Transformation from LEL to UML

Megha
ME (Software Engineering)
CSE Department
Thapar University, Patiala

Shivani Goel
Assistant Professor
CSE Department
Thapar University, Patiala

ABSTRACT

Software Product Lines have emerged as a well-known approach for software reuse. Requirements of product line are organized into features in Feature Oriented Domain Analysis approach. Feature models are widely used to model the information gathered during domain analysis and it is not simply comprehensible to stakeholders. During the early stages of software development the interaction with stakeholders is mainly inconvenient. For this explanation, natural language (Language Extended Lexicon) is still widely used to model requirements information. It is in general understandable by stakeholders thus encouraging their participation but LEL does not provide design level of a system. To obtain design level of a system there is need to transform LEL symbols to UML class diagram as its elements are in abstract form representing blue print of a system. To achieve this transformation we describe in this paper a transformation process to derive a UML class diagram from natural language oriented requirement model, known as Language Extended.

General Terms

Language Extended Lexicon, UML class diagram, Feature Models

Keywords

UML class diagram, Natural Language, Requirements Models, Software Product Lines, Feature Models, Language Extended Lexicon.

1. INTRODUCTION

Software Product Line Engineering is a procedural frame for engineering Software Product Lines (SPLs). SPLs are developed as an entire and share many assets, together with this increasing reusability. The problem space in SPLs is captured with feature models. A feature model, first introduced by Kang [1], is represented with a feature diagram. It includes actions for identifying and modeling the commonalities and variabilities of a product line in terms of features and for analyzing dependencies between these features. Existing feature models frequently tend to produce very large in size. To treat such large feature models as monolithic entities makes them very hard to understand, develop, manage and evolve. Usually every change performed on a feature model must be verified by a group of experts with different expertise [2], which is time consuming as well as costly.

This paper presents Feature Model which introduces the notion of concerns in feature models. Particularly, we distinguish between non-crosscutting concerns called base concerns and crosscutting concerns called aspects. However, a feature model is not simply comprehensible to stakeholders. During the early stages of software development the interaction with stakeholders is mainly inconvenient. For this explanation, natural language is still widely used to model requirements information. In this paper, we develop an example of admission process, which is used to demonstrate the viability of the approach on a real world scenario and to analyze guidelines for the constructions of the feature model of the admission process. Natural language (Language Extended Lexicon) is the only notation that can be read and understood by the stakeholders, hence encouraging their active participation which is crucial in first steps of software development and this paper presents a transformation process to define a UML class diagram from natural language oriented models, concretely the Language Extended Lexicon (LEL).

The paper is organized as follows. A short presentation of a case study is shown in Section 2. In Section 3, it describes a Feature model. Section 4 briefly introduces the LEL. The transformation that obtains an UML class diagram from LEL Model is described in Section 5, where we describe it with the case study. Finally, in Section 6 we present some conclusions and outline possible future work**.**

2. A CASE STUDY

Admission process describes software systems which enable organizations to do admissions over the Internet. They must be able to deal with a large number of students and coordinate multiple resources to deliver the services to the students. This process enables admissions of the students where organizations provide services to an individual. Tables below show LEL symbols belonging to this case study. A partial description is provided for each symbol.

3. FEATURE MODEL

Feature models are modeling details, used in the context of domain analysis, which considers features as the source for analyzing and describing commonalities and variabilities of systems within a domain [3, 4, 5]. Feature modeling is defined as a mechanism to manage variability in a system family. A feature is an attribute or quality of a system which is of interest to a stakeholder and each feature represents a common or variable aspect of a product. A feature model represents the typical features of a family of products in the domain and relationships between them. i.e., tree-like structures consist of nodes that represent features of modeled SPL and their interrelationships. It describes the decomposition of features into sub-features in a hierarchical manner. For each sub-feature below a certain feature it can be specified if it is mandatory, alternative or optional. There are

