Combining configuration and recommendation to enable an interactive guidance of product line configuration

Raouia Triki, Raul Mazo, Camille Salinesi

Triki Raouia, Mazo Raúl, Salinesi Camille. Combining configuration and recommendation to enable an interactive guidance of product line configuration. Les systèmes de recommandation. Editeurs: Ghislaine CHARTRON, Imad SALEH et Gérald KEMBELLEC. Editions Hermès Sciences, collection «Information, hypermédias et communication». Paris-France, 2013

Combining configuration and recommendation to enable an interactive guidance of product line configuration

RAOUIA TRIKI, RAUL MAZO, CAMILLE SALINESI

1. Introduction

E-commerce makes a wide use of recommendation techniques to help customers identify relevant products or services in large collections of offers. Customers' requirements are seamlessly elicited by observing purchase habits, information requests, features of products formerly acquired, etc. A trend in the industry is to go a step further, beyond the selection of pre-defined products from a catalogue by handling products customization. The systems engineering community has shown that, based on product line engineering methods, techniques and tools, it is possible to produce customized products (such as software, cars, machine tools, etc.) efficiently and at low cost. A Product Line (PL) is a family of products that share common characteristics and satisfy the needs of a particular mission [POH 05].The products of a PL have many common features, but also many features that vary from one product to another. The problem is that there are usually so many products in a PL that it is impossible to specify all of them explicitly, and therefore traditional recommendation techniques cannot be simply applied. Scoping on a subset of

products is not a good idea as it impairs all the benefits of customization offered by the PL strategy, which is acquired by reasoning on variability rather than on variants.

A straightforward alternative idea is to let customers configure their products consistently with their requirements, based on a PL specification that describes the common and varying features of all products artefacts, and constraints and dependencies between them. In order to achieve this, marketing and engineers must define upstream models that specify the salient features of reusable artefacts, and the constraints and dependencies between them. Customers just have to select among series of options. Then, correct configurations are prescribed by the configurators that check whether customer choices satisfy the PL constraints and dependencies, and propagate the impacts on open choices that have not yet been made. State of the art approaches go even a step further by offering to let customers specify more complex requirements than just selecting among series of options [DJE 11], [MAZ 12].

Whereas this strategy works quite well on the engineering level, using it in the e-commerce context (e.g. through Web configurators) raises scalability issues [AST 10]. Indeed, customers they want to find out the consequences of their decisions as soon as they make them. This issue is extremely difficult to solve at the industrial level, because the computational complexity of computing full PL configuration grows quadratically [AST 10]. The second issue is the lack of guidance. In practice, choosing from a wide range of options (e.g. a car has typically over a hundred of customer options) is a cumbersome task for customers who don't know where to start, or which feature to choose when there are several alternatives. Besides each time a decision is made, it can be contradictory with former decisions, or have a negative impact on downstream decisions. One doesn't want to disappoint customers by telling them that former decisions should be revised, or that later on choices are not possible anymore because of a decision which consequences were not clear straight ahead. Ultimately, the risk is that customers turn to competitors. Therefore, it is crucial to guide the customer in the PL configuration process.

We are thus faced to a situation where configuration systems and recommendation systems solve two parts of our problem, but not in an integrated way [FAL 11].

Our research goal is to explore the combination of two complementary forms of guidance: recommendation and configuration. We call the approach that combines recommendation with configuration: "Interactive Configuration". The aim of interactive configuration is to inform the customer in real time about desirable/possible/unattainable features according to his/her choices, as well as to suggest decision to focus on, and what choice to make by reasoning with known configurations.

Four key issues should be handled to solve the problem:

- What shall the recommendation apply to? (options, alternatives, requirements, etc)
- Which recommendation technique shall be used? (collaborative filtering, content based filtering, etc)

- What type of data should be used for recommendation? (experiences of past configurations, profiles, contextual data, etc)
- How to combine configuration and recommendation?

This chapter addresses these issues by showing how to apply the “content based” recommendation method [VAN 00] to the “a priori” configuration approach. The recommendation is applied to the product line features and based on textual data.

2. Context

2.1. C*onfiguration*

In the context of PLs, the model that represents the relationships of commonality and variability between the valid products is called a Product Line Model (PLM). One of the most popular notations to specify the common and variable requirements of a product line is the Feature Models (FMs). In the context of PLs, FMs are used for product configuration, variability reasoning and code generation [KAN 90], [VAN 02]. A feature is a prominent or distinctive user-visible aspect, requirement, quality, or characteristic of a software system [KAN 90]. A FM defines the valid combinations of features in a software product line. Several extensions have been proposed to improve and enrich their expressiveness. Two of these extensions are cardinalities [RIE 02], [CZA 05] and attributes [STR 03], [WHI 09]. An attribute is a variable with a name, a domain and a value. The value of each attribute is assigned for each particular configuration. Figure 1 illustrates an example of a feature model.

