International Journal of Engineering Research and General Science Volume 2, Issue 6, October-November, 2014
ISSN 2091-2730

# Survey on ExFeature: A Feature Modelling and Recommending Technique For Domain Oriented Product Listing

Kanifnath S. Hirave
M.E. Computer-II,
Vidya Pratisthan`s College Of Engineering-Baramati,pune

Prof. Dinesh Bhagwan Hanchate
Assistant Professor (Computer Dept.)
Vidya Pratisthan`s College Of Engineering-Baramati,pune

**Abstract**— This The activity of detecting and documenting the similarities and differences in related software products in domain is called as domain analysis. For efficient detection and utilization of commonality across related software systems, the effective software reuse is required. Domain experts providing information about a domain under analysis. Feature oriented domain analysis (FODA) is used for efficient detection and utilization of commonality across related software systems. Reuse of software product is one of the encouraging resolutions to the software disaster. Feature Oriented Reuse Environment (FORM) model constructed during the analysis is called a feature model, it captures commonality as an AND/OR graph. Domain analysis and Reuse Environment (DARE) is CASE tool that supports domain analyst in carrying out a clear domain analysis method. DARE is useful for detection and documenting domain information from documents and program. Recommender systems are used to find affinity among features across products, for which two different techniques are used. First technique uses association rule mining algorithm, in which two algorithms are used i.e. Apriori and AprioriTID. Another technique which is collaborative filtering recommender system is to analyze neighborhoods of similar products to identify new feature of items

**Keywords**— Domain analysis, FORM, FODA, DARE, Collaborative Filtering. Recommendations System,Association Rule Mining.

## INTRODUCTION

The activity of detecting and documenting the similarities and differences in related software products in certain domain is called as Domain Analysis. At the beginning, there is no any methodology for domain analysis, Domain analysis is conducted manually. Domain analysis is carried out with help of data flow diagrams. Domain analysis can be considered as a process which is occurring prior to system analysis. [1]. Organized detection and use of cohesion across related software systems is required for successful software reuse. Domain analysis coffer, a general report of the necessities of that class of structures and a set of methods for their implementation with the help of observing related software systems. FODA [2] will create methods for accomplishment a domain analysis and define the products of the domain analysis process. The important technical condition for completing effective software reuses efficient detection and use of unity across related software systems. In FORM [3] inspection of a class of related systems and the cohesion of primary systems exist. It is possible to achieve a set of reference models. FORM starts with an analysis of agreement among applications in a particular domain in terms of services, operating environments, domain tools. The feature model(FM)[3] is defined as construction during the analysis is called feature model. Feature model captures commonality. Domain Analysis and Reuse Environment[4] is CASE tool which helps in domain analysis of finding and recording the similarities and differences of related software systems. DARE[4] helps to capture of domain information from experts in a domain. Captured domain information is stored in a domain catalog, which is enclosed a general architecture for the domain and domain specific components.

We also studied the problem of finding out association rules amongst items in huge database of sales transactions. There are two algorithms i.e. Apriori and Apriori-TID [5,6] algorithm for Association Rule Mining which is well known algorithm to find Association rules which are used for affinities among items [5,6].The process of estimating items through the views of other people is called as Collaborative filtering (CF)[7] .CF technology fetches organized views of large interrelated publics on the web which supporting filtering of large amounts of data. We studied the very important part of collaborative filtering, its key uses for users of the principle and exercise of CF [7] algorithms. We also studied challenges of a CF recommendation system and evaluation of Collaborative Filtering [8].

## RELATED WORK

G. Arango et al [1], states that domain analysis is information severe activity for which no idea or any sort of formalization is accessible. Domain analysis is directed casually and all reported encounters focus on the result, not on the methodology. Two technologies are broken down after analysis. Domain Analysis transforms in set of data flow diagrams the model distinguishes transitional exercises and work items. The generation of reusable components that can be reused in applications is the central problem. Previously, it was assumed that reusable components were readily available. Available components are hard to understand and adjust to new applications, Instead software components are available for reuse then programmers have chosen to create their own. Domain analysis as important step in making real reusable components have exposed better success in reusability. Components which are the result of domain analysis are better suited for reusability because these components capture the vital functionality required in that particular domain. So that, developers catch them easier to add new systems. Analysis is very important issue in the victory of reusability. In this technique recommend, development and validation of model domain analysis (DA)process.

