Introducing Collaboration for Locating Features in Models: Approach and Industrial Evaluation

Francisca Pérez, Ana C. Marcén, Raúl Lapeña, and Carlos Cetina

Universidad San Jorge. SVIT Research Group
Autovía A-23 Zaragoza-Huesca Km.299, 50830, Zaragoza, Spain
{mfperez, acmarcen, rlapena, ccetina}@usj.es

Abstract. Feature Location (FL) is one of the most important tasks in software maintenance and evolution. However, current works on FL neglected the collaboration of different domain experts. This collaboration is especially important in long-living industrial domains where a single domain expert may lack the required knowledge to fully locate a feature, so the collaboration among different domain experts could alleviate this lack of knowledge. In this work, we address collaboration among different domain experts by automatically reformulating their feature descriptions. With our approach, we extend existing FL approaches based on Information Retrieval and Linguistic rules to locate features in models. We evaluate our approach in a real-world case study from our industrial partner, which is a worldwide leader in train manufacturing. We analyze the impact of our approach in terms of recall, precision, and F-Measure. Moreover, we perform a statistical analysis to show that the impact of the results is significant. Our results show that our approach for collaboration boosts the quality of the results of FL.

Keywords: Collaborative Information Retrieval, Feature Location, Query Expansion, Model Driven Engineering

1 Introduction

Nowadays, work environments are characterized by an emphasis on collaborative team work [9]. Many empirical studies identified collaborative information seeking and retrieval as everyday work patterns in order to solve a shared information need and to benefit from the diverse expertise and experience of the team members [13].

Despite the importance of collaboration, Feature Location (FL) approaches neglected collaboration among different domain experts to find the set of software artifacts (e.g., code or models) that realize a specific feature. Even though collaboration is a useful and often necessary component of complex projects in industrial contexts when the task at hand is difficult or cannot be carried out by one individual [36].

To cope with this lack, the contribution of this paper is the introduction of collaboration for locating a target feature in models from different domain experts. First, each domain expert provides both a feature description and an estimation of confidence level. Then, our approach uses the confidence level to identify relevant feature descriptions. After, our approach automatically reformulates the relevant feature descriptions in a single query using a technique that is based on Rocchio's method [34]. The resulting query is used to to find the model fragment that realizes the feature being located using two different FL cores (Information Retrieval (IR) or Linguistic rules) since they obtain the best results in the literature [30,21,29,38].

We analyze the impact of collaboration in a real-world industrial case study from the railway domain. Our industrial partner, Construcciones y Auxiliar de Ferrocarriles (CAF)[1], is a worldwide leader in train manufacturing. CAF provided us with both the models of software that control and manage the trains and the oracle (the realization of features validated by our industrial partner). Then, we involve 19 domain experts from our industrial partner to obtain feature descriptions and confidence levels as the input of our approach. We compare the model fragment that realizes each of the target features that our approach obtains as a result with the oracle (which is considered to be the ground truth) in terms of recall, precision, and F-measure. Finally, we perform a statistical analysis in order to provide quantitative evidence of the impact of the results.

The results of this paper show that introducing collaboration boost the quality of the results in the existing FL approaches: IR obtains an improvement of 37.19% in F-measure, and Linguistic rules obtain an improvement of 29.31% in F-measure. We hope that these results promote the introduction of collaborative mechanisms in FL.

The rest of the paper is structured as follows: Section 2 provides the required background. Section 3 presents our approach to introduce collaboration, and the cores of IR and Linguistic rules. Section 4 describes the evaluation carried out. Section 5 describes the threats to validity. Section 6 reviews the related work. Finally, Section 7 concludes the paper.

2 Background

The Domain Specific Language (DSL) of our industrial partner has the expressiveness required to describe both the interaction between the main pieces of equipment installed in a train unit and the non-functional aspects that are related to regulation. We present an equipment-focused simplified subset of the DSL for the sake of understandability and legibility and due to intellectual property rights concerns. This subset of the DSL will be used to present a running example throughout the rest of the paper.

Fig. 1 shows an example of a product model from a real-world train. It shows two separate pantographs (High Voltage Equipment) that collect energy from the overhead wires and send it to their respective circuit breakers (Contactors), which, in turn, send it to their independent Voltage Converters. The converters

[1] `www.caf.net/en`

then power their assigned Consumer Equipment: the HVAC on the left (the train's air conditioning system), and the PA (public address system) and CCTV (television system) on the right.

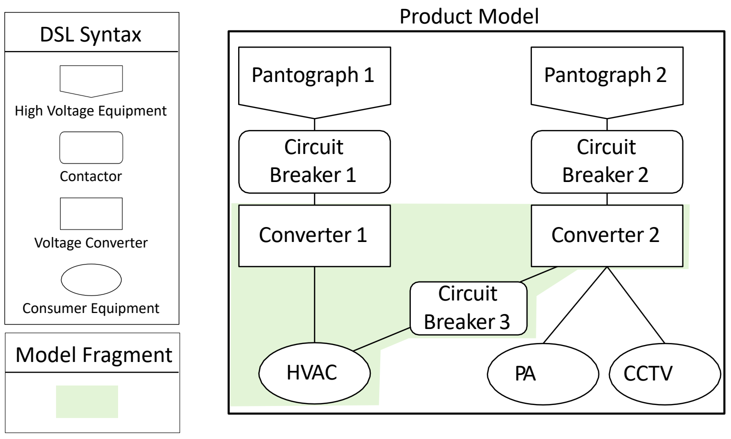


Fig. 1. Example of product model and model fragment

An example of model fragment is also shown in Fig. 1. The elements of the model fragment are highlighted in green, which are the realization of the feature: HVAC Assistance. This feature allows the passing of current from one converter to the HVAC that is assigned to its peer for coverage in case of overload or failure of the first converter.