Figure 1. *Feature Model*

The semantic of the FM's relationships in Figure 1 can be summarized as follows:

–Mandatory: Given two features F1 and F2, where F1 is the father of F2, a mandatory relationship between F1 and F2 means that if F1 is selected in a product, then F2 must be selected too, and vice versa.

–Optional: Given two features F1 and F2, where F1 is the father of F2, an optional relationship between F1 and F2 means that if F1 is selected in a product, then F2 may be selected or not. However, if F2 is selected then F1 must also be selected. For example, in the Fig.1, F5 and F6 are optional child features with respect to the father feature F3.

–Requires: Given two features F1 and F2, a relationship F1 requires F2 means that if F1 is selected in a product then F2 has to be selected as well. Additionally, it

means that F2 can be selected even when F1 is not selected. For example, in the Fig.1, F6 requires F12.

–Exclusion: Given two features F1 and F2, a relationship F1 excludes F2 means that F1 and F2 cannot be selected in the same product. For example, in the Fig.1, F2 excludes F5 and F5 excludes F2: the mutex relation.

–Group cardinality: A group cardinality is an interval denoted <n..m>, with n as lower bound and m as upper bound limiting within a group of features the number of features that can be part of a product. All the features in the group must have the same parent feature, and none can be selected if the parent is not itself selected. For example, in the Fig.1, <1..1> is a group cardinality.

–Constraint: It is a logic relation between features specified in the PLM, and must be satisfied when configuring the PL. For example, in the Fig.1, F9=>NOT(F5^F6).

In Product Line Engineering (PLE), product configuration describes the process of specifying a product according to user-specific needs and the constraints specified in the PLM. The user specifies the features of the product step by step according to his requirements, thus, gradually reducing the search space of the configuration problem.

According to Falkner et al. [FAL 11], a configuration is an instantiation of variables defining the product I={v1=i1, v2=i2, …,vn=in} where ij is one of the elements of dom(vj) which is the variable domain and vj is a domain variable.

In Product Line Engineering (PLE), a Product Line configuration is a methodology for developing a product by reuse from a Product Line Model (PLM) and aiming to satisfy user needs.

For example the following configuration {F1, F2, F3, F4, F7, F9, F11} is correct because it complies with the constraints of the FM in Figure1, unlike the configuration {F1, F2, F3, F4, F7, F9, F10} that is not correct.

2.2. Recommendation

The recommendation is a form of guidance to support users to reach a decision and to find products among a large panel of options and offers reasoning on known configurations [FEL 08].

In the configuration process, users are often in the need of advice to choose which features to select next and which features to select where users lack information and details of the product knowledge. The recommendation can address

these issues and guides the user in the configuration process by predicting the user opinion for a given product.

There are several recommendation techniques. The content-based recommendation is one of them. It treats the recommendation problem as a search for related items [BAL 97]. Given the user's purchased and rated items, the algorithm constructs a search query to find other popular items with the similar keywords or subjects [LIN 03].

The content of the profile depends on the method used in the analysis of document content. To achieve this, several learning techniques were used, such as analytical techniques taken from textual information retrieval for the recommendation of textual documents [BAL 97], [PAZ 97], [MOO 00]. The profile often takes the form of a vector of keywords with weights [1]. The weight associated with each word reflects the importance of this term to the user. These words are often extracted using the TF-IDF measure [SAL 89] [4]. This vector is then compared to that of the document [2]. To achieve this, several measurements can be used such as the measurement vector [3]. The formulas [PAZ 07] of the content-based recommendation are as follows:

[1] $User\_\Pr ofile(P) = \left\{\left(t_j^P, w_j^P\right)\right\}, j = 1..k$

[2] $Document(D) = \left\{\left(t_i^D, w_i^D\right)\right\}, i = 1..n$

[3] $sim(P,D) = \dfrac{P \cdot D}{\|P\| \cdot \|D\|} = \dfrac{\sum_{i=1}^{k} w_i^P \cdot w_i^D}{\sqrt{\sum_{i=1}^{k} w^2{}_i^P}\sqrt{\sum_{i=1}^{k} w^2{}_i^D}}$

[4] $W_{i,D} = Tf_{i,D} \cdot \log(\dfrac{n}{DF_i})$

With:

– t^P_j: the term j in the user profile vector P.

– t^D_i: the term i in the document vector D.

– w^P_j: the weight of the term j in the user profile vector P.

