

Article

## Towards cardinality-based service feature diagrams

Ghulam Mustafa Assad<sup>1</sup>, Muhammad Naeem<sup>1</sup>, Hafiz Abdul Wahab<sup>2</sup>

<sup>1</sup>Department of Information Technology, Hazara University, Mansehra, Pakistan

<sup>2</sup>Department of Mathematics, Hazara University, Mansehra, Pakistan

E-mail: gm\_assad@yahoo.com, naeem@hu.edu.pk, wahabmaths@yahoo.com

Received 16 April 2014; Accepted 20 May 2014; Published online 1 March 2015



### Abstract

To provide efficient services to end-user it is essential to manage variability among services. Feature modelling is an important approach to manage variability and commonalities of a system in product line. Feature models are composed of feature diagrams. Service feature diagrams (an extended form of feature diagrams) changed the basic framework of feature diagrams by proposing new feature types and their relevance. Service feature diagrams provide selection rights for variable features. In this paper we argue that it is essential to put cardinalities on service feature diagrams. That is, the selection of features should be done under some constraints, to provide a lower and upper limit for the selection of features. The use of cardinalities on service feature diagrams reduces the types of features to half, while keeping the integrity of all features.

**Keywords** feature modeling; service feature diagrams; software product line; variability and similarity management; cardinality-based service feature modelling.

Computational Ecology and Software  
ISSN 2220-721X  
URL: <http://www.iaees.org/publications/journals/ces/online-version.asp>  
RSS: <http://www.iaees.org/publications/journals/ces/rss.xml>  
E-mail: [ces@iaees.org](mailto:ces@iaees.org)  
Editor-in-Chief: Wenjun Zhang  
Publisher: International Academy of Ecology and Environmental Sciences

### 1 Introduction

Software product line engineering is one of the ways used by the researchers in industry to automate product development of the product line. One of the challenges for product development is the use of variability and commonality among the features of products. Feature modelling is established notation to deal with such type of challenges (Batory et al., 2006). Feature diagrams were introduced as a part of the Feature-Oriented Domain Analysis (FODA) in (Kang et al., 1990). Feature diagrams are used in number of domains including telecom systems (Griss et al., 1998), template libraries (Czarnecki and Eisenecker, 2000), network protocols (Barbeau and Bordeleau, 2002), and embedded systems (Czarnecki et al., 2002).

Feature models are, hierarchical models to record commonalities and variabilities among the products of a product line. In the model, each characteristic relevant to the problem space is said to be a feature. So in this sense, a feature is called a characteristic of a system. We can say that a feature can be a requirement, a quality, a technical function or a non-functional characteristic (Czarnecki and Kim, 2005). For example, colour, tires,

and doors are features of a product line of a car.

The original feature diagrams have many extensions proposed by different authors. For example, the first extension of FODA diagrams is Feature-RSEB, proposed in (Griss et al., 1998). The second extension of FODA diagrams is Cardinality-based feature diagrams, proposed in (Czarnecki, 2005). Another extension of FODA diagrams is *service feature diagrams (SFD)*, proposed by Naeem in (Naeem and Heckel, 2011).

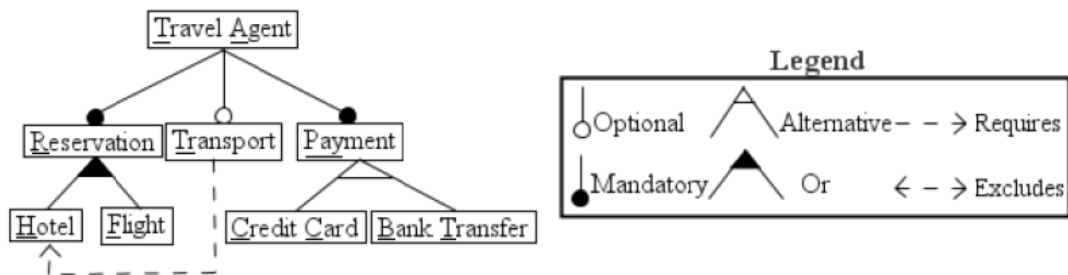
This paper is an extension of our previous work in (Naeem and Heckel, 2011; Naeem, 2012) and first step towards developing the framework for cardinality-based service feature diagrams (CSFD). In this paper, we argue to use cardinalities in service feature diagrams; we believe that the full benefit of service feature diagrams can be obtained by using cardinalities. CSFD, not only, reduces the feature types of SFD, but subsumes all the features of SFD, as well.

The rest of the paper is arranged as follows: Section 2 elaborates on the information which is necessary to understand the technical contents of the paper, while the Section 3 consists of the proposed scheme of CSFD. Section 4 concludes the paper.

## 2 Background and Related Approaches

### 2.1 Classical feature diagrams

Usually, feature diagrams represent hierarchies of common and variable features in software product lines (Kang et al., 1990). Here, we use them to describe variability of services building on the notation provided in (Czarnecki and Eisenecker, 2000). For example, Fig. 1 shows a feature diagram D for an online travel agent.



**Fig. 1** Feature diagram of an on-line travel agent service.

An instance of D is a subset of features that is consistent with the constraints specified by D. For example, a valid purchase from the travel agent shown in Fig. 1 must obey the following rules:

1. The root feature is always selected
2. If a feature is selected, its parent must also be selected
3. If a feature is selected, all the mandatory features of its And-group are selected
4. If a feature is selected, exactly one feature of its Alternative-group must be selected
5. If a feature is selected, at least one feature of its Or-group must be selected
6. When two features are linked by requires constraint, the target feature must be included whenever the source is
7. A source and target of excludes constraint (not shown in Fig. 1) cannot be selected in an instance.

Thus, an instance of D is given by {TA, Res, H, Pay, BT} (For brevity, we use the underlined characters of the feature name in instances and logical formulas.). A declarative way of defining the notion of instance is by means of Propositional Logic (Czarnecki and Wasowski, 2007). For example, a propositional equivalent of the feature diagram in Fig. 1 is:

$$(TA \leftrightarrow Res) \wedge (Res \leftrightarrow (H \vee F)) \wedge (Tr \leftrightarrow TA) \wedge (Tr \rightarrow H) \wedge (TA \leftrightarrow Pay) \wedge (CC \leftrightarrow (\sim BT \wedge Pay)) \wedge (BT \leftrightarrow (\sim CC \wedge Pay))$$

Valuation for which this formula is true characterise the valid instances. In our example, a possible instance is the valuation that assigns true to {TA, Res, H, Pay, BT} and false to {F, Tr, CC}.

## 2.2 Cardinality-based feature diagrams

Cardinality-based feature diagrams uses multiplicities on features. Cardinality-based feature modelling is an integration and extension of existing approaches. Czarnecki in (Czarnecki and Kim, 2005) say, “A cardinality-based feature model is a hierarchy of features where each feature has feature cardinality” i.e., cardinality-based feature diagrams put constraints on features, provides a lower and upper limit for the selection of features.

Feature cardinality denotes the number of clones of sub-features which can be selected for a parent feature. Cardinalities are shown as [m...n], where m and n denote minimum and maximum number of selection for a feature, respectively. Feature with cardinality [1...1] are called mandatory, whereas features with cardinality [0...1] are called optional. Group cardinality is an interval of the form [m-n], where  $n \in \mathbb{Z} \wedge 0 \leq m \leq n \leq k$ , where k is the number of features in the Group (Czarnecki and Kim, 2005).

Fig. 3 depicts a feature diagram showing seating capacity of a car manufacturer using cardinality-based feature diagrams. A possible instance of this feature diagram is {SC, F, LB, PS, R, H}.

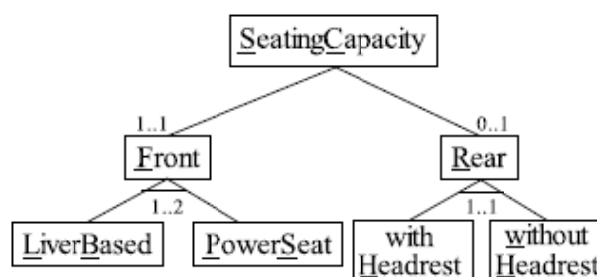


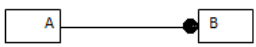
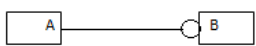
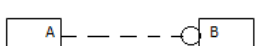
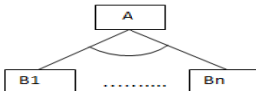
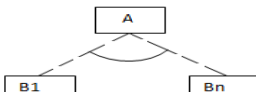
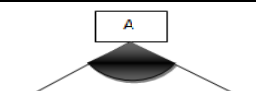
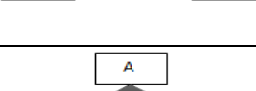
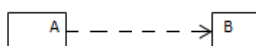
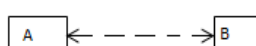
Fig. 3 Cardinality-based feature diagram showing seats of a car.

## 2.3 Service feature diagrams

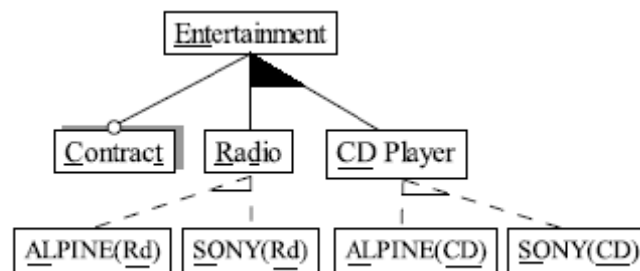
Service feature diagrams introduced some new type of notations to the classical feature diagrams in the context of service specification and matching. These new notations include:

1. *A solid edge*: This edge is used when the selection of features is given to the requestor.
2. *A dashed edge*: A dashed edge is used when the selection rights of features are left with provider to choose from.
3. *A resource feature*: This feature can only be used once; whereas the classical representation is used for resource feature.
4. *A shareable feature*: This feature can be used multiple times. A rectangular box with gray background is used to represent shareable features.

**Table 1** Notations used in service feature diagrams.

Features	Feature Representation	Comments
Mandatory		Feature B must be selected if A is, in an instance
Optional		Feature B may be selected or rejected with A in an instance depending on requester's choice.
		Feature B may be selected or rejected with A in an instance depending on provider's choice.
Alternative-group		Exactly one feature from the group of $B_1, \dots, B_n$ must be selected with A in an instance based on the requestor's preference.
		Exactly one feature from the group of $B_1, \dots, B_n$ must be selected with A in an instance based on the provider's preference.
Or-group		At least one feature from the group of $B_1, \dots, B_n$ must be selected with A in an instance based on the requestor's preference.
		At least one feature from the group of $B_1, \dots, B_n$ must be selected with A in an instance based on the provider's preference.
Implies		Target feature B must be selected if the source feature A is.
Exclude		Feature A and B cannot be selected in one instance.

Using the notations discussed above a service feature diagram for an entertainment system of a car manufacturer is shown in Fig. 2 below.

**Fig. 2** SFD showing entertainment system of a car manufacturer.

### 3 Cardinality-based Service Feature Diagrams

#### 3.1 Motivation

Although, SFDs have many advantages over classical feature diagrams, but detailed study of SFDs raises some questions. Let us consider an example of a graduate program at a university, where the university offers three different specializations in graduate program, i.e., software engineering (SE), data mining (DM), and networks (NW). Each research student at this university has to choose maximum of eight subjects to complete their degree. Five of them should be chosen from the set of core, while three from any of the set of electives. Furthermore, a student is not allowed to select subjects from multiple elective groups. University has the right to offer any of core or electives to the students, whereas students are given the right to choose their field of specialization. It is also assumed that the students do not choose more than eight subjects.

For example, university may offer generative development, semantic database, and network administration to the graduate students in software engineering, data mining and networking, respectively whereas the students are given to choose from three available specializations. The service feature diagram of the graduate program is shown in the figure below.

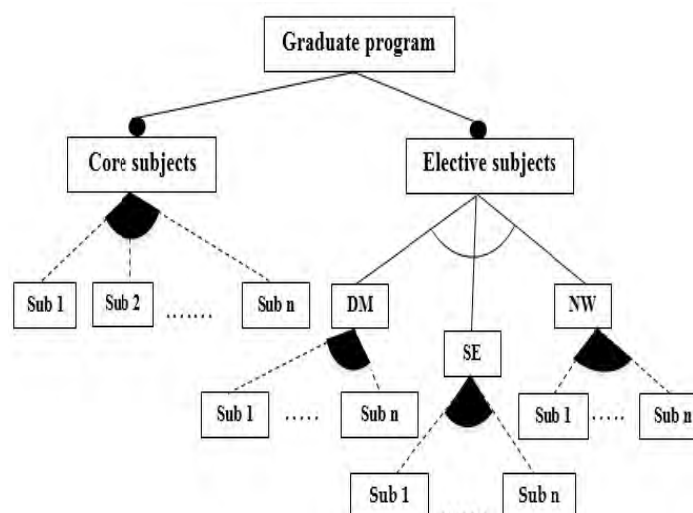


Fig. 4 Service feature diagrams for graduate program.

Fig. 4 captures the overall scenario of discussed graduate program, but it does not meet the complete requirements. As it does not give any information about number of subjects a graduate student is bound to choose to complete the degree. The Or-group of *Core Subjects* and Or-group of sub-categories of *Elective Subjects* do not restrict to select the required number of subjects. According to Fig. 4, one can assume that a student should be awarded by graduate degree even if they read less number of subjects than required, or exceeds the required number of subjects.

The main reason behind the discussed problem is the absence of maximum limit of selecting from the group of features. The use of cardinalities in service feature diagrams could do the trick for us. In our proposed scheme, we are using cardinalities on service feature diagrams, which we call cardinality-based service feature diagram. CSFD model is a hierarchy of features, where each feature has feature cardinality. A feature cardinality is an interval of the form  $[m..n]$ , where  $m$  and  $n$  both are real number. A feature with cardinality  $[1..1]$  referred as mandatory, whereas feature with cardinality  $[0..1]$  referred as optional. A group cardinality is an interval of the form  $[m..n]$ , where  $m$  and  $n$  both are real numbers and  $0 \leq m \leq n \leq k$ , where  $k$  is the number

of features in the group. Group cardinality denotes how many group members can be selected.

### 3.2 Proposed framework

CSFD offers two types of relationship of feature with its sub features: 1) Single feature; 2) Group feature. Single feature is either mandatory or an optional feature, while the Group feature is the combination of multiple features. Notation of CSFD is shown in Table 2. The first column under “Features” shows the types of features, whereas the diagrammatic representation of the corresponding feature type is shown in the second column under “Graphical Representation”.

**Table 2** Notations used in cardinality-based service feature diagrams.

Features	Graphical Representation	Comments
Single Feature		If feature B is mandatory sub-feature then it must be selected on selection of A, otherwise it may be selected or rejected based on the requestor's preference in an instance.
		A feature B may be selected depending on provider's preference in an instance, if A is selected.
Group Feature		If the feature A is selected then features $B_m$ to $B_n$ must be selected from this group in an instance, where $0 \leq m \leq n \leq k$ . This selection of features should be decided on the basis of requester's preferences.
		If the feature A is selected then features $B_m$ to $B_n$ must be selected from this group in an instance, where $0 \leq m \leq n \leq k$ . This selection of features should be decided on the basis of provider's preferences.

Fig. 4 is revised by using the notations of CSFD and shown in the Fig. 5 below. Please note that requestor in this case is the student and the provider is university. Now, Fig. 5 captures completely the example discussed in Section 3. Now no student could be considered to be Graduate except passing five subjects from Core Subjects and 3 from set of any specializations available at campus.

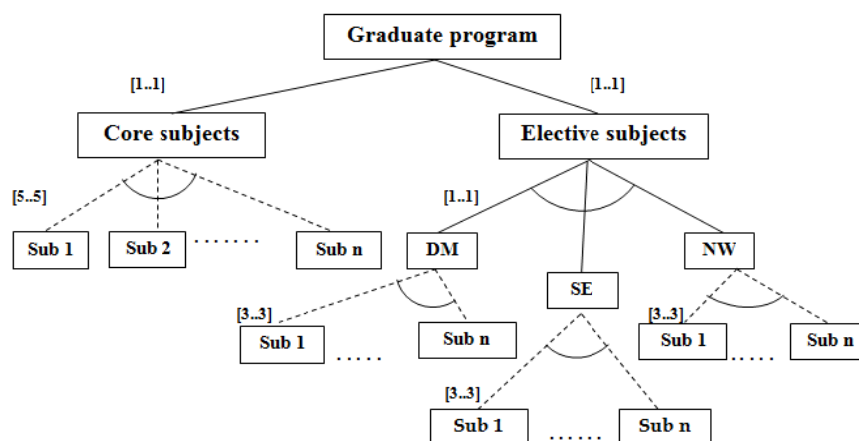


Fig. 5 Revised version of Figure 4 using notations of cardinality-based service feature diagram.

Furthermore, the selection of specialization is left with the students, whereas the selection of core and elective subjects is left with the university.

#### 4 Conclusions and Future Work

The concept of service feature diagrams was proposed in (Naeem and Heckel, 2011; Naeem, 2012). This paper is first step towards the development of a framework for the cardinality-based service feature diagrams. In this paper, we have argued the need of constraints on service feature diagrams with the help of a real example of a graduate program at a university. Apart from providing the full support to SFD, CSFD provides the following benefits:

- 1 Reduced number of features
- 2 Developers of feature models can easily focus on the cardinalities rather than filled and unfilled arcs or circles,
- 3 Arbitrary number of features can be chosen from the group feature. This number can range from zero to maximum number of features available in the group.

Future work will mainly focus on formalizing the concept of CSFD using Linear Logic (Troelstra, 1992; Girard, 1987, 1995) and developing a tool support for our proposed approach.

#### References

- Barbeau M, Bordeleau F. 2002. A protocol stack development tool using generative programming. In : Proceedings of the ACM Conference on Generative Programming and Component Engineering (GPCE'02), Pittsburgh, Vol. 2487 of LNCS. 93-109, Springer-Verlag, Heidelberg, Germany
- Batory D, Benavides D, Ruiz-Cortes A. 2006. Automated analysis of feature models: challenges ahead. Communications of the ACM, 49(12): 45-47
- Czarnecki K, Bednasch T, Unger P, Eisenecker UW. 2002. Generative programming for embedded software: An industrial experience report. In: Proceedings of the ACM Conference on Generative Programming and Component Engineering (GPCE'02), Pittsburgh, Vol. 2487 of LNCS. 156-172, Springer-Verlag, Heidelberg, Germany
- Czarnecki K, Eisenecker UW. 2000. Generative Programming: Methods, Tools, and Applications, Addison-Wesley, Boston, MA, USA
- Czarnecki K, Helsen S, Eisenecker U. 2005. Formalizing cardinality-based feature models and their specialization. Software Process Improvement and Practice, 10(1): 7-29

- Czarnecki K, Kim CHP. 2005. Cardinality-Based Feature Modeling and Constraints: A Progress Report. OOPSLA'05 Workshop on Software Factories, San Diego, California, USA
- Czarnecki K, Wasowski A. 2007. Feature diagrams and logics: there and back again. Proceedings of International Conference on Software Product Lines (SPLC'07). 23-34, IEEE Computer Society, Washington DC, USA
- Girard JY. 1987. Linear logic. Theoretical Computer Science, 50: 1-102
- Girard JY. 1995. Linear Logic: Its Syntax and Semantics. In: Proceedings of the Workshop on Advances in Linear Logic (ALL'95). 1-42, New York, USA
- Griss M, Favaro J, d'Alessandro M. 1998. Integrating feature modeling with the RSEB. In: Proceedings of the Fifth International Conference on Software Reuse (ICSR). 76-85, IEEE Computer Society Press, Los Alamitos, CA, USA
- Kang K, Cohen S, Hess J, Novak W, Peterson S. 1990. Feature-Oriented Domain Analysis (FODA) Feasibility Study. Technical Report, Carnegie-Mellon University, USA
- Naeem M, Heckel R. 2011. Towards matching of service feature diagrams based on linear logic. Proceedings of Workshops of SPLC 2011. 13, Munich, Germany
- Naeem M. 2012. Matching of Service Feature Diagrams using Linear Logic. PhD Dissertation. Department of Computer Sciences, University of Leicester, UK
- Troelstra AS. 1992. Lectures on Linear Logic. Center for the Study of Language and Information, Stanford, CA, USA