3 Our approach for introducing Collaboration in FL

Fig. 2 presents our approach to introduce collaboration from different domain experts for locating features in models. First, each domain expert involved provides both a feature description and a self-rated confidence level for the feature name as input. Second, one of the feature descriptions is automatically reformulated to include relevant terms from other feature descriptions. To do this, feature descriptions are ordered from the highest to the lowest confidence level. The first feature description is set as base query, and the k subsequent feature descriptions are set as relevant documents. Next, the base query is automatically reformulated to expand it with the most representative terms found in the relevant documents. Finally, both the product model and the reformulated query are taken as input to locate the relevant model fragments. To locate the relevant model fragments, we use two different FL cores (IR or Linguistic rules) since they obtain the best results in the literature [30,21,29,38]. The result after the core is executed is a model fragment to the input reformulated query.

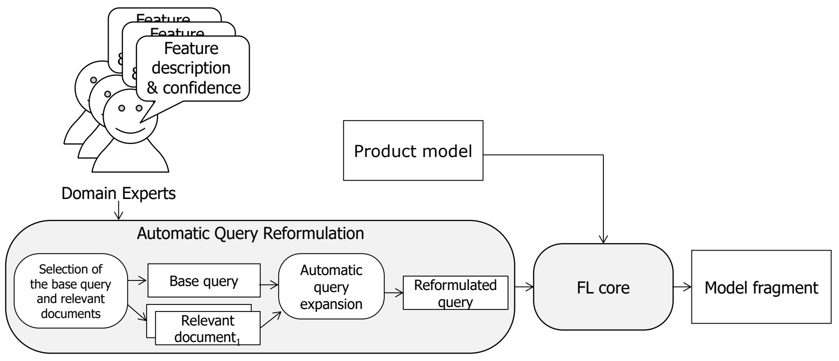

Feature description & confidence
Product model
Domain Experts
Automatic Query Reformulation
Selection of the base query and relevant documents
Base query
Relevant document1
Automatic query expansion
Reformulated query
FL core
Model fragment

Fig. 2. Our approach for introducing collaboration in feature location

In the next three subsections, we describe how the automatic query reformulation is performed, and how FL can be performed using one of the FL cores: IR or the Linguistic rules.

3.1 Automatic Query Reformulation

In order to introduce collaboration from different domain experts' feature descriptions, our approach starts with the selection of one feature description as the base query (Step 1). Afterwards, the base query is automatically reformulated to expand it with the most representative terms found in other domain experts' feature descriptions set as relevant documents (Step 2). These two steps are performed as follows:

Step 1) Selection of the base query and relevant documents.

Our approach sorts the feature descriptions provided by the domain experts from the highest to the lowest self-rated confidence level in order to select the feature description in the first position (i.e., the highest self-rated confidence) as the base query. The self-rated confidence level is supplied for each feature description using a Likert scale ranging from 7 (the highest self-rated confidence) to 1 (the lowest self-rated confidence). Then, our approach selects k feature descriptions sorted by confidence level, where k is the number of domain experts who collaborate to reformulate the base query. Each of the selected feature descriptions is set as a relevant document.

For example, the feature description *"Passing of current from one converter to the HVAC assigned to its peer for coverage in case of overload or failure of the first converter"* provided by Domain expert A is selected as the base query since it has the highest self-rated confidence level (6). Next, two subsequent feature descriptions with the highest self-rated confidence level are set as relevant documents since two exerts are established to collaborate in the reformulation of the query (k=2). These feature descriptions are: *"The circuit breaker changes to another converter in case of failure in the HVAC converter"* (from Domain expert B since the self-rated confidence level is 4); and *"In case of failure or overload in the converter that provides energy to the air conditioning unit, the circuit breaker provides energy from its converter"* (from Domain expert C since the self-rated confidence level is 3).

Step 2) Automatic Query Expansion.

Once the base query and the relevant documents are set, our approach homogenizes the Natural Language (NL) text before the base query is expanded. Text homogenization is a frequent practice [17] by combining Natural Language Processing (NLP) techniques, such as the analysis of POS tags, removal of stopwords, and stemming. Our approach adopts the NLP techniques as follows:

- The text is tokenized (divided into words). A white space tokenizer can usually be applied (which splits the strings whenever it finds a white space); however, for some sources of description, more complex tokenizers need to be applied such as CamelCase naming.
- The Parts-of-Speech (POS) tagging technique is applied to analyze the words grammatically and to infer the role of each word in the text provided. As a result, each word is tagged, which allows the removal of some categories that do not provide relevant information. For instance, conjunctions (e.g., *or*), articles (e.g., *a*), or prepositions (e.g., *at*) are words that are commonly used and do not contribute relevant information to describe the feature, so they are removed.
- Stemming techniques are applied to unify the language that is used in the text. This technique consists of reducing each word to its root, which allows different words that refer to similar concepts to be grouped together. For instance, plurals are turned into singulars (*doors* to *door*) or verb tenses are unified (*using* and *used* are turned into *use*).
- The Domain Term Extraction and Stopword Removal techniques are applied. In order to carry out these techniques, domain experts provide two separate lists of terms: one list of both single-word and multiple-word terms that belong to the domain and must be kept for analysis, and a list of irrelevant words that have no analysis value. Both kinds of terms can be automatically filtered in or out of the final query.