Output of domain analysis supports systems. The output of system analysis supports the designer's tasks. In the predictable waterfall model of software development, The task of systems designers is to produce a particular design from a set of requirements and specifications. The system analyst task is to create model of an current system and propose replacements for automation or improvements. Both activities focus on a specific model for particular system. In DA authors tried to simplify all systems in an applications, DA is at a higher level of abstraction than system analysis. In domain analysis, similar features from similar systems are generalized, objects and operations common to all systems within the same domain are identified, and a model is defined to describe their relationships. If this process succeed in identifying the objects and operations in a domain specific language. Language becomes our domain model and is used to describe objects and operations common in that domain.

**Advantages**

- The domain analysis is very simple

**Disadvantages**

- This is manual process for no any methodology is available so it vey time consuming process.

K. Kang et al [2] The efficient finding and utilization of commonality across related software systems is an important technical requirement for achieving successful software reuse. By inspecting a associated software systems and the mutual essential theory of those systems, domain analysis can offer a reference model for describing the class. The main aim of this report is to establish the Feature-Oriented Domain Analysis (FODA) is used for accomplishment a domain analysis. The feature-oriented idea is based on the importance of the method on finding those features a user commonly assumes in applications in a domain. This method is depend on a study of other domain analysis methodologies which defines both the products and the process of domain analysis. This thesis is focused toward three groups:

1. Domain experts give detail information about a domain.
2. Domain analysts performing the domain analysis.
3. Systems analysts and developers are the Consumers of domain analysis products.

For establishing requirements for software reuse domain analysis is a necessary
The analysis can help a variety of purposes toward these end specifications for reusable resources which imparts.

- tool support such as catalogs
- Process model for reuse
- In overall, the analysis provides a complete overview of the problems which is solved by software in a given domain.

The Feature-Oriented Domain Analysis (FODA) method is built on the consequences of other successful domain analysis endeavors. The method founds three stages of a domain analysis:
Context analysis: In order to establish scope.
Domain modeling: In order to define the problem space and
Architecture modeling: In order to characterize the solution space

**Advantages**

- Communication language among participants
- Easy to capture similarities & difference of requests on feature oriented models
- Representative of application

**Disadvantages**

- There is a no any organized method for use of reusable assets.
- More man power is required for domain analysis.

K.C. Kang et al [3] Reuse of software product is one of the most favourable solutions to the software disaster. Previous work for efficient reuse i.e. Feature oriented domain analysis (FODA) which focuses on creating reusable assets. FORM model is constructed during the analysis is called a feature model, it captures commonality as an AND/OR graph, where AND nodes indicate mandatory features and OR nodes indicate alternative features selectable for different applications. The goals of FORM are as follows,
Method to create reusable software artifact effectively
Method to develop new application from reusable artifacts
Domain analysis can be well-defined as examining a class of related systems and mining the commonalities and differences of these systems. Feature oriented domain analysis can be defined as doing domain analysis based on features. Domain engineering can be defined as using analysis results to create a set of reference models, Creating reusable software architectures and components. The purpose of the FORM is to creating a set of reusable assets and also increasing new application from the assets. Feature model[3] is studied and advanced to cover non-functional features and to support complete architecture.

**Advantages**

- Focus on reuse concept and introduce practical method.

**Disadvantages**

- Meaning of feature is not clearly fixed among researches.
- FORM is semiformal.
- Provide construction language but not demanding analysis of Models.
- More man power is required for domain analysis.

W. Frakes, R. Prieto-Diaz, and C. Fox[4] Software engineering has usually concerned with the development of custom built single system. In order to support single system engineering, software engineering tools and methods have been developed. Modern software engineering practice requires a move from creating one at a time to the use of formal engineering principles for generating system and tactics. The practice of preplanned reuse of systematic software gives way for accomplishing such transition. Systematic software reuse is depend upon the observation which helps to increasing quality and productivity meaningfully by moving the attention of software engineering to domain centered view, and which helps to identify that most software industries do not build totally new software applications. Systematic reuse is the important part in quality and productivity improvements lies accepting a process for building multiples related systems in a problem domain. Infrastructure support is required for systematic reuse. Domain engineering is the activity of making an infrastructure to support systematic reuse. In domain engineering various activities are involved such as manufacturing plant, producing software applications. Domain engineering is divided into two phases
1.Domain analysis.
2.Domain Implementation
1.The activity of detecting and documenting the similarities and differences in related software products in domain is called as Domain analysis
There are two types of domain analysis methods
A. Bottom-up analysis
In Bottom-up analysis validation of basic architecture and features through analysis of documents and source code.
B. To-down analysis
In Top-down analysis propose common architecture and features depend upon experience and knowledge of domain experts.
The use of information fetched in domain analysis is to grow reusable components for domain and creation of production process for systematically reusing components to produce new systems. Domain analysis and Reuse Environment (DARE) is CASE tool that provide domain analyst in carrying out a clear domain analysis method. DARE is useful for detection and recording domain information from documents and code.