– w^D_i : the weight of the term i in the document vector D.

– sim(P,D): the cosine measure of similarity between a user profile vector and a document vector.

–$Tf_{i,D}$: the number of occurrences of the term i in the document D.

– n: the number of documents in the collection.

–DF_i: the number of documents that contain the term i at least once

3.3. Obstacles and challenges of interactive product line configuration

The goal of PLE is to efficiently manage variability and commonalities of a particular domain and allow configuring new products. Products are configured from a particular PLM, or collection of models, that represent the "legal" combination of "artefacts" in a particular domain. Even if the automatic configuration of products from PLMs is a well known and handled problem thanks to the use of off-the-shelf solvers that execute PLMs and generate valid products, interactive configurations of new products from PLMs has been poorly treated in literature and constitutes itself an important challenge for industry. In this section we will discuss this challenge by means of several technical and economical issues. The first issue of industrial configurators of PLMs is performance [AST 10]. Indeed, as soon as a customer or a vendor makes a configuration decision (e.g. find the less expensive and more performing products) he wants to find out what the consequences of the decisions are. From a marketing perspective, it is unpleasant for the customer to wait for several seconds to know whether his requirements are correct or not in terms of configuration. This issue is extremely difficult to solve at the industrial level, because the computational complexity of computing PL configurations grows quadratically with the size of the PLM [AST 10]. The second issue is guidance. In practice, choosing from a wide range of options is quickly difficult for the customer who doesn't know where to start, or which alternative choose. Besides each time a customer makes a decision, this can be contradictory with previous decisions, or have a negative impact on downstream decisions. The customer will be disappointed to see that his choices are not correct, with the risk that he ultimately turns to a competitor. Therefore, it is crucial to guide the customer in a process that guaranty the existence of at least one satisfactory configuration at the end of the configuration process. Thus, a user without any guidance has one possibility among 1000 to achieve a correct configuration in the Renault van family "Traffic" represented in a PLM with 10^{21} possible configurations [DAU 10]. It is worth noting that guidance does not mean limitation, and it is because, the third issue is about configuration freedom. We are convinced that PLM configuration should be a user driven process, and in this sense users should be free to (i) choose the configuration order; (ii) make negative choices; (iii) do not make a choice; (iv) change their minds; and (v) query the PLM with criteria that are not necessarily considered in the model. Users should be free to choose the order that best fits their priorities and not only "restrictive order" imposed by the structure of the PLM. The challenge from this point of view is about how to increase freedom and scalability in the guidance of the configuration process, while maintaining a reasonable performance?

Industrial systems (such as Amazon) already offer guidance to choose in a very large collection of products using recommendation techniques. However, these recommendation techniques are not adapted to complex systems such as product lines, configurable software, or composite systems with a large number of options.

Indeed, although recommendation techniques can deal with many options to decide upon, they do not take into account the constraints between them. We are thus faced to a situation where configuration systems and recommendation systems solve two parts of our problem, but not in an integrated way.

4. Overview of the proposed approach

One way to deal with the problem consists in guiding the user by intertwining the recommendation and configuration activities in an iterative way.

In this context, the recommendation consists on a set of partial recommendations. Indeed, at each iteration of the recommendation-configuration process, the recommendation technique is applied for each partial configuration. We call this "partial recommendation". The recommendation technique used in our approach is the mix between the knowledge-based recommendation and the content-based recommendation. In fact, our goal is to recommend partial configurations that are valid with respect to the product line constraints and that meet user requirements already specified. The knowledge-based recommendation allows to configure and to recommend features product using the constraint-based recommendation. However, we want to reason on some features but not with the knowledge-based recommendation but rather with a content-based or collaborative recommendation. Therefore, our approach consists on the combination of the knowledge-based recommendation and the content-based recommendation.

An overview of the process is shown in Figure 2. As the figure shows it, the process consists in focusing on packs of features: recommendation is used to help customers make their choices, and configuration to check whether recommendations and customer choices are correct, and to propagate them to the rest of the features of the PL.

For example, the interactive configuration can be started with an "a priori approach" by focusing on a pack of features, which can for instance be pre-selected by marketing people, or be themselves identified using recommendation technique based on the customer profile. In PL engineering, this is called a "partial configuration". PL configurators are able to handle partial configurations by telling if they are correct (i.e., there is at least one product that satisfies the requirements), counting the number of correct products, show a list, indicate which features have become mandatory (resp. forbidden), etc. The first thing is to check that the partial configuration is correct. If so, and if the number of candidate products is still too high for manual selection, then recommendation can be used to help the customer make decisions for the next pack of features. The recommendation technique

compares the requirements already specified by the customer with those recorded in a database of correct configurations (e.g., former configurations produced in other contexts, or specified by marketing people). Requirements for the next pack of features are recommended under the form of a list of partial configurations shown from the most recommended to the least recommended. As these partial configurations combine actual requirements with possible ones deduced from other configurations, it may happen that some of them are not correct. Incorrect configurations are thus removed from the list before the list is displayed to the customer. The customer specifies requirements for the second pack of features by selecting from the list, then the process proceeds in the same way until the customer decides to stop, either by selecting a full configuration, or by asking the configurator decide on the full configuration.

