are grouped into criterions of features. For each criterion of features c, the score s of the approach a in the criteria c is calculated using the formula:

$$score\ (a,c) = \frac{\sum_{f=1}^{Nf(c)} score\ (a,f)}{Nf(c)} \qquad (1)$$

Equation 1: Nf (c) is the number of features of the criteria c. After this, s=score (a,c) Є [0,1] is discretized on a 3-interval scale, the intention being to discriminate extreme scores:

$$\text{Discretization(s)} = \begin{cases} \text{Lowest Score Group; } s \in (0{:}0; 0{:}3); \\ \text{Medium Score Group; } s \in [0{:}3; 0{:}5); \\ \text{Highest Score Group; } s \in [0{:}5; 1{:}0]; \end{cases} \qquad (2)$$

A total of nine criterions were studied (Table 1): requirement types (3 features), process aspect (4 features), artifact aspect (4 features), variability types (3 features), process domain engineering (3 features), and process application engineering (3 features), product (2 features), adoption strategy and tooling support (3 features), and other tool capabilities (3 features). The scores of the tools can be represented by means of 9-tuples, ordered lists of values that codify the scores achieved in each category of features e.g. the scores of the approach number 1 (A1) are (0.50, 0.50, 0.13, 1.00, 1.00, 0.00, 0.50, 0.67, 0.83). All the individual scores of each approach are shown in Table 3. Furthermore, the raw data to each single feature is available in the Table 1, thus guaranteeing the reproducibility of my research [4].

Summary statistics are presented in Table 3 to describe the criterions of the study. The arithmetic mean (a measure of central tendency) and median (a measure of central tendency) are included.

Approaches	Requirement Types	Process Aspect	Artifact Aspect	Variability Types	Process DE	Process AE	Product	Adoption Strategy	Tooling Support
A1	0.5	0.5	0.13	1	1	0	0.5	0.67	0.83
A2	0.67	0.75	0.5	1	1	0.67	0.5	0.67	0.67
A3	0.33	0.83	0.5	0	1	0.33	0	0.67	0
A4	0.67	0.88	0	0	0.67	0.33	0	1	0
A5	0.33	0.63	0.25	0	1	0.33	0	0.67	0
A6	0.33	0.5	0.33	0	1	0	0	0.67	0
A7	0.67	0.13	0	0	0.67	0	0	0.67	0
A8	0.67	0.25	0	0	0.67	0	0	0	0
A9	0.33	0.38	0	0	0.67	0.33	0	0.33	0
A10	0.33	0.13	0.13	0	1	0.33	0	0	0
A11	0.67	0.38	0.25	0	0.33	0	0	0.67	0
A12	0.67	0.5	0.5	0	1	0	0	0.67	0
A13	0.67	0.5	0.13	0	1	0.33	0	1	0

A14	0.33	0.5	0	1	1	0.67	0.5	0.33	0.67
A15	0.33	0.75	0.25	1	1	0.67	0.5	0.67	1
A16	0.5	0.5	0.5	1	1	0.67	1	0.33	0.83
A17	0.33	0.5	0.75	1	1	0.33	0.5	0.33	0.33
A18	0.67	0.5	0.75	1	1	0.67	0.5	0.33	1
A19	0.67	0.75	0.5	1	1	0.67	1	1	0.67
A20	0.33	0.75	0.25	1	1	0.33	1	0.67	0.67

Table 3: Approach score A1 to A20 representing the approaches in Table 2 in order.

Most approaches achieve broad coverage of features in process domain engineering, and adoption strategy (the median of their scores is very high), which therefore seem to be better covered than in the case of the other variables. In this sense, the categories requirements types and process aspect represent a second step, since the median of their scores is high (Table 4).

Criteria	Min	Max	Mean	Median
Requirements types	0.33	0.67	0.5	0.5
Process aspect	0.13	0.88	0.53	0.5
Artifact aspect	0	0.75	0.29	0.25
Variability types	0	1	0.42	0
Process DE	0.33	1	0.9	1
Process AE	0	0.67	0.33	0.33
Product	0	1	0.29	0
Adoption strategy	0	1	0.57	0.67
Tooling support	0	1	0.3	0

Table 4: Descriptive Statistics.

Moreover, the median of the distribution is low in variability types, product and tooling support features than in the other categories, meaning that these capabilities are not yet common in the RE approach market. The max and the min score shows that there are approaches covering all the categories of features, though it might not be one approach that covers all the features. The mean and median of process domain engineering is the highest concluding that it is the most covered feature in all approaches. We deduce from this that a lot of features need yet to be included in the approaches.