For example, the terms of the feature description that is set as the base query (*Passing of current from one converter to the HVAC assigned to its peer for coverage in case of overload or failure of the first converter*) are homogenized as follows: *current, convert, hvac, coverag, overload, failur, convert,* and *assign.*

Once the NL text is homogenized, our approach automatically reformulates the base query to expand it with terms of the relevant documents using a technique that is based on Rocchio's method [34], which is perhaps the most commonly used method for query reformulation [37]. Rocchio's method orders the terms in the top K relevant documents based on the sum of the importance of each term of the K documents using the following equation:

$$Rocchio = \sum_{d \in R} TfIdf(t,d) \tag{1}$$

where R is the set of top K relevant documents in the list of retrieved results, d is a document in R, and t is a term in d. The first component of the measure is the Term Frequency (Tf), which is the number of times the term appears in a document; it is an indicator of the importance of the term in the document compared to the rest of the terms in that document. The second component is the Inverse Document Frequency (Idf), which is the inverse of the number of documents that contain that term; it indicates the specificity of that term for a document that contains it. Once the terms of the relevant documents are ordered, we consider the first 10 term suggestions to expand the base query, as is recommended in the domain literature [5].

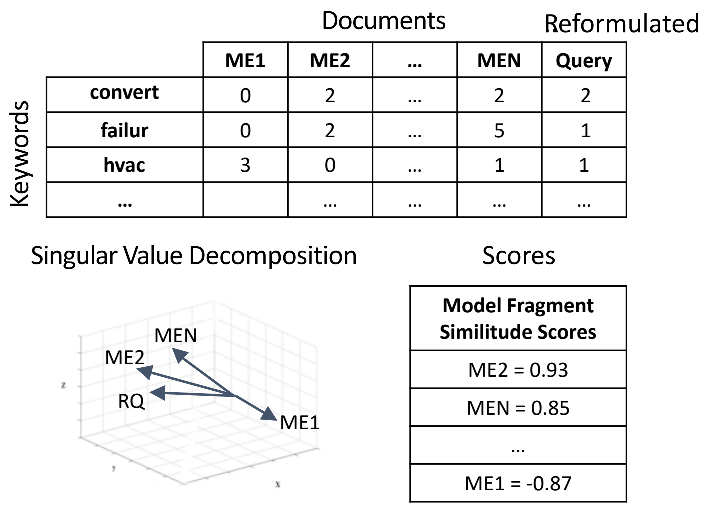
For example, the first 10 terms from the relevant documents set in the previous step (*convert, energi, provid, overload, circuit, breaker, failur, hvac, air, condit*) are used to reformulate the base query by adding these terms. Therefore, the reformulated query is made up of the following terms: *current, convert, hvac, converag, overload, failur, assign, energi, provid, circuit, breaker, air,* and *condit.*

3.2 IR FL CORE

IR [12,23,35] is a sub-field of computer science that deals with the automated storage and retrieval of documents. There are many IR techniques, but most of the efforts show better results when applying LSI [30,21,29]. Hence, we use LSI to recover the model fragment that realizes a feature description as one of the FL cores of our approach.

LSI [19] is an automatic mathematical/statistical technique that analyzes relationships between *queries* and *documents* (bodies of text). It constructs vector representations of both a user *query* and a corpus of text *documents* by encoding them as a *term-by-document co-occurrence matrix*, and analyzes the relationships between those vectors to get a similarity ranking between the reformulated *query* and the *documents*. Each row in the co-occurrence matrix (*term*) stands for each of the words that compose the reformulated query and NL representation of the input model, extracted through the technique presented in [28]. In Fig. 3, it is possible to appreciate an example of matrix in which the rows are a set of representative terms in the domain such as 'pantograph' or 'door'. Each column in the matrix (*document*) stands for one model element from one input model, taken from our real world case study. In Fig. 3, it is possible to appreciate identifiers in the columns such as 'ME1' or 'ME2', which stand for the *documents* of those particular model elements. The final column stands for the *reformulated query*. Each cell in the matrix contains the frequency with which the *term* of its row appears in the *document* denoted by its column. For instance, in Fig. 3, the *term* 'pantograph' appears twice in the 'ME2' *document* and once in the *reformulated query*.

Afterwards, vector representations of the *documents* and the *reformulated query* are obtained by normalizing and decomposing the *term-by-document co-occurrence matrix* using a matrix factorization technique called *Singular Value Decomposition* (SVD) [19]. SVD is a form of factor analysis, or more properly the



Documents | Reformulated

Keywords

	ME1	ME2	...	MEN	Query
convert	0	2	...	2	2
failur	0	2	...	5	1
hvac	3	0	...	1	1
...		...	...	...	...

Singular Value Decomposition

Scores

Model Fragment Similitude Scores
ME2 = 0.93
MEN = 0.85
...
ME1 = -0.87

Fig. 3. Latent Semantic Indexing Example using the reformulated query

mathematical generalization of which factor analysis is a special case. In SVD, a rectangular matrix is decomposed into the product of three other matrices. One component matrix describes the original row entities as vectors of derived orthogonal factor values, another describes the original column entities in the same way, and the third is a diagonal matrix containing scaling values such that when the three components are matrix-multiplied, the original matrix is reconstructed.

The relevancy ranking (which can be seen in Fig. 3) is produced according to the calculated similarity degrees. In this example, LSI retrieves 'ME2' and 'MEN' in the first and second position of the relevancy ranking due to *query-documents* cosines being '0.9343' and '0.8524', implying a high similarity degree between the model elements and the reformulated query. On the opposite, the 'M1' model element is returned in a latter position of the ranking due to its *query-document* cosine being '-0.8736', implying a lower similarity degree.

From the ranking of all the model elements, only those model elements that have a similarity measure greater than x must be taken into account. A good heuristic that is widely used is $x = 0.7$. This value corresponds to a $45°$ angle between the corresponding vectors. Even though the selection of the threshold is an issue under study, the heuristic chosen for this work has yielded good results in other similar works [25,33].