**Advantages**

- DRAE overcome the limitations of previous domain analysis methods such as FODA[3] and FORM[4]
- DARE is automatic process.
- DARE provides method which is pointing on the extraction of high level domain information from domain experts.

**Disadvantages**

- DARE focus on their efforts only on small software requirements specifications.
- The scope of available specifications in extracted features is minimum.

R. Agrwal et al [5,6] states that the algorithm named association rule mining to discover affinities among items. association rule problem has a formal statement of the association rule Mining Definition 1: Let I ={$IT_1$ $IT_2$, … , $IT_n$} be a set of n dissimilar attributes. Let DB be a database, where each record TN has a distinctive identifier, and has a set of items such that TN⊆IT An association rule is an consequence of the form A⇒B, where A, B⊂IT, are sets of items called item sets, and A∩ B=φ. Here, A is called ancestor, and B successive. Two important measures for association rules, support (s) and confidence (α), can be defined as follows.
Definition 2: The support (s) of an association rule is the ratio (calculated in percentage) of the records that contain A ∪ B to the total number of records in the database So, if we say that the support of a rule is 15% then it means that 15% of the total records enclose A ∪ B. Support is the statistical meaning of an association rule. Mathematically support represented as,

$$P(A,B) = \frac{Number\ of\ Transaction\ contaning\ both\ A\ and\ B}{Total\ Number\ of\ Transaction}$$

Definition 3: For a given number of records, confidence (α) is the ratio (which is calculated in form of precent) of the number of records that contain A∪B to the number of records that contain X. Thus, if we say that a rule has a confidence of 95%, it means that 95% of the records containing A also contain B. The confidence of a rule indicates the amount of correspondence in the dataset between A and B, mathematically support is represented as

$$P(A,B) = \frac{Number\ of\ Transaction\ contaning\ both\ A\ and\ B}{Number\ of\ \ Transaction\ containing\ A}$$

The Apriori algorithm [5,6] produced for an incredible accomplishment in the past of mining affiliation principle. It is far the most well -known affiliation principle algorithm. This method utilizes the property that any subset of a huge item set must be an extensive item set. Additionally, it is expected that item in a item set are kept in lexicographic request. The imparted item sets are stretched out to other individual items in the exchange to produce competitor item sets. On the other hand, those individual items may not be substantial. As we realize that a superset of one substantial item set and a little item set will bring about a little item set, these methods create an excess of applicant item sets which end up being little. The Apriori algorithm addresses this paramount issue. The Apriori creates the applicant item sets by joining the expansive item sets of the past pass and erasing those subsets which are little in the past pass without considering the exchanges in the database. By just considering expansive item sets of the past pass, the quantity of hopeful extensive item sets are fundamentally diminished. In the first pass, the item sets with stand out items are checked. The found expansive item sets of the first pass are utilized to create the competitor sets of the second pass utilizing the apriori_gen () [5][6] capacity. After the competitor item sets are discovered, their backings are checked to find the huge item sets of size two by filtering the database. In the third pass, the huge item sets of the second pass are considered as the applicant sets to find vast item sets of this pass. This iterative methodology ends when no new huge item sets are found. Each one pass i of the calculation filters the database once and decides extensive item sets of size i. $L_i$ indicates vast item sets of size i, while Ci is competitors of size i. The apriori_gen() work as follows
1. While performing the first step, Lk-1 is joined with itself to obtain Ck. ( competitors of size k)
2. In the second step, apriori_gen() deletes all item sets from the join result, which have some (k-1) subset that is not in Lk-1. Then, it returns the remaining large k-item sets.[5,6]

**Advantages of Apriori Algorithm**

- It uses large item set property
- Apriori is easily parallelized
- It is easy to implement
- The Apriori algorithm implements level-wise search using repeated item property

**Disadvantages of Apriori Algorithm**

- There is too much database scanning to calculate frequent item so it reduces performance.
- It assumes that transaction database is memory resident
- Generation of candidate item sets is expensive in both space and time
- Support counting is expensive because of subset checking, and multiple database I/O scans.