Evaluation and Discussion

This chapter reports the results of the ideas proposed in this thesis. In particular, this chapter's goal is to evaluate the results of the machine learning implementation and discuss the originality of our work. The implementations related with machine learning and website could not be compared with other solutions due to the fact that, to the best of my knowledge, there are not implementations in literature to be compared to.

Validation

From the machine learning algorithm implementation, we achieved three distinct groups of approaches, highest score group, medium score group and lowest score group. We calculated the min and max of every group in all the 9 criterions.

The following step was then carried out to find out which pairs of clusters were significantly different from each other, as a significant result of the overall test *i.e.* ANOVA, Kruskal-Wallis) just points out that at least one pair is different. Post-hoc paired comparisons can thus be applied to get the exact result [5].

As it can be seen from the tables below, the minimum of the lowest score group is lower than the minimum of the medium and highest scoring group of approaches. And the maximum of the lowest score group is lesser than the maximum of the medium and highest scoring group (Tables 5-7).

Approaches	Requirement Types	Process Aspect	Artifact Aspect	Variability Types	Process DE	Process AE	Product	Adoption Strategy	Tooling Support	Average
A1	0.5	0.5	0.13	1	1	0	0.5	0.67	0.83	0.59
A2	0.67	0.75	0.5	1	1	0.67	0.5	0.67	0.67	0.71
A14	0.33	0.5	0	1	1	0.67	0.5	0.33	0.67	0.5
A15	0.33	0.75	0.25	1	1	0.67	0.5	0.67	1	0.65
A16	0.5	0.5	0.5	1	1	0.67	1	0.33	0.83	0.7
A17	0.33	0.5	0.75	1	1	0.33	0.5	0.33	0.33	0.56
A18	0.67	0.5	0.75	1	1	0.67	0.5	0.33	1	0.71
A19	0.67	0.75	0.5	1	1	0.67	1	1	0.67	0.81
A20	0.33	0.75	0.25	1	1	0.33	1	0.67	0.67	0.67
Min	0.33	0.5	0	1	1	0	0.5	0.33	0.33	0.5
Max	0.67	0.75	0.75	1	1	0.67	1	1	1	0.81

Table 5: Highest scoring group of approaches.

Approaches	Requirement Types	Process Aspect	Artifact Aspect	Variability Types	Process DE	Process AE	Product	Adoption Strategy	Tooling Support	Average
A3	0.33	0.83	0.5	0	1	0.33	0	0.67	0	0.41
A4	0.67	0.88	0	0	0.67	0.33	0	1	0	0.39
A5	0.33	0.63	0.25	0	1	0.33	0	0.67	0	0.36
A6	0.33	0.5	0.33	0	1	0	0	0.67	0	0.31
A12	0.67	0.5	0.5	0	1	0	0	0.67	0	0.37
A13	0.67	0.5	0.13	0	1	0.33	0	1	0	0.4
Min	0.33	0.5	0	0	0.67	0	0	0.67	0	0.31
Max	0.67	0.88	0.5	0	1	0.33	0	1	0	0.41

Table 6: Medium scoring group of approaches.

Approaches	Requirement Types	Process Aspect	Artifact Aspect	Variability Types	Process DE	Process AE	Product	Adoption Strategy	Tooling Support	Average
A7	0.67	0.13	0	0	0.67	0	0	0.67	0	0.24
A8	0.67	0.25	0	0	0.67	0	0	0	0	0.18
A9	0.33	0.38	0	0	0.67	0.33	0	0.33	0	0.23
A10	0.33	0.13	0.13	0	1	0.33	0	0	0	0.21
A11	0.67	0.38	0.25	0	0.33	0	0	0.67	0	0.26
Min	0.33	0.13	0	0	0.33	0	0	0	0	0.18
Max	0.67	0.38	0.25	0	1	0.33	0	0.67	0	0.26

Table 7: Lowest scoring group of approaches.

For our results this is valid except for two contradictions, which is shown in red. This shows that lowest score group is well separated from the rest of the groups. Thus proving our implementation is valid. In future more algorithms could be implemented to diminish the above mentioned contradiction and improving the results.

To measure the quality of clustering results, there are two kinds of validity indices: external indices and internal indices. Silhouette index was used from internal indices was used to evaluate the results using quantities and features inherent in the data set. A higher silhouette coefficient score relates to a model with better defined clusters.

The Silhouette coefficient for my algorithm is +0.38. As the score is positive, this means that the data is clustered correctly. We don't have a+1 because my data is not very populated. But from the positive scoring it can be seen that the more the dataset will be populated in the future, the more the score will move towards +1 as the clusters will be denser. Since the score is around zero, it indicates there are some overlapping approaches, which is also shown in the validation table above as contradicting values of minimum and maximum [6].