Following this principle, the elements with a similarity measure equal or superior to $x = 0.7$ are taken to conform a model fragment, candidate for realizing the feature. Through the example provided in Fig. 3, ME2 and MEN are model elements that conform part of the model fragment obtained by this core for the reformulated query, due to their cosine values being superior to the 0.7 threshold.

3.3 Linguistic rules FL CORE

This core is based on an approach presented by Spanoudakis et al. [38], which is a linguistic rule-based to support the automatic generation of Traceability Links between feature descriptions and models. Specifically, the Traceability Links are generated following two stages:

1. First, a Parts-of-Speech (POS) tagging technique [20] is applied on the feature descriptions that are defined using natural language.
2. Second, the Traceability Links between the feature descriptions and the models are generated through the *description-to-object-model* rules.

The *description-to-object-model* (DTOM) rules are specified by investigating grammatical patterns in feature descriptions. Moreover, the DTOM rules are based on two kinds of relations between feature descriptions and models. On the one hand, *Overlap* relations are understood as the relation between a sequence of terms in a feature description and a class, attribute, association or association end in model. On the other hand, *Requires_Execution_Of* relations are understood as the relation between a sequence of terms in a feature description and an operation in model.

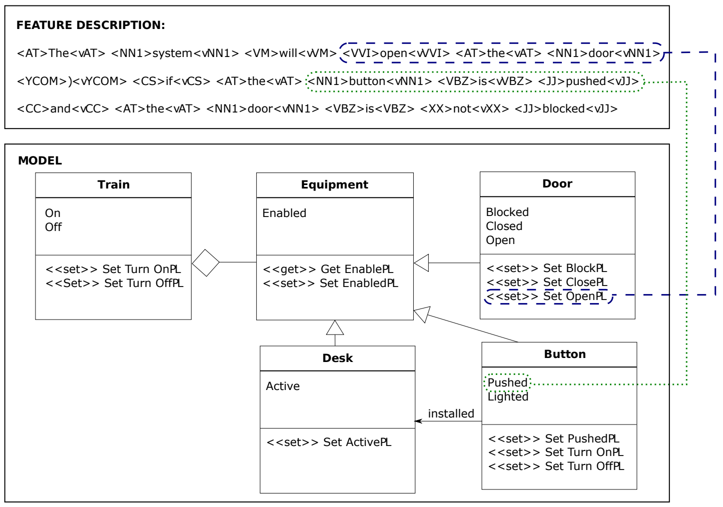


Fig. 4. Example of Traceability Links generation based on DTOM rules

In Fig. 4 the sequence of terms `<NN1>button</NN1>` `<VBZ>is</VBZ>` `<JJ>pushed</JJ>` in the feature description and the attribute `Pushed` of the class `Button` satisfy the conditions of a rule and, as a consequence, an *Overlap* relation would be created between them. In Fig. 4 the sequence of terms `<VVI>open</VVI>` `<AT>the</AT>` `<NN1>door</NN1>` in the feature description and the operation `Set Open` of the class `Door` satisfy the conditions of a rule and, as a consequence, a *Requires_Execution_Of* relation would be created between them.

In [38], the authors propose rules that we apply between a reformulated feature description and a model. Hence, a set of elements of the model are related to the reformulated feature description. These elements compose the model fragment as traceability result.

4 Evaluation

This section presents the research questions that our work tackles, the data set of our real-world case study, the implementation details, the planning and execution, the results and the statistical analysis.

4.1 Definition

We aim to answer the following research questions:

RQ$_1$: *Does the collaboration produce an improvement in the existing FL approaches that obtain the best results (IR and Linguistic rules)?*

RQ$_2$: *If a positive answer in RQ_1, how much is the quality of the solution improved on IR?*

RQ$_3$: *If a positive answer in RQ_1, how much is the quality of the solution improved on Linguistic rules?*

The first research question investigates the results of our approach using IR and Linguistic rules, the results of IR, and the results of Linguistic rules. While, the second and third question investigates the improvement of introducing collaboration in IR and Linguistic rules, respectively.

4.2 Data set

Our industrial partner, CAF, is an international provider of railway solutions all over the world that can be seen in different types of trains (regular trains, subway, light rail, monorail, etc.). The data set that CAF provided us is made up of 23 trains where each product model on average is composed of more than 1200 elements. They are built from 121 different features that can be part of a specific product model.

Furthermore, CAF provided us with the model fragments of 43 features from different trains. Nineteen domain experts from CAF were involved in obtaining different descriptions for each feature. Also, CAF provided us with lists of domain terms and stopwords to process the NL. The domain terms list has around 300 domain terms, and the stopwords list has around 60 words. Both lists were created by CAF domain experts who are associated with the provided products.

4.3 Implementation details

We have used the Eclipse Modeling Framework to manipulate the models and CVL [14] to manage the model fragments. The techniques used to process the NL have been implemented using OpenNLP [1] for the POS-Tagger and the English (Porter2) stemming algorithm [3] for the stemming algorithm (originally created using snowball and then compiled to Java). The LSI has been implemented using the Efficient Java Matrix Library (EJML [2]).

4.4 Planning and execution

Fig. 5 shows an overview of the process that was planned to answer the research questions taking as input both the documentation from our industrial partner, and the 43 feature descriptions and their self-rated confidence for each of the 19 domain experts from our industrial partner, who were involved.