Figure 2. *The combining configuration and recommendation process*

It is necessary to use a partial configuration strategy to avoid the issues raised by a priori configuration. Indeed, when the requirements are specified, there are chances that they are not consistent with the PL constraints. Focusing on packs of features rather than on all features at the same time helps reduce the risk of incorrect requirements, which results in cumbersome backtracking. In the partial configuration strategy, the user does not have to decide on all features. Only a subset of features are considered first, then decisions are automatically propagated through the constraints onto the other features. Not only this diminishes the risk of incorrect configurations but it also allows propagating the impact of requirements specified for some features onto the rest of the PL features. In customer terms, they have the possibility to foresee the impact of their decisions when they specify their requirements.

At the extreme it would be possible to focus on the product feature after feature. This may not be necessarily always interesting in some situations particularly when the PL specification contains many features, but it could be interesting when there are abstract features that are relevant and determinant.

Different recommendation techniques can be used in this process. The one, which adaptation we want to illustrate in this chapter is the content based filtering. Content based filtering uses the definition of existing products to support recommendation. This is consistent with the process described above in which products are defined as a combination of features that have already shown correct. The idea is of course to recommend only partial configurations that (a) are valid with respect to the PL constraints, and (b) satisfy the requirements that the customer has already specified. In order to do that, the recommendation algorithm shown in Figure 3 searches for the intersection of recommended partial configurations, correct ones, and partial configuration that satisfy customer requirements. This is achieved by first looking for recommendations, then removing those that do not comply with the customer's requirements, then let customer make a choice, then propagate constraints from the features concerned by the choices already made to the rest of the features of the PL.

E_0: Group features into packages
E_1: Partial Configuration on P_1
E_2:
    For P_i=P_2 to P_n do
        If P_i is "free" then
            $E_{2.1}$: Recommend
            $E_{2.2}$: Choose a partial configuration
            $E_{2.3}$: Propagate constraints
E_3: Return solution

Figure 3. *Recommendation algorithm embedding content based filtering*

Each iteration of the algorithm focuses on a pack of features that are defined in step 0. There is no need that step 0 coincides with the rest of the process: this decision can be made long before, once and for all customers or purchase situations. It is also possible to skip step 0 and calculate the pack of features dynamically based on different kinds of heuristics. One is of course recommendation itself as mentioned earlier. Another one consists in focusing on parts of the PL that show least variability. The rationale for this heuristic is that in a satisfaction problem such as PL configuration, you have more chance to find correct solutions when you focus on the least bounded variables than when you focus on variables that have more possible values. Another heuristic would consist in walking through the structure of the PL specification so as to focus on features that relate to each other. The rationale behind is that it is more interesting to decide together on product features that have to do with one another (e.g., characteristics of the car engine), than on features that are not or less directly related (e.g., horsepower and colour of the car). A more advanced way of working consists in exploiting dependencies between decisions and regroup features in packs that have the least dependencies, so that making decision on features in one pack will propagate as little as possible on features in other packs. This strategy requires however advanced reasoning techniques that are far beyond the scope of this chapter.

The results at each iteration, except the last one, is a partial configuration where some features are explicitly selected or drawn aside, and other features are implicitly selected or drawn aside through constraint propagation.

Indeed, for each remaining package, we test if there are still choices to do on selecting features. This is what means “free”. If the package Pi is free, then we recommend other configurations by reasoning on the subset of features of the package Pi. The user chooses one of the proposed solutions, which is a partial configuration on Pi. Next, we propagate constraints on the other packages of features. Finally, we provide the solution which is a complete and consistent configuration according the user requirements.

The following example illustrates the application of our approach to the Feature Model example presented in Fig.1.

We assume that packs of features are defined as follows:

P1: {F1,F2}; P2:{F3,F4,F5,F6}; P3:{F7,F8}; P4:{F9,F10}; P5:{F11,F12};

The following table presents the steps of the configuration combined with recommendation.