Originality of this framework

We have used multiple research methods to search for related papers to our research. And have come across plethora of papers that compared RE approaches compared RE approaches for SPL. But there were no papers to the best of our knowledge that had any research done to automate the comparison of RE approach for SPL and motivate in guiding system engineers and domain experts to choose an appropriate approach for their goals.

Our contribution or add value to the above mentioned research papers is guiding system users in choosing an adequate approach easily. For helping in guiding users, we have used machine learning algorithms to cluster a set of RE approaches applied to SPL into three groups of approaches, namely, highest scoring approach group, medium scoring approach group and lowest scoring approach group. We have also built a website that has a previous dataset of approaches and their details and also includes a form page, in case a user wants to enter the details of another approach, and finally the website will display all the approaches with their mean score in a sorted table. And the user could use this as a guide to choose their adequate approach.

Our contribution could be improved in many ways in the future. The algorithm could be improved to achieve better clustering results, the dataset could be improved by adding in more number of approaches and the website could be improved in numerous ways to help guide users with better UI experience. These are few of the research goals we would like to achieve in the near future.

Conclusion

The main objective of this thesis is to answer the research question: How can we assess RE approaches for SPL? To answer this question, this thesis proposes:

- A state of the art in requirement engineering approaches for software product line.
- An evaluation criterion to evaluate the approaches based on the criterion selected from an existing research work. The evaluation was done by studying expert research papers in the literature.
- A descriptive analysis of the data and the calculations used to create the final data.
- A framework that would help assess these approaches.
- A tool to automate the comparison of these approaches that was implemented in machine learning using the programming language python.
- A demonstration of the framework, a website created using the programming language JavaScript.
- Validation of results of the framework using Kruskal-Wallis Test and Silhouette Coefficient.

This research consists of a framework that clusters the RE approaches for SPL, into three groups, namely, highest score group, medium score group and lowest score group of approaches. The score of an approach represents to what extent does it cover the features of RE and SPL. Therefore, the highest score group displays the approaches that might be most preferable by the system engineers.

To achieve this proposed framework uses ML to improve comparison issues. ML is a field of computer science that uses statistical techniques to give computer systems the ability to "learn" (*i.e.*, progressively improve performance on a specific task) with data, without being explicitly programmed.

The proposed framework also uses statistics, such as mean value, to compare the approaches. The main goal of this thesis is to guide users in selecting an adequate approach to achieve their particular need. In order to achieve our goal, we have developed a website that allows users to add in, details of a new approach, if they want to compare it with other approaches that are available in the website. Finally, the results proposed framework had been validated.

This project focuses on proposing a framework for assessing RE approaches using SPL, that programs two algorithms combined (k-means and PCA) in machine learning. The K-means method as a strategy that attempts to find optimal partitions. Since this development, K-means has become extremely popular, earning a place in several textbooks on multivariate methods. In the future, we will propose several other algorithms to assess the approaches and compare their runtimes. Also, the website built in this framework can be made more efficient to guide the requirement engineers to choose their adequate approach.

In future the clustering could be made using homogenous scores in features of the approaches therefore making it easier for users to select an approach with the features of their requirement. Another approach could be to use binary digits 1 and 0 for each feature and cluster the approaches according to the score of each feature. We could also use weighted features to cluster the approaches. These all will lead to help guide the system engineers in the future in a more improved way. A comparative study by using other clustering algorithms will be considered to validate our framework.

References

1. Mellado D, Fernández-Medina E, Piattini M (2010) Security requirements engineering framework for software product lines. Inf Softw Technol 52:1094-1117.
2. Mellado D, Mouratidis H, Fernández-Medina E (2014) Secure Tropos framework for software product lines requirements engineering. Comput Stand Interfaces 36:711-722.
3. Alves V, Niu N, Alves C, Valença G (2010) Requirements engineering for software product lines: A systematic literature review. Inf Softw Technol 52:806-820.
4. Arias M, Buccella A, Cechich A (2018) A framework for managing requirements of software product lines. Electron Notes Theor Comput Sci 339:5-20.
5. Chen L, Babar MA (2011) A systematic review of evaluation of variability management approaches in software product lines. Inf Softw Technol 53:344-362.
6. Neto PA, do Carmo Machado I, McGregor JD, De Almeida ES, de Lemos Meira SR (2010) A systematic mapping study of software product lines testing. Inf Softw Technol 53:407-423.