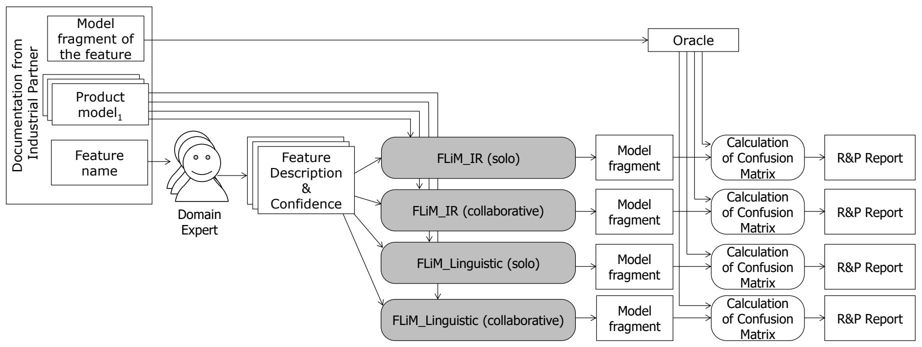

Documentation from Industrial Partner
Model fragment of the feature
Product model$_1$
Feature name
Domain Expert
Feature Description & Confidence
Oracle
FLiM_IR (solo)
FLiM_IR (collaborative)
FLiM_Linguistic (solo)
FLiM_Linguistic (collaborative)
Model fragment
Calculation of Confusion Matrix
R&P Report

Fig. 5. Evaluation process

First, we execute two variants (solo and collaborative) for each FL core (depicted as shaded boxes in Fig. 5):

- **FLiM_IR (solo):** this variant uses the IR FL core to locate the model fragment that realizes the feature description provided by each of the 19 domain experts for each of the 43 features, i.e., 43 (features) x 19 domain experts' feature descriptions = 817 independent runs.
- **FLiM_IR (collaborative):** this variant enables that different domain experts collaborate to locate the model fragment that realizes each of the 43 target features (as described in Section 3.1) using the IR FL core. Specifically, we set k=5 (i.e., one domain expert's feature description is set as base query and five domain experts' feature descriptions are set as relevant documents for the location of the target features). This decision was made based on recommendations found in the literature [5].

- **FLiM_Linguistic (solo):** this variant uses the Linguistic rules FL core to locate the model fragment that realizes the feature description provided by each of the 19 domain experts for each of the 43 features, i.e., 43 (features) x 19 domain experts' feature descriptions = 817 independent runs.
- **FLiM_Linguistic (collaborative):** this variant not only uses the Linguistic rules FL core but also, it enables that different domain experts collaborate to locate the model fragment that realizes each of the 43 target features (as described in Section 3.1) by setting k=5.

When a variant is executed, we obtain as a result a model fragment that realizes a target feature. Then, we compare the model fragment with an oracle as Fig. 5 shows. The oracle is prepared using the model fragments that realize each target feature provided by our industrial partner. The oracle will be considered the ground truth and will be used to calculate a confusion matrix.

The confusion matrix is a table that is often used to describe the performance of a classification model on a set of test data (the solutions) for which the true values are known (from the oracle). In our case, each solution obtained is a model fragment that is composed of a subset of the model elements that are part of the product model. Since the granularity is at the level of model elements, each model element presence or absence is considered as a classification. The confusion matrix distinguishes between the predicted values and the real values classifying them into four categories: True Positive (TP), False Positive (FP), True Negative (TN) and False Negative (FN).

Finally, some performance measurements are derived from the values in the confusion matrix. Specifically, we create a report for the confusion matrix including three performance measurements: recall, precision, and F-measure.

Recall measures the number of elements of the solution that are correctly retrieved by the proposed solution and is defined as follows:

$$Recall = \frac{TP}{TP + FN}$$

Precision measures the number of elements from the solution that are correct according to the ground truth (the oracle) and is defined as follows:

$$Precision = \frac{TP}{TP + FP}$$

F-measure corresponds to the harmonic mean of precision and recall and is defined as follows:

$$F - measure = 2 * \frac{Precision * Recall}{Precision + Recall}$$

Recall values can range from 0% (which means that no single model element obtained from the oracle is present in the solution) to 100% (which means that all the model elements from the oracle are present in the solution). Precision values can range from 0% (which means that no single model fragment from the solution is present in the oracle) to 100% (which means that all the model fragments from the solution are present in the oracle). A value of 100% precision and 100% recall implies that both the solution and the oracle are the same.

4.5 Results

Table 1 shows the mean values of recall, precision, and the F-measure. In terms of recall, FLiM_IR (solo) obtains the best result, providing a precision value of 70.28%. The second best results are obtained by the collaborative approaches (an average value of 59.19% in FLiM_IR (collaborative) and 57.13% in FLiM_Linguistic (collaborative)). The worst result is obtained by FLiM_Linguistic (solo), which obtains an average value of 24.64%. In terms of precision, the collaborative approaches obtain the best results. FLiM_IR (collaborative) obtains the best results in precision, providing an average value of 68.24%, whereas FLiM_Linguistic (collaborative) obtains an average value of 54.49%. FLiM_Linguistic (solo) provides an average value of 33.95%, whereas the worst results are obtained by FLiM_IR (solo) (16.51%). In terms of the F-measure, the collaboration among domain experts improves the results in IR and Linguistic (up to 37.19 and 29.31, respectively).

Table 1. Mean values and standard deviations for Precision, Recall, and F-measure in the industrial case study

	Recall ± (σ)	Precision ± (σ)	F-measure ± (σ)
FLiM_IR (solo)	70.28±15.93	16.51±10.86	24.81±14.15
FLiM_IR (collaborative)	59.19±13.99	68.24±14.32	62.00±11.35
FLiM_Linguistic (solo)	24.64±13.61	33.95±15.59	25.00±12.42
FLiM_Linguistic (collaborative)	57.13±13.37	54.49±12.89	54.31±9.90

4.6 Statistical analysis

To answer whether the collaboration produce an improvement (RQ_1), we compare the solo variant with the collaborative variant for the two cores (IR and Linguistic rules). To properly compare the variants, all of the data was analyzed using statistical methods to provide formal and quantitative evidence (statistical significance). The statistical tests provide a probability value, $p - value$. The $p - value$ obtains values between 0 and 1. It is accepted by the research community that a $p - value$ under 0.05 is statistically significant.