E0	P1:{F1,F2}	P2:{F3,F4,F5,F6}	P3:{F7,F8}	P4:{F9,F10}	P5:{F11,F12}
E1	**{F1,F2}**				
E2.1		{F3,F4,x,x} {F3,F4,F5,x} {F3,F4,x,F6} {F3,F4,F5,F6}			
E2.2		**{F3,F4,x,x}**			
E2.3 & E2.1		**{F3,F4,x,x}** →	{F7,x} → {F7,F8}→	{F9,x}→ {x,F10}→ {x,x}→	{F11,x} {x,F12} {F11,x} {x,F12} {F11,x} {x,F12}
E2.2			**{F7,x}**		
E2.3			**{F7,x}**	{F9,x}→	{F11,x} {x,F12}
E3	**{F1,F2}** **{F1,F2}**	**{F3,F4,x,x}** **{F3,F4,x,x}**	**{F7,x}** **{F7,x}**	**{F9,x}** **{F9,x}**	**{F11,x}** **{x,F12}**

Table 1.*Illustration of the application of the proposed approach*

In the example presented in Table 1, the algorithm returns two possible solutions for the user, {F1,F2,F3,F4,x,x,F7,x,F9,x,F11,x} and {F1,F2,F3,F4,x,x,F7,x,F9,x,x,F12}.

5. Preliminary evaluation

The proposed approach was implemented on a case study for a company R. The case study consists on combining the configuration and the recommendation for an "Electric Board" product line.

The modelling process has had several phases. At the beginning, the PL was modelled using the Feature Oriented Domain Analysis (FODA) formalism that is a domain analysis method that focuses on the features of the domain systems. The feature model (FM) in FODA serves as a communication medium between the user, domain experts, domain analysts, system analysts and developer. However, a multi-instantiation issue rises. Indeed, in the "Electric Board" PL, there are many features that can be instantiated multiple times for a single configuration, but the FM does not allow the multi-instantiation. Thus, to cope with this problem, we used the TVL (Textual Variability Language) [CLA 11], one of the latest incarnations of FMs, due to its support for the multi-instantiation and other advantages such as the high expressiveness, the formal semantics and the tool support [CLA 11].

The combination of the content-based recommendation and the knowledge-based recommendation, in this case study, shows that several questions are raised such as the question on the order of features. In this case study, the order of product features was predefined, but the user may want to have the choice of choosing a personalized sequence of product features. Thus, it would be interesting to support the user in making decision on the order of features by recommending the most relevant features at first. Secondly, we have noticed that users could be interested on what other users have already purchased and not only on given features, for example, the user could not be interested on the feature "price" of elements of Electric Board but on what Electric Board elements was purchased by the other users.

6. Discussion and Related work

6.1. Recommendation techniques

There are three recommendation approaches that are comprised of a set of knowledge sources and are algorithmic approaches to producing recommendations using those sources [FEL 08]:

– Content-based recommendation

– Collaborative recommendation

– Knowledge-based recommendation

The content-based recommendation [PAZ 97] is like a "classification task in the machine learning sense" [FEL 08]. It is based on items descriptions and a user profile. It analyzes items descriptions to recommend items that are similar to those that a user liked in the past. The user profile is based on a weighted vector of item features to denote the importance of each feature for the user. There are many classification learning algorithms used to predict the user degree of interest for an item such as Bayesian classifiers, decision trees and K-nearest neighbours [PAZ 07]. The content-based technique does not suffer from the new item problem but the new user problem remains because a user profile must be built from multiple user ratings [FEL 08]. The second issue is that this kind of recommendation technique is limited to recommend content of the same type as the user is using. Another issue can be raised in the context of product line recommendation is that the content-based approach is not adapted for complex products combining different features with respect for PL constraints. In product line context, the goal is not to recommend a product but to recommend product features combination in a consistent way by respecting the PL constraints and satisfying user needs.

The collaborative recommendation [LIN 03] uses two types of algorithms for calculating prediction: memory based collaborative filtering algorithms and model based filtering algorithms [BRE 98].Memory-based algorithms uses the user-item matrix to make predictions. These systems use the user neighbourhoods to make collections of similar users. Then, these systems gather neighbours preferences to compute a prediction for the user [SAR01]. Model-based algorithms develop a model of user ratings to provide recommendation. These systems compute the expected value of a user prediction, on the basis of his ratings on other items [SAR01]. Models are developed using different machine learning algorithms such as Bayesian network and clustering approaches [BAS 98], [BRE 98]. However, this approach presents issues because it depends on human ratings. Then, when data are sparse, the performance of this technique decreases. This makes this approach unable of scaling up with an important amount of data. In the product line context, the same issue, described above with the content-based approach, can be raised with collaborative approach.