**Apriori-TID Algorithm [5] [6]**

Apriori scans the complete database in every pass to count support. Scanning of the entire database may not be required in all passes. Based on this estimation**,** proposed another algorithm called Apriori-TID. Similar to Apriori, Apriori-TID uses the Apriori's candidate generating function to determine candidate item sets before the start of a pass. Apriori-TID does not use the database for counting support after the first pass. Slightly, it uses an encoding of the candidate item sets used in the previous pass denoted by *Ck.*Each member of the set *Ck* is of the form *<TID, Xk>* where *Xk* is a potentially large k-item set present in the transaction with the identifier *TID.* In the first pass, $C_1$ links to the database. Though, each item is replaced by the item set. In other passes, the member of *Ck* corresponding to transaction T is *<TID, c>* where *c* is a candidate belonging to *Ck* contained in *T.* Therefore, the size of *Ck*may be smaller than the number of transactions in the database. Still, each entry in $C_k$may be smaller than the corresponding transaction for larger k values. This is because limited candidates may be enclosed in the transaction. It should be mentioned that each entry in*Ck*may be larger than the corresponding transaction for smaller *k* values At first, the entire database is scanned and $C_1$is obtained in terms of item sets. That is, each entry of $C_1$has all items along with *TID.* Large item sets with 1-item $L_1$ are calculated by counting entries of $C_1$ . Then, apriori_gen () is used to obtain $C_2$. Entries of $C_2$ corresponding to a transaction *T* is obtained by considering members of *C2* which are present in *T.* To perform this task, $C_1$ is scanned rather than the entire database. Afterwards, $L_2$ is obtained by counting the support in $C_2$. This process continues until the candidate item sets are found to be empty.

**Advantage of Apriori-TID Algorithm**

- It uses encoding function later passes the size of the encoding function becomes smaller than the database, thus saving much reading effort.

**Disadvantage of Apriori-TID Algorithm**

- Encoding function is more complex

J. Schafer, D. Frankowski, J. Herlocker, and S. Sen[7] Collaborative Filtering is the process of estimating items using the views of other peoples. Collaborative filtering approaches depend upon the ratings of user. The term user tells that any individual who provides ratings to a system. Generally we use this term to refer to the people using a system to receive information such as recommendations though it, also refers to those who provided the data used in creating this information systems to determine the quality of items. Items can consist of anything for which a human can provide a rating, such as art, books, CDs, journal articles, or vacation destinations

Ratings in a collaborative filtering system can take on a following of forms.

- Scalar ratings can consist of either numerical ratings, such as the 1-5 stars provided in Movie Lens system[15] which is used for CF system for Movie or ordinal ratings such as strongly agree, agree, neutral, disagree, strongly disagree.
- Binary ratings model choices between agree/disagree or good/bad.
- Unary ratings are used to indicate that a user has examined or purchased an item, or otherwise rated the item positively. The absence of a rating indicates that we have no information relating the user to the item

Ratings may be collected through explicit means, implicit means, or both. If user is asked to provide an opinion on an item is called as explicit ratings. If ratings concluded from a user's actions is called as implicit ratings. For example, a user who brows a product page possibly has some interest in that product while a user who afterwards purchases the product may have a much stronger interest in that product. The vital idea is that the rating of user u for a new item i is expected to be similar to that of another user v, if u and v have rated other items in a similar way. Likewise, u is likely to rate two items i and j in a similar fashion, if other users have given similar ratings to these two items. Also, collaborative recommendations are based on the quality of items as calculated by peers. Finally, collaborative filtering ones can recommend items with very different content, as long as other users have already shown interest for these different items.
Collaborative filtering methods can be g classified in the two general classes
1.Neighborhood methods
2. Model-based methods.

**Neighborhood method.**
In neighborhood based memory-based or heuristic-based collaborative filtering the user-item ratings stored in the system are directly used to predict ratings for new items. This can be done in two ways
user based

**User based recommendation**
User-based systems, such as evaluate the interest of a user u for an item using the ratings for this item by other users, called neighbors that have similar rating patterns. The neighbors of user u are typically the users v whose ratings on the items rated by both u and v, i.e. Iuv, are most correlated to those of u.

**Item-based recommendation**
Item-based approach on the other hand, predict the rating of a user u for an itemi based on the ratings of u for items similar to i. In such approaches, two items aresimilar if several users of the system have rated these items in a similar fashion.

$$Pred(u,i) = \frac{\sum_{n \in neighbors} \gamma_{ni}}{No\ of\ neighbors}$$

Where,
u is User,
` I is item,
$\gamma_{ni}$ is neighbor n`s rating item

**Model-based methods.**
In contrast to neighborhood based systems, which use the stored ratings directly in the prediction, model-based approaches use these ratings to learn a predictive model. The general idea is to model the user-item interactions with factors representing latent characteristics of the users and items in the system, like the preference class of users and the category class of items. This model is then trained using the available data, and later used to predict ratings of users for new items.

**Advantages of neighborhood-based Collaborative filtering method.**
**1.Simplicity**
Neighborhood-based methods are in-built and quite simple to implement.in their simplest form.
**2. Justifiability**
This method also provides a small and normal justification for the computed predictions.
**3. Efficiency**
One of the strong points of neighborhood-based systems are their Efficiency. They require cheap training phases, which need to be carried at frequent intervals in large commercial applications.
**4. Stability**
Another useful property of recommender systems based on this approach is that they are little affected by the constant addition of users, items and ratings, which are typically observed in large commercial applications.

**Disadvantages of Collaborative filtering method**

**1. Privacy**

In order to provide personalized information to users,CF system must know things about the user.so privacy is big challenge.

**2. Security**

Even system that maintains the security of users ratings can be exploited to reveal personal information.

**3. Trust**

Recommender system may break when malicious users give rating that is not representative of their true preferences.

J. Herlocker et al[8] states that, recommender systems have been evaluated in many ways. In this artifact, the main focus on the key decisions in estimating collaborative filtering recommender systems the user tasks being estimated, the types of analysis and datasets being used, the ways in which prediction quality is measured, the estimation of prediction attributes other than quality, and the user-based evaluation of the system as a whole.

H. Dumitru et al[13] states that, in order to modeling and recommending product features in particular domain, a recommender system is used. This approach mines product descriptions from openly existing online specifications, utilizes text mining[12] and a new IDC algorithm to find out domain-specific features, generates a probabilistic feature model that represents unities, variants, and cross-category features, and then uses association rule mining [5,6] and the k-Nearest- Neighbor (kNN)[7] learning strategy to design a content-based recommender algorithm. This approach has been shown to perform well in the previous work in forum recommendations [7,9,10]. The kNN algorithm computes a feature-based similarity measure between a new product and each existing product. The top k most similar products are considered neighbors of the new product. It then uses information from these neighbors to infer other potentially interesting features and to make recommendations. product similarity
*productsim(p,n)* between a new product p and each existing product n is computed using the binary equivalent of the cosine similarity as follows

$$productsim(p,n) \quad \frac{\sqrt{F_{p \cap F_n}}}{\sqrt{|F_p||F_n|}}$$

where Fp denotes the set of features of product p [12]. This metric generates numbers between 0 and 1, where products with identical features score 1, and products with no common features score 0. *productSim(p, n)* is computed between the new product and all previously mined products. A neighborhood is then computed for the new product p by selecting the top k most similar products.

**Advantages**

- Potentially increasing opportunities for reuse.
- Reducing time-to-market.
- Delivering more competitive software products.

**Disadvantages**

- This recommender system supports the relatively labor-intensive task of domain analysis.

J. Sandvig et al [15] states that open nature of collaborative recommender systems has a security problem. Standard memory-based collaborative filtering algorithm[14], such as k-nearest neighbor, have been shown quite vulnerable to such attacks. Model-based techniques have shown different degrees of improvement over kNN with respect to robustness in the face of profile injection attacks. Here we examine the robustness of a model-based recommendation algorithm based on the data mining technique of association rule mining particularly in Apriori algorithm [5,6]gives substantial enhancement in steadiness and robustness compared to k-nearest neighbor and other model-based techniques.[7]

**Advantages**

- It has more robustness i.e. There is no any vulnerability attack.