To compare the variants, we carry out a Holm's post hoc analysis, which performs a pair-wise comparison among the results of each variant. The $p - Values$ of Holm's post hoc analysis are smaller than the corresponding significance threshold value (0.05) in the comparison between the solo and collaborative variants for each of the two cores.

RQ_1 answer. From the results, we can conclude that introducing collaboration when locating features in models produces an improvement in terms of solution quality using both the IR and Linguistic rules core.

To answer how much is the quality of the solution improved using IR (RQ_2), and how much is the quality of the solution improved using Linguistic rules

(RQ_3), we perform statistical analysis of the results since statistically significant differences can be obtained even if they are so small as to be of no practical value. Hence, it is important to assess if the results of our approach are statistically better than another and to assess the magnitude of the improvement. *Effect size* measures are needed to analyze this. For a non-parametric effect size measure, we use Vargha and Delaney's $\hat{A}_{12}$ [39]. $\hat{A}_{12}$ measures the probability that running one approach yields higher values than running another approach. If the two approaches are equivalent, then $\hat{A}_{12}$ will be 0.5.

Table 2 shows the values of the effect size statistics for our approach in IR and Linguistic rules. The second row of the table shows the comparison that entails IR. FLiM_IR (collaborative) would obtain better results than FLiM_IR (solo) in 44.94% of the runs for recall and 99.68% of the runs for precision.

Table 2. $\hat{A}_{12}$ statistic for each core vs. its collaborative variant

	Recall	Precision
FLiM_IR (collaborative) vs FLiM_IR (solo)	0.4494	0.9968
FLiM_Linguistic (collaborative) vs FLiM_Linguistic (solo)	0.9799	0.9237

RQ_2 answer. From the results, we conclude how much the solution of the quality is improved introducing collaborative in IR. Although the value for recall is near to be statistically equivalent, the value for precision shows a pronounced superiority to perform collaborative feature location in models (99.68% of the runs would obtain better results for precision introducing collaboration).

The third row of Table 2 shows the comparison that entails Linguistic rules, which obtains the largest differences. FLiM_Linguistic (collaborative) would obtain better results than FLiM_Linguistic (solo) in 97.99% of the runs for recall and 92.37% of the runs for precision.

RQ_3 answer. The results confirm that introducing collaboration to locate features in models using Linguistic rules has a pronounced superiority since more than 92% of the runs would obtain better results.

5 Threats to validity

We use the classification of threats of validity of [32,42], which distinguishes four aspects of validity to acknowledge the limitations of our evaluation.

Construct validity: To minimize this risk, our evaluation is performed using three measures: precision, recall and F-measure. These measures are widely accepted in the software engineering research community [35].

Internal Validity: We used an oracle (obtained from our industrial partner and considering the ground truth) to evaluate our approach using feature descriptions or reformulated feature descriptions as queries where the expected solution was known beforehand. By doing so, we were able to compute the recall, precision and F-measure. With regard to the number of relevant documents and terms used to expand the query, we used the values of 5 and 10, respectively as recommended in the literature [5]. However, we do not know at this stage how using different values would impact the results.

External Validity: In order to mitigate this threat, our approach has been designed to be applied not only to the domain of our industrial partner but also, to different domains. The requisites to apply our approach are that the set of models where features have to be located conform to MOF (the OMG metalanguage for defining modeling languages), and the query must be provided as a textual description.

Furthermore, query reformulation techniques can only work if the original query is reasonably strong to retrieve at least some of the relevant documents [37]. As occurs in other works [37,15], results depend on the quality of the queries. Poor queries assign high rank to irrelevant model fragments. It is also worth noting that the language used for the textual elements of the models and the feature descriptions in the query provided must be the same. This language is particular for each domain.

Hence, despite our approach can be applied to locate features on MOF-based models from different domains, our approach should be applied to other domains before assuring its generalization.

Reliability: To reduce this threat, the feature descriptions and the product family are provided by our industrial partner, who is not involved in this research.

6 Related work

Several approaches have been proposed to reformulate queries in a semi-automatic or automatic way by expanding the query of a user [36] based on relevant documents such as source code and Internet sites. For example, Yang and Tan [43] reformulate the query by extracting synonyms, antonyms, abbreviations, and related words from the source code. Rivas et al. [31] add relevant terms from a scientific documental database to a query to improve the documents initially retrieved. Hill et al. [15] also obtain possible query expansion terms from the code. Lu et al. [22] improve code search by expanding the query with synonyms. Marcus et al. [25] expand the query using LSI in order to determine the terms from the source code that are most similar to the query. Other approaches expand the query by adding information from external sources of information such as public repositories [8].

Table 3 compares the above query expansion works with our work. As the table shows, the base query that is going to be expanded is obtained from a human, who can play different roles (developer, user, analyst, and domain expert). The relevant documents used to find the terms to expand the query are usually source code, online documentation, or text. In contrast, to support collaboration in our work, we use other domain experts' feature descriptions as relevant documents in order to enrich the base query feature description with the knowledge of other domain experts. Moreover, in contrast to the above works, our work aims

Table 3. Comparison with query expansion works

Author	**Base query**	**Relevant documents**	**Industrial domain**	**Artifact**
Yang and Tan [43]	Developer	Source code	No	Code
Rivas et al. [31]	User	Biomedical articles	No	Text
Hill et al. [15]	Developer	Source code	No	Code
Lu et al. [22]	Developer	Internet site	No	Code
Marcus et al. [25]	User	Source code	No	Code
Dimitru et al. [8]	User	Internet sites	No	Product specifications
Our work	Domain expert	Domain experts	Yes	Models

to apply query expansion techniques for introducing collaboration in industry since the context is not the same as in academia [4].