The knowledge-based recommendation relies on knowledge about users and products to generate recommendations of products that meet user requirements [MOO 00]. It is characterized by its emphasis on user preferences and how to recommend products satisfying his/her needs [LIN 03]. The constraint-based recommendation is one of the approaches to knowledge-based recommendation [FEL 08]. It is based on a set of recommendation rules that describes the possible combinations of product features to provide recommendations from choices that have been made by the user ensuring the consistency. Constraint-based recommenders support users by explaining items and proposing repair actions [FEL 07] which indicates interesting and minimal changes to the user requirement such that the calculation of a recommendation becomes possible. Furthermore, they help the user by identifying and proposing interesting attribute settings based on the preferences of user community [FEL 08]. The prediction of attribute settings is used when users are not able to specify all relevant requirements because of lack of detailed technical knowledge. In the PL context, the approach presented in this paper describes the configuration process like the knowledge-based recommendation. Indeed, the configuration-recommendation process uses the constraint-based recommendation. Felfernig et al. [FEL 08] make the recommendation on a product. Indeed, given P1, P2 and P3 three products that the user has already selected. Their approach consists on recommending a product P4 that is compatible with decisions already made on products P1, P2 and P3. However, the approach proposed in this paper does not focus a product but on features product. In fact, given the product features f1, f2 and f3 that has been selected by the user, the rationale is to recommend a product feature f4. Secondly, in the proposed approach, the knowledge-based recommendation is specified by features and constraints between them. It is also possible to handle with partial configurations unlike Felfernig et al.[FEL 08] who recommend products.

There is another approach to knowledge-based recommendation which is heuristics – based recommendation. In a recent work, Mazo et al [MAZ 14] coin the term heuristics-based recommendation to improve PLM configuration processes. According to authors, heuristics-based recommendation "can be classified as a knowledge-based technique as it is based on a set of knowledge sources that were implemented as heuristics to recommend what element(s) to configure during PLM configuration processes". In that work, Mazo et al present six recommendation heuristics to improve the interactivity of product line configuration so as to make it scalable to common engineering situations. In particular, these heuristics are intended to help customers specify the characteristics of their products step-by-step according to their requirements, and to avoid useless or inefficient decisions. Mazo et al [MAZ 14] describe the principles, benefits and the implementation of each heuristic using constraint programming. In addition, they demonstrate the application and usability of the heuristics by means of a real-world case study from the car

industry. This approach does not recommend a product but it recommends features product using heuristics and not using similarity on past configurations.

7. Conclusion and Future work

In this paper, we have presented a partial recommendation approach combined with the product line configuration. The proposed approach consists on combination between the knowledge-based recommendation and the content-based recommendation. In other words, it is a combination between the dynamic recommendation (constraint-based recommendation) and the similarity measure (content-based recommendation).

The approach was implemented for a real case study in a company R. The case study consists on an "Electric Board" product line. We have presented a preliminary evaluation for this case study and we discussed our positioning relative to other works in the literature.

As a future work, we can use other kinds of combination between the recommendation and the configuration of product lines. For example, we can model the PL by different models simultaneously. Then, we apply different recommendation techniques for each model. Next, we combine the recommendations to have final recommendation.

Bibliography

[ALF 12] ALFÉREZ G.H., PELECHANO V. *Dynamic Evolution of Context-Aware Systems with Models at Runtime*. To appear in Proceedings of the ACM/IEEE 15th International Conference on Model Driven Engineering Languages & Systems, Innsbruck, Austria, 2012.

[AST 10] ASTESANA J-M., COSSERAT L, FARGIER H. *Constraint-based vehicle configuration: a case study*. In International Conference on Tools with Artificial Intelligence (ICTAI), IEEE Computer Society. Arras, 2010.

[BAL 97] BALABANOVIC M., SHOHAM Y. *Combining Content-based and collaborative recommendation*. Communications of the ACM, 40 (3), 1997.

[BAS 98] BASU C., HIRSH H., COHEN W. *Recommendation as classification: Using social and content-based information in recommendation*. In Proceedings of the Fifteenth National Conference on Artificial Intelligence, pp. 714-720, 1998.

[BEN O5] BENAVIDES D., RUIZ-CORTÉS A., AND TRINIDAD P. *Using constraint programming to reason on feature models*. In The Seventeenth International Conference on Software Engineering and Knowledge Engineering, pages 677–682, 2005.

[BEN 10] BENAVIDES D., SEGURA S., RUIZ-CORTÉS A. *Automated analysis of feature models 20 years later: A literature review*. Information Systems, pp. 615-636, 2010 (6e édition).