Chuan Duan and Jane Cleland-Huang[17] states that, automated trace tools dynamically produce links amongst several software articles such as requirements, design elements, code, test cases. Trace algorithms usually make use of information retrieval methods[11] to calculate similarity scores between pairs of artifacts. The similarity score between any pair of artifacts $a_1$ and $a_2$ is then computed based on the similarity $s(a_1,a_2)$ of their vectors as follows,

$$S(a1,a2)=pr(a2|a1)=|\sum_{i=1}^{m} pr(a2|t_i)pr(a_{1,}t_i)|pr(a_{1,})$$

Where ,

K= No. of Clusters
i =No. of artifacts

$pr(a2|ti)$ is computed as $= pr(a_{1,}t_i)=f_{2i} / \sum_k f_{2k}$ and estimates the extent to which the information in $t_i$describes the concept $a_2$ in respect to the total number of terms in the artifact, while $pr(a_1,t_i)$ is computed as $pr(a_1,t_i) = f1i / n_i$, where $n_i$represents the number of artifacts in the collection containing the term $t_i$. Finally, $pr(a_1)$ is computed as $pr(a_1)=\sum_i pr(a_1,t_i)$. During the clustering phase, similarity scores are computed for each potential pair of artifacts.

**Advantages**

- The technique is fully automated in java so it is more secured.

**Disadvantages:**

- This method is able to evaluate cluster based tracing in a broader set of artifact types, and in larger and more complicated datasets

V. Alves et al[19] states that domain analysis includes not only looking at standard requirements documents such as use case specifications but also at customer information packets. Considering diagonally every documents and deriving, in a practical and scalable way, a feature model that is comprised of coherent abstractions is a fundamental and non-trivial challenge. Here we focus on conducting an exploratory study to examine the appropriateness of Information Retrieval (IR) techniques for scalable identification of commonalities and variability's in requirement specifications for software product lines.

M. Acher et al [20] states that domain analysis is the process of evaluating related products to recognize their corporate and movable features in product line engineering. This procedure is normally carried out by experts on the basis of existing product descriptions, which are expressed in a more or less organized way. Demonstrating and reasoning about product descriptions are error-prone and time consuming tasks. Feature models[3] constitute popular means to specify product similarities and differences in a compressed way, and to provide computerized support to the domain analysis process. Here main focus on simplification the change from product descriptions stated in a tabular format to feature model accurately representing them. This procedure is parameterized through a committed language and high-level directives i.e. features scoping. Here assurance that is the resulting feature model represents the set of permitted feature groupings supported by the considered products and has a clear tree hierarchy together with variability information.

S. Apel et al [21] states that feature-oriented software development (FOSD) is a model for the building, customization and combination of significant software systems. The main focus of this article is to give an outline and a personal view on the roots of FOSD, relations to other software development paradigms, and current expansions in this field. The aim is to point to relations between different lines of research and to identify open issues.

## ACKNOWLEDGMENT

I express great many thanks to Prof. D.B Hanchate for his great effort of supervising and leading me, to accomplish this fine work. Also to college and department staff, they were a great source of support and encouragement. To my friends and family, for their warm, kind encourages and loves. To every person gave us something too light my pathway, I thanks for believing in me.

## CONCLUSION

In this paper we studied domain analysis is the process of identifying, organizing, analyzing, and modeling features common to a particular domain .It is conducted in early phases of the software development life-cycle to generate ideas for a product, discover commonalities and variants within a domain. Most domain analysis techniques, such as the feature-oriented domain analysis (FODA) and feature-oriented reuse method (FORM) depend upon analysts manually reviewing the existing requirement specifications or product brochures and websites, and are therefore quite labor intensive. The success of these approaches is dependent upon the availability of relevant documents and/or access to the existing project repositories as well as the knowledge of the domain analyst. Another approach such as the domain analysis and reuse environment (DARE) which use data mining and information retrieval methods to provide automated support for feature identification and extraction, but tend to focus their efforts on only a small handful of requirements specifications. The extracted features are therefore limited by the scope of the available specifications. In this paper we also focus on recommender systems first technique uses Association Rule Mining algorithm, which uses two algorithms Apriori and AprioriTid, for discovering all significant association rules amongst items in a huge database of transactions. Another technique which is collaborative filtering recommender systems to analyze neighborhoods of similar products to identify new feature of items.

## REFERENCES

[1] G. Arango and R. Prieto-Diaz, Domain Analysis: Acquisition of Reusable Information for Software Construction. IEEE CS Press, May 1989.
[2] K. Kang, S. Cohen, J. Hess, W. Nowak, and S. Peterson, "Feature Oriented Domain Analysis (FODA) Feasibility Study," Technical Report CMU/SEI-90-TR-021, Software Eng. Inst., 1990.
[3] K.C. Kang, S. Kim, J. Lee, K. Kim, G.J. Kim, and E. Shin, "FORM: A Feature-Oriented Reuse Method with Domain-Specific Reference Architectures," Annals of Software Eng., vol. 5, pp. 143-168, 1998.

[4] W. Frakes, R. Prieto-Diaz, and C. Fox, “Dare: Domain Analysis and Reuse Environment,” Annals of Software Eng., vol. 5, pp. 125-141, 1998.
[5] R. Agrawal and R. Srikant, “Fast Algorithms for Mining Association Rules,” Proc. 20th Int’l Conf. Very Large Data Bases,1994.
[6] R. Agrwal and R. Srikant, “Fast Algorithms for Mining Association Rules,” Proc. 20th Int’l Conf. Very Large Data Bases (VLDB ’94), pp. 487-499, Sept. 1994.
[7] J. Schafer, D. Frankowski, J. Herlocker, and S. Sen, “Collaborative Filtering Recommender Systems,” The Adaptive Web, pp. 291-324, Springer, 2007.
[8] J. Herlocker, J. Konstan, L. Terveen, and J. Riedl, “Evaluating Collaborative Filtering Recommender Systems,” ACM Trans. Information Systems, vol. 22, pp. 5-23, 2004.
[9]C. Castro-Herrera, C. Duan, J. Cleland-Huang, and B. Mobasher. A recommender system for requirements elicitation in large-scale software projects.Proc. of the 2009 ACM Symp.on Applied Computing, pages 1419–1426, 2009.
[10] K. Chen, W. Zhang, H. Zhao, and H. Mei.An approach to constructing feature models based on requirements clustering. Requirements Engineering, IEEE International Conference on, 0:31–40, 2005.
[11]C.D. Manning, P. Raghavan, and H. Schutze, Introduction to Information Retrieval. Cambridge Univ. Press, 2008.
[12]E. Spertus, M. Sahami, and O. Buyukkokten. Evaluating similarity measures: a large-scale study in the orkut social network.pages 678–684, Chicago, Illinois, USA, 2005. ACM.
[13]. H. Dumitru, M. Gibiec, N. Hariri, J. Cleland-Huang, B. Mobasher, C. Castro-Herrera, and M. Mirakhorli, “On-Demand Feature Recommendations Derived from Mining Public Software Repositories,”Proc. 33rd Int’l Conf. Software Eng., p. 10, May 2011
[14] Breese, J.S., Heckerman, D., Kadie, C.: Empirical analysis of predictive algorithms for collaborative filtering. In: Proc. of the 14th Annual Conf. on Uncertainty in Artificial Intelligence, pp. 43–52. Morgan Kaufmann (1998)
[15]J. Sandvig, B. Mobasher, and R. Burke, “Robustness of Collaborative Recommendation Based on Association Rule Mining,” Proc. ACM Conf. Recommender Systems, 2007.
[16] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, “Analysis of Recommendation Algorithms for e-Commerce,” Proc. ACM Conf. Electronic Commerce, pp. 158-167, 2000.
[17]Chuan Duan and Jane Cleland-Huang “Clustering Support for Automated Tracing Center for Requirements Engineering” DePaul University 243 S. Wabash Avenue, Chicago IL 60604 312-362-8863{duanchuan,jhuang}@cs.depaul.edu
[18]J. Herlocker, J. Konstan, L. Terveen, and J. Riedl, “Evaluating Collaborative Filtering Recommender Systems,” ACM Trans. Information Systems, vol. 22, pp. 5-23, 2004.
[19] V. Alves, C. Schwanninger, L. Barbosa, A. Rashid, P. Sawyer, P. Rayson, K. Pohl, and A. Rummler, “An Exploratory Study of Information Retrieval Techniques in Domain Analysis,” Proc. 12th Int’l Software Product Line Conf., pp. 67-76, 2008.
[20] M. Acher, A. Cleve, G. Perrouin, P. Heymans, C. Vanbeneden, P Collet, and P. Lahire, “On Extracting Feature Models fromProduct Descriptions,” Proc. Sixth Int’l Workshop Variability Modeling of Software-Intensive Systems (VaMoS ’12), pp. 45-54, 2012.
[21]S. Apel and C. Ka¨stner, “An Overview of Feature-Oriented Software Development,” J. Object Technology, vol. 8, no. 5, pp. 49-84, July/Aug. 2009