Also, there are many feature location approaches that have been proposed to find features in code by taking textual information as input [7] such as [6,18,40]. Other works such as [41,16,45,44,27,11,26] focus on the location of features in models by comparing the models with each other to formalize the variability among them, whereas Font et al. [10] use an evolutive algorithm to locate features among a family of models. In contrast to these feature location approaches, our work introduces collaboration among different domain experts to locate a target feature in models.

7 Concluding remarks

Although collaboration is a useful and a necessary component in industrial contexts to take advantage of the experience of different domain experts, it is neglected in existing FL approaches. In this paper, we propose an approach that introduces collaboration in two existing FL approaches (IR and Linguistic rules) to locate features in models. To introduce collaboration, the relevant feature descriptions provided by the domain experts are identified using an estimation of confidence level. After, our approach automatically reformulates the relevant feature descriptions in a single query.

The results show that introducing collaboration for locating features in models boosts the quality of the results of existing FL approaches (IR and Linguistic rules). The statistical analysis of the results assesses the magnitude of the improvement of introducing collaboration. Moreover, our results show that our approach can be applied in real world environments.

As future work, we plan to evaluate the influence in the quality of the solution whether the number of domain experts who collaborate changes. In addition, we plan to evaluate the quality of the solution with new approaches based on Machine Learning [24].

Acknowledgements

This work has been partially supported by the Ministry of Economy and Competitiveness (MINECO) through the Spanish National R+D+i Plan and ERDF funds under the project Model-Driven Variability Extraction for Software Product Line Adoption (TIN2015-64397-R).

References

1. Apache opennlp: Toolkit for the processing of natural language text. https://opennlp.apache.org/ (2017)
2. Efficient java matrix library. http://ejml.org/ (2017)
3. English (porter2) stemming algorithm. http://snowball.tartarus.org/algorithms/english/stemmer.htm (2017)
4. Ambreen, T., Ikram, N., Usman, M., Niazi, M.: Empirical research in requirements engineering: trends and opportunities. Requirements Engineering pp. 1–33 (2016)
5. Carpineto, C., Romano, G.: A survey of automatic query expansion in information retrieval. ACM Comput. Surv. 44(1), 1:1–1:50 (Jan 2012)
6. Cavalcanti, Y.a.C., Machado, I.d.C., Neto, P.A.d.M.S., de Almeida, E.S., Meira, S.R.d.L.: Combining rule-based and information retrieval techniques to assign software change requests. In: Proceedings of the 29th ACM/IEEE International Conference on Automated Software Engineering. pp. 325–330. ASE '14 (2014)
7. Dit, B., Revelle, M., Gethers, M., Poshyvanyk, D.: Feature location in source code: a taxonomy and survey. Journal of Software: Evolution and Process 25(1), 53–95 (2013)
8. Dumitru, H., Gibiec, M., Hariri, N., Cleland-Huang, J., Mobasher, B., Castro-Herrera, C., Mirakhorli, M.: On-demand feature recommendations derived from mining public product descriptions. In: Proceedings of the 33rd International Conference on Software Engineering. pp. 181–190. ICSE '11 (2011)
9. Fidel, R., Pejtersen, A.M., Cleal, B., Bruce, H.: A multidimensional approach to the study of human-information interaction: A case study of collaborative information retrieval. J. Am. Soc. Inf. Sci. Technol. 55(11), 939–953 (Sep 2004)
10. Font, J., Arcega, L., Haugen, Ø., Cetina, C.: Feature Location in Model-Based Software Product Lines Through a Genetic Algorithm. In: Proceedings of the 15th International Conference on Software Reuse: Bridging with Social-Awareness (2016)
11. Font, J., Ballarín, M., Haugen, Ø., Cetina, C.: Automating the variability formalization of a model family by means of common variability language. In: Proceedings of the 19th International Conference on Software Product Line (SPLC). pp. 411–418 (2015)
12. Frakes, W.B., Baeza-Yates, R.: Information Retrieval: Data Structures and Algorithms (1992)
13. Hansen, P., Shah, C., Klas, C.P.: Collaborative Information Seeking: Best Practices, New Domains and New Thoughts. Springer Publishing Company, Incorporated, 1st edn. (2015)