[CLA 11] CLASSEN A., BOUCHER Q., HEYMANS P. *A text-based approach to feature modelling. Syntax and semantics of TVL," SCP, vol. 76, pp.1130–1143, December 2011.*

[COE 02] COESTER R., GUSTAVSSON A., OLSSON R., RUDSTROEM A. *Enhancing web-based configuration with recommendations and cluster-based help*. In *AH'02 Workshop On Recommendation and Personalization in Ecommerce*, pp.1-10, 2002.

[CZA 05] CZARNECKI K., HELSEN S., EISENECKER U.W. *Formalizing cardinality-based feature models and their specialization*. Software Process: Improvement and Practice, pp. 7–29, 2005 (1e edition).

[DAU 10] DAURON A., ASTESANA J-M. *Spécification et configuration de la ligne de produits véhicule de Renault.* Journée Lignes de Produits. Université Panthéon Sorbonne, 2010.

[DJE 11] DJEBBI O.*L'ingénierie des exigences par et pour les lignes de produits*. Université Paris1 Panthéon-Sorbonne, France, PhD Thesis, 2011.

[FAL 11] FALKNER A., FELFERNIG A., HAAG A. *Recommendation Technologies for Configurable Products*. AI Magazine pp. 99-108, 2011 (3eédition).

[FEL 07] FELFERNIG A., ISAK K., SZABO K., ZACHAR P. *The VITA Financial Services Sales Support Environment.*AAAI/IAAI, pp. 1692-1699, 2007.

[FEL 08] FELFERNIG A., BURKE R.*Constraint-based Recommender Systems: Technologies and Research Issues*. Proceedings of the ACM International Conference on Electronic Commerce (ICEC'08*)*, Innsbruck, Austria, pp. 17-26, 2008.

[FEL 09] FELFERNIG A., SCHUBERT M., FRIEDDRICH G., MANDL M., MAIRITSCH M., TEPPAN E.*Plausible Repairs for Inconsistent Requirements*.21st International Joint Conference on Artificial Intelingence, pp. 791-796, 2009.

[HAD 04a] Hadzic T., Andersen H. R.*An introduction to solving interactive configuration problems*.Technical Report TR-2004-49, The IT University of Copenhagen, 2004.

[HAD 04b] HADZIC T., SUBBARAYAN S., JENSEN R. M., ANDERSEN H. R., MOLLER J., HULGAARD H. *Fast backtrack-free product configuration using a precompiled solution space representation*. In International Conference on Economic, Technical and Organizational aspects of Product Configuration Systems (PETO), pp. 131-138, 2004.

[KAN 90] KANG K., COHEN S., HESS J., NOVAK W., PETERSON S. *Feature–Oriented Domain Analysis (FODA) Feasibility Study*. Technical Report CMU/SEI-90-TR-21, Software Engineering Institute, Carnegie Mellon University, 1990.

[JAN 08] JANOTA M.*Do SAT solvers make good configurators?* In First Workshop on Analyses of Software Product Lines, 2008.

[LIN 03] LINDEN G., SMITH B., YORK J.*Amazon.com recommendations: item-to-item collaborative filtering*. IEEE Internet Computing, pp. 76-80, 2003 (1e edition).

[MAZ 11a] MAZO R., GRÜNBACHER P., HEIDER W., RABISER R., SALINESI C., DIAZ D. *Using constraint programming to verify DOPLER variability models*. VaMoS 2011, pp. 97-103, 2011.

[MAZ 11a] MAZO R., LOPEZ-HERREJON R. E., SALINESI C., DIAZ D., EGYED A. *Conformance Checking with Constraint Logic Programming: The Case of Feature Models*. COMPSAC 2011: 456-465

[MAZ 11b] MAZO R., SALINESI C., DIAZ D., LORA-MICHIELS A. *Transforming Attribute and Clone-enabled Feature Models into Constraint Programs over Finite Domains*. ENASE 2011, pp. 188-199, 2011.

[MAZ 11c] MAZO R., SALINESI C., DIAZ D. *Abstract Constraints: A General Framework for Solver-Independent Reasoning on Product Line Models*. INSIGHT - Journal of International Council on Systems Engineering (INCOSE), Volume 14 Issue 4, pp. 22-24, 2011. http://www.panetto.fr/INSIGHT_vol-14-issue-4.pdf

[MAZ 12] MAZO R., SALINESI C, DIAZ D., DJEBBI O., LORA-MICHIELS A. *Constraints: the Heart of Domain and Application Engineering in the Product Lines Engineering Strategy*. International Journal of Information System Modeling and Design IJISMD, ISSN 1947-8186, eISSN 1947-819.Vol. 3, No. 2, pp. 33-68, 2012.

