IJISET - International Journal of Innovative Science, Engineering & Technology, Vol. 3 Issue 5, May 2016.
www.ijiset.com
ISSN 2348 – 7968

An Analysis on Various Approaches to Model Variability

Ekta Yadav
M.tech scholar: Computer Science and Engineering
NorthCap University
Gurgaon, India
ektay18@gmail.com

Ms. Hitesh Yadav
Department of Computer Science and Engineering
NorthCap University
Gurgaon, India
hiteshyadav@ncuindia.edu

Abstract

In this era, where technology is changing rapidly there is strong need for software products that can adapt change. Variability is mainly required in software product line and it also imposes some challenges on software development. In variability, a software is customized according to a user's requirement. Software variability is aptness of a software product in which its components can be used for different purposes [1]. In this paper, Author analyze various approaches used to model variability in software product line with the help of a case study (library management system) LMS. Basic idea of the research is to find variability which will be pointed out by variant point.

1. INTRODUCTION

Technology is changing rapidly as per customers requirement. If a consumer wants little variation of a software product then it will be very time consuming for a developer to make it from scratch and that's resource consuming too. So software variability says that to make a product reusable, some points should be defined called variant points [3,11]. Variant point is the location at which variability is defined. Variant points extends the reuse of a software by giving reuser advancement of customize their product. Software reusability is the technique in which a software product is made to be used for different requirements for a specific customer or group of customers. Software reusability is a crucial step to make products dynamic. Section 2 of the paper describes variability. Section 3 of the paper describes different approaches to model variability using UML notations. Section 4 of the paper describes different issues in handling variability.

2. VARIABILITY

Variability deals with variation. Variability is a mechanism in which a software product can be used in different manners. A software product consists of two parts – core assets and application [2]. Core asset is the main core of the software product while application is the functionality related to it. Variability is identified by variant point. A variant point is the different point in the product where variability exists. There are two methods in which variant point is used. First one, in which design of variant point is understand by developer only. Second one in which reuser can make variant points according to its requirements.

3. MODELLING VARIABILITY USING UML NOTATIONS

There are various methods in which variability can be modeled [2]. First, value assignment and second is information hiding. This paper uses examples from library management system (LMS) to describe approaches for the identification of variant points. LMS is altered to confine several variation points to create a product line that can be used on various classes. Author is analyzing different classes (BookPrice, User, payment details, catalog) and their attributes to identify variation points.

3.1 Parameterization

In parameterization [2,7], variability is defined at the value of the parameter. Parameterization is used in domain analysis method. FODA (Feature Oriented Domain Analysis) [2] method helps in acquainting feature modelling to domain engineering. In this technique, different values can be passed to parameter by a reuser which has the authority over variant points, this makes system more dynamic. A reuser can pass multiple values. It can be achieved through component interface in which user is able to either initialize or change value of attributes.

In LMS, Fig.1 shows a class BookPrice has a method setBookPrice(in BookPriceParameter) which is used to show different values of different books with respect to different buyers(faculty and students). So, this is a variant point in this class.

ISSN 2348 – 7968

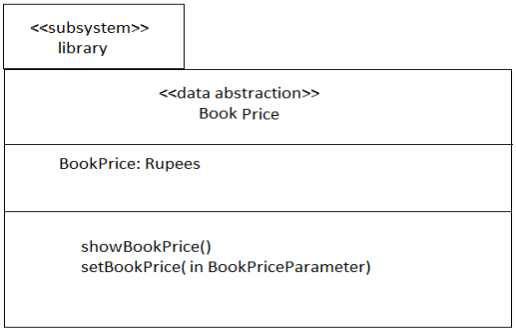

Fig.1. BookPrice - Parameterization

3.2 Inheritance

In this approach, two versions of a class does not need to use same interface [2]. Super class has an interface which is extended by the sub class, it can use already existing functions or use new operations [6]. Fig.2 shows an example of modeling variability using inheritance. It shows a super classs user which is extended by two subclasses, Faculty and PhdScholar. For both classes there is a subclass which will inherit functions from the super class interface. User has different data members such as User_id, User_name which are inherited by subclasses, Faculty and PhdScholar.

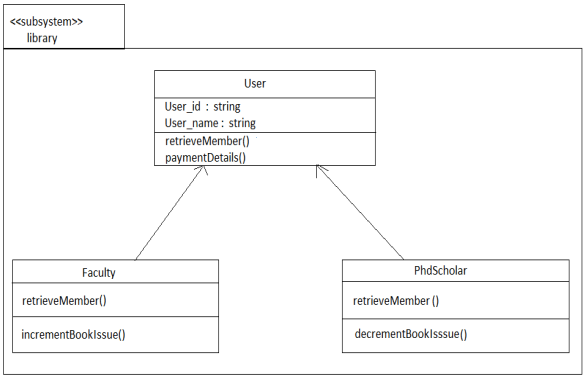

Fig.2. User - Inheritance

3.3 Information hiding

In this approach, by using similar interface, various versions of a component can be created . There are different versions of same component which is called variant[2]. Variability is defined in these versions. A reuser can select any one of the versions and can use it in application. In information hiding variability is obscured in different version of the same interface. In LMS case study, an interface payment fine() has two version payment card() and payment cash().

Fig. 3 shows this implementation. Payment card() and payment cash() implements payment fine(). These two versions shows the variability.

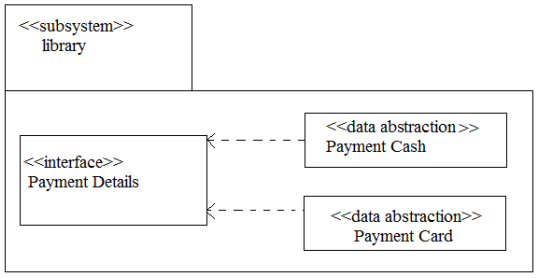

Fig.3. Payment – Information hiding

Fig.4 shows a class called catalog which list of all book items. It can be implemented with two interfaces show_authorname and show_subjectwise. show_authorname shows list of all books according to authors and show_ subjectwise shows all the books according to subject wise. Variability is hidden between these two interfaces.

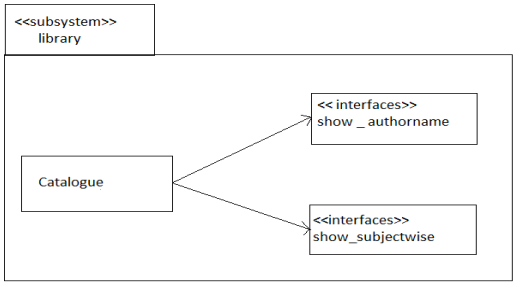

Fig.4. Catalogue – Information hiding

ISSN 2348 – 7968

4 MODELLING VARIABILITY IN ACTIVITY DIAGRAMS THROUGH PARAMETERIZATION

There are various mechanisms described by PESOA [?] In which variability is identified in activity diagrams. Activity diagrams are very commonly used uml diagrams in software engineering. It is a flow chart which describes control flow between activities(functions).

In this mechanism variability is realized through parameter values . A reuser can select any value from the range of values.

4.1 Decision Nodes

Variation can be realized in control flow by parameterizing decision nodes[?]. In the given example there is a class employee which has two attributes regular and contractual . Now Bonus which is a decision nodes is a variation point. If employee is contractual then he will not be rewarded with bonus but to the regular employee bonus will be rewarded.

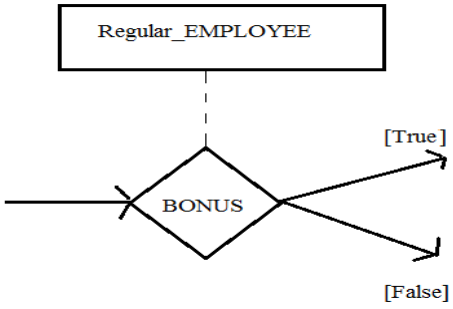

Fig 5. Parameterization- Decision nodes

4.2 Join Nodes

In this mechanism join node will generate an issue for a particular condition. Token in arc should contain an object which should satisfy that particular condition.

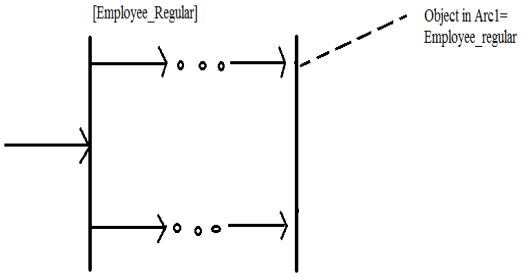

Fig 6. Parameterization - Join nodes

4.3 Data Flow Between Two Actions

Another approach in parameterization is data flow b/w two actions. Let's cosider data flow between two action now it is parameterizable by employing parameter set X and Y.

In this example there are two actions 'Eligibility Criteria' and 'Interview' , now X is a parameter set for Assistant lecture and Y for Associate lecture X and Y has different values to complete action 1.

For example, X{B.TECH, M.TECH} and Y{B.TECH, M.TECH, PHD}. Now assistant lecture should satisfy parameter set X and associate lecture should satisfy parameter set Y for the next action to be taken.

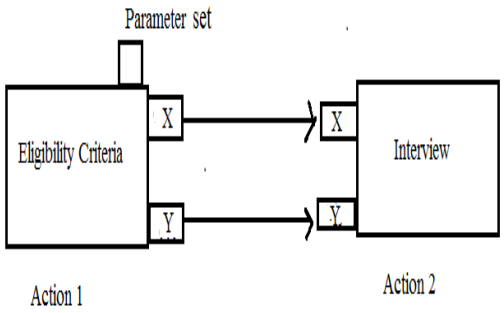

Fig 7. Parameterization - Data Flow

ISSN 2348 – 7968

UML DIAGRAMS	DETAILS	TOOL	METHODS
CLASS DIAGRAMS	DIFFERENT METHODS TO IDENTIFY VARIABILITY IN CLASS DIAGRAMS , *Modeling variability in software product lines with the variation point model*	FODA	PARAMETERIZATION, INHERITANCE, INFORMATION HIDING
ACTIVITY DIAGRAMS	APPROACHES ARE USED TO IDENTIFY VARIANT POINTS,PESOA		PARAMETERIZATION(DECISION NODES, JOIN NODES, ACTIVITY EDGES, DATA FLOW B/W ACTIONS) INHERITANCE
STATE DIAGRAMS	APPROACHES ARE USED TO IDENTIFY VARIANT POINTS, PESOA		PARAMETERIZATION, ENCAPSULATION, INHERITANCE, DESIGN PATTERNS
FEATURE MODELS	APPROACHES ARE USED TO IDENTIFY VARIANT POINTS	SPLOT	
BPMN	APPROACHES ARE USED TO IDENTIFY VARIANT POINTS,PESOA		ENCAPSULATION, PARAMETERIZATION, INHERITANCE

5 COMPARISION BETWEEN DIFFERENT VARIABILITY APPROACHES IN DIFFERENT UML DIAGRAMS

ISSN 2348 – 7968

6 ISSUES IN HANDLING VARIABILITY

There are various issues in handling variability. Major among them is lack of tools .There are very few tools available for representing variability. And whatever systems exists, they are not user friendly. There is a major fault in correctness and consistency. It also has limitations in quality attributes like performance overheads.

7 SUMMARY

Every software has its own demands but some of them are little variant of each other. This paper describes three different approaches, modeling variability using parameterization, inheritance, information hiding. Examples from LMS is used to provide an explanation of different approaches. Variability provides a crucial role in software reuse to extend its horizons. Through this paper, author has analyzed some approaches to model variability.

References

[1] Matthias Galster, Danny Weyns, Dan Tofan, Bartosz Michalik, and Paris Avgeriou, "Variability in Software Systems—A Systematic Literature Review", IEEE transactions on software engineering, vol. 40, no. 3, march 2014.

[2] Diana L. Webbera, Hassan Gomaab, "Modeling variability in software product lines with the variation point model", ELSEVIER, April 2003.

[3] Jilles van Gurp, Jan Bosch, Mikael Svahnberg, "On the notion of variability in software product lines", IEEE ,Software Architecture,August 2001.

[4] Jilles van Gurp, Jan Bosch, "Design , implementation and avolution of object oriented frameworks: concept and guidelines",Software: Practice and Experience,Volume 31, Issue 3, pages 277–300, March 2001.

[5] Deepak Kumar Sharma, Hitesh, Varun Rao , "Individualization of Process Model from Configurable Process Model constructed in C-BPMN" , IEEE, computing, Communication & Automation (ICCCA), 2015.

[6] Hassan Gomaa, Diana L Webber, Modeling Adaptive and Evolvable Software Product Lines Using the Variation Point Model, Proceedings of the 37th Hawaii International Conference on System Sciences – 2004.

[7] K. Kang, Feature-Oriented Domain Analysis, Technical Report No. CMU/SEI-90-TR-21, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA, 1990.

[8] H. Gomaa, Designing Concurrent, Distributed, and Real-Time Applications with UML, Addison-Wesley, 2000.

[9] H. Gomaa, Object oriented analysis and modeling for families of systems with the UML, in: Proc. IEEE International Conference on Software Reuse, Vienna, Austria, June, 2000.

[10] G. Booch, J. Rumbaugh, I. Jacobson, The Unified Modeling Language User Guide, Addison-Wesley, Reading, MA, 1999.

[11] M. Svahnberg, J. van Gurp, and J. Bosch, "A Taxonomy of Variability Realization Techniques," Software—Practice and Experience,vol. 35, no. 8, pp. 705-754, Apr. 2005.