14. Haugen, Ø., Moller-Pedersen, B., Oldevik, J., Olsen, G., Svendsen, A.: Adding standardized variability to domain specific languages. In: Software Product Line Conference, 2008. SPLC '08. 12th International. pp. 139–148 (Sept 2008)
15. Hill, E., Pollock, L., Vijay-Shanker, K.: Automatically capturing source code context of nl-queries for software maintenance and reuse. In: Proceedings of the 31st International Conference on Software Engineering. pp. 232–242. ICSE '09, IEEE Computer Society, Washington, DC, USA (2009)
16. Holthusen, S., Wille, D., Legat, C., Beddig, S., Schaefer, I., Vogel-Heuser, B.: Family model mining for function block diagrams in automation software. In: Proceedings of the 18th International Software Product Line Conference: Volume 2. pp. 36–43 (2014)
17. Hulth, A.: Improved automatic keyword extraction given more linguistic knowledge. In: Proceedings of the 2003 conference on Empirical methods in natural language processing. pp. 216–223 (2003)
18. Kimmig, M., Monperrus, M., Mezini, M.: Querying source code with natural language. In: Proceedings of the 2011 26th IEEE/ACM International Conference on Automated Software Engineering. pp. 376–379. ASE '11 (2011)
19. Landauer, T.K., Foltz, P.W., Laham, D.: An Introduction to Latent Semantic Analysis. Discourse processes 25(2-3), 259–284 (1998)
20. Leech, G., Garside, R., Bryant, M.: Claws4: the tagging of the british national corpus. In: Proceedings of the 15th conference on Computational linguistics-Volume 1. pp. 622–628. Association for Computational Linguistics (1994)
21. Liu, D., Marcus, A., Poshyvanyk, D., Rajlich, V.: Feature location via information retrieval based filtering of a single scenario execution trace. In: Proceedings of the Twenty-second IEEE/ACM International Conference on Automated Software Engineering. pp. 234–243. ASE '07, ACM, New York, NY, USA (2007)
22. Lu, M., Sun, X., Wang, S., Lo, D., Duan, Y.: Query expansion via wordnet for effective code search. In: 2015 IEEE 22nd International Conference on Software Analysis, Evolution, and Reengineering (SANER). pp. 545–549 (March 2015)
23. Manning, C.D., Raghavan, P., Schütze, H., et al.: Introduction to Information Retrieval, vol. 1. Cambridge University Press (2008)
24. Marcén, A.C., Pérez, F., Cetina, C.: Ontological evolutionary encoding to bridge machine learning and conceptual models: Approach and industrial evaluation. In: Proceedings of the 36th International Conference on Conceptual Modeling (2017)
25. Marcus, A., Sergeyev, A., Rajlich, V., Maletic, J.I.: An information retrieval approach to concept location in source code. In: Proceedings of the 11th Working Conference on Reverse Engineering. pp. 214–223. WCRE '04 (2004)
26. Martinez, J., Ziadi, T., Bissyand, T.F., Klein, J., l. Traon, Y.: Automating the extraction of model-based software product lines from model variants (t). In: Automated Software Engineering (ASE), 2015 30th IEEE/ACM International Conference on. pp. 396–406 (Nov 2015)
27. Martinez, J., Ziadi, T., Bissyandé, T.F., Klein, J., Traon, Y.L.: Bottom-up adoption of software product lines: a generic and extensible approach. In: Proceedings of the 19th International Conference on Software Product Line. pp. 101–110 (2015)
28. Meziane, F., Athanasakis, N., Ananiadou, S.: Generating Natural Language Specifications from UML Class Diagrams. Requirements Engineering 13(1)
29. Poshyvanyk, D., Gueheneuc, Y.G., Marcus, A., Antoniol, G., Rajlich, V.: Feature location using probabilistic ranking of methods based on execution scenarios and information retrieval. IEEE Transactions on Software Engineering 33(6), 420–432 (Jun 2007)

30. Revelle, M., Dit, B., Poshyvanyk, D.: Using data fusion and web mining to support feature location in software. In: IEEE 18th International Conference on Program Comprehension (ICPC). pp. 14–23 (June 2010)
31. Rivas, A., Iglesias, E., Borrajo, L.: Study of query expansion techniques and their application in the biomedical information retrieval. The Scientific World Journal (2014)
32. Runeson, P., Höst, M.: Guidelines for conducting and reporting case study research in software engineering. Empirical software engineering 14(2), 131–164 (2009)
33. Salman, H.E., Seriai, A., Dony, C.: Feature location in a collection of product variants: Combining information retrieval and hierarchical clustering. In: The 26th International Conference on Software Engineering and Knowledge Engineering. pp. 426–430 (2013)
34. Salton, G.: The SMART Retrieval System—Experiments in Automatic Document Processing. Prentice-Hall, Inc., Upper Saddle River, NJ, USA (1971)
35. Salton, G., McGill, M.J.: Introduction to Modern Information Retrieval (1986)
36. Shah, C.: Collaborative information seeking: A literature review. Exploring The Digital Frontier Advances In Librarianship 32 (2010)
37. Sisman, B., Kak, A.C.: Assisting code search with automatic query reformulation for bug localization. In: Proceedings of the 10th Working Conference on Mining Software Repositories, MSR '13. pp. 309–318 (2013)
38. Spanoudakis, G., Zisman, A., Pérez-Minana, E., Krause, P.: Rule-Based Generation of Requirements Traceability Relations. Journal of Systems and Software 72(2), 105–127 (2004)
39. Vargha, A., Delaney, H.D.: A critique and improvement of the cl common language effect size statistics of mcgraw and wong. Journal of Educational and Behavioral Statistics 25(2), 101–132 (2000)
40. Wang, S., Lo, D., Jiang, L.: Active code search: Incorporating user feedback to improve code search relevance. In: Proceedings of the 29th ACM/IEEE International Conference on Automated Software Engineering. pp. 677–682. ASE '14 (2014)
41. Wille, D., Holthusen, S., Schulze, S., Schaefer, I.: Interface variability in family model mining. In: Proceedings of the 17th International Software Product Line Conference: Co-located Workshops. pp. 44–51 (2013)
42. Wohlin, C., Runeson, P., Höst, M., Ohlsson, M.C., Regnell, B., Wesslén, A.: Experimentation in software engineering (2012)
43. Yang, J., Tan, L.: Inferring semantically related words from software context. In: Mining Software Repositories (MSR). pp. 161–170 (2012)
44. Zhang, X., Haugen, Ø., Møller-Pedersen, B.: Augmenting product lines. In: Software Engineering Conference (APSEC). vol. 1, pp. 766–771 (Dec 2012)
45. Zhang, X., Haugen, Ø., Moller-Pedersen, B.: Model comparison to synthesize a model-driven software product line. In: Proceedings of the 2011 15th International Software Product Line Conference (SPLC). pp. 90–99 (2011)