[MAZ 14] MAZO R., DUMITRESCU C., SALINESI C., DIAZ D**.** *Recommendation Heuristics for Improving Product Line Configuration Processes***.** In: Recommendation Systems in Software Engineering. Robillard M, Maalej W., Walker R. and Zimmermann T. (Eds.), ISBN 978-3-642-45135-5, DOI 10.1007/978-3-642-45135-5__19, Springer-Verlag Berlin Heidelberg, 2014.

[MEN 09] MENDONÇA, M., BRANCO, M., COWAN, D.*S.P.L.O.T.: software product lines online tools.*In OOPSLA Companion. ACM, 2009,http://www.splot-research.org

[MEN 09] MENDONÇA, M., WASOWSKI, A., CZARNECKI, K.*SAT–based analysis of feature models is easy*. In Proceedings of the SofwareProduct Line Conference, 2009.

[MIR 07] MIRZADEH N., RICCI F. *Cooperative Query Rewriting for Decision Making Support*. Journal of Applied Artificial Intel-ligence, 21(10):895—932, 2007.

[MOO 00] MOONEY R. J., ROY L. *Content-based book recommending using learning for text categorization*. In proceedings of DL-00, 5th ACM Conference on Digital Libraries, pp. 195-204, ACM Press, San Antonio, US, 2000.

[PAZ 97] PAZZANI M., BILLSUS D. *Learning and Revising User Profiles: The Identification of interesting Web Sites*. Machine Learning, Vol.27, p.313-331, 1997.

[PAZ 99] PAZZANI M.*A Framework for Collaborative, Content-Based, and Demographic Filtering*. Artificial Intelligence Rev., pp. 393-408,1999.

[PAZ 07] PAZZANI M., BILLSUS D. *Content-Based Recommendation Systems*. In: The Adaptive Web: Methods and Strategies of Web Personalization, Lecture Notes in Computer Science, Vol. 4321. Springer-Verlag, Berlin Heidelberg New York, 2007.

[POH 05] POHL K., BÖCKLE G., VAN DER LINDEN F. *Software Product Line Engineering – Foundations, Principles, and Techniques*. Springer, Berlin, Heidelberg, New York, 2005.

[REI 87] Reiter R. *A theory of diagnosis from first principles*. Artificial Intelligence, 23(1): 57-95, 1987.

[RIE 02] RIEBISCH M., BÖLLERT K., STREITFERDT D., PHILIPPOW I. *Extending Feature Diagrams with UML Multiplicities*. In Proceedings of the Sixth Conference on Integrated Design and Process Technology (IDPT 2002), Pasadena, CA, 2002.

[SAL 10] SALINESI C., MAZO R., DIAZ D., AND DJEBBI O. *Solving Integer Constraint in Reuse Based Requirements Engineering*. International Conference on Requirement Engineering (RE), Sydney, Australia, 2010.

[SAL 89] SALTON G. *Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer*. Addison-Wesley, Reading, MA, 1989.

[SAW 12] SAWYER P., MAZO R., DIAZ D., SALINESI C., HUGHES D. *Constraint Programming as a Means to Manage Configurations in Self-Adaptive Systems*. Accepted to be published in the Special Issue in IEEE Computer Dynamic Software Product Lines, pp. 1-12. 2012.

[STR 03] STREITFERDT D., RIEBISCH M., PHILIPPOW I. *Details of formalized relations in feature models using ocl*. In ECBS, pages 297–304. IEEE Computer Society, 2003.

[SUB 04] SUBBARAYAN S., JENSEN R. M., HADZIC T., ANDERSEN H. R., HULGAARD H., MOLLER J. *Comparing two implementations of a complete and backtrack-free interactive configurator*. In Workshop on CSP Techniques with Immediate Application (CSPIA), pp. 97-111, 2004.

[SUB 05] SUBBARAYAN S. *Integrating CSP decomposition techniques and BDDs for compiling configuration problems*. In International Conference on Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems (CPAIOR), volume 3524 of LNCS, pages 351-365. Springer, 2005.

[TII 10] TIIHONEN J., FELFERNIG A. *Towards Recommending Configurable Offerings*. International Journal of Mass Customization, 3(4):389-406, 2010

[VAN 02] VAN DEURSEN A., KLINT P. *Domain-Specific Language Design Requires Feature Descriptions*. Journal of Computing and Information Technology, 10(1):1-17, 2002.

[VAN 00] VAN METEREN R., VAN SOMEREN M. *Using content-based filtering for recommendation*. ECML 2000 workshop, 2000.

[WHI 09] WHITE J., DOUGHERTY B., SCHMIDT D.C. *Selecting highly optimal architectural feature sets with filtered Cartesian flattening*. Journal of Systems and Software, 82(8), pp. 1268–1284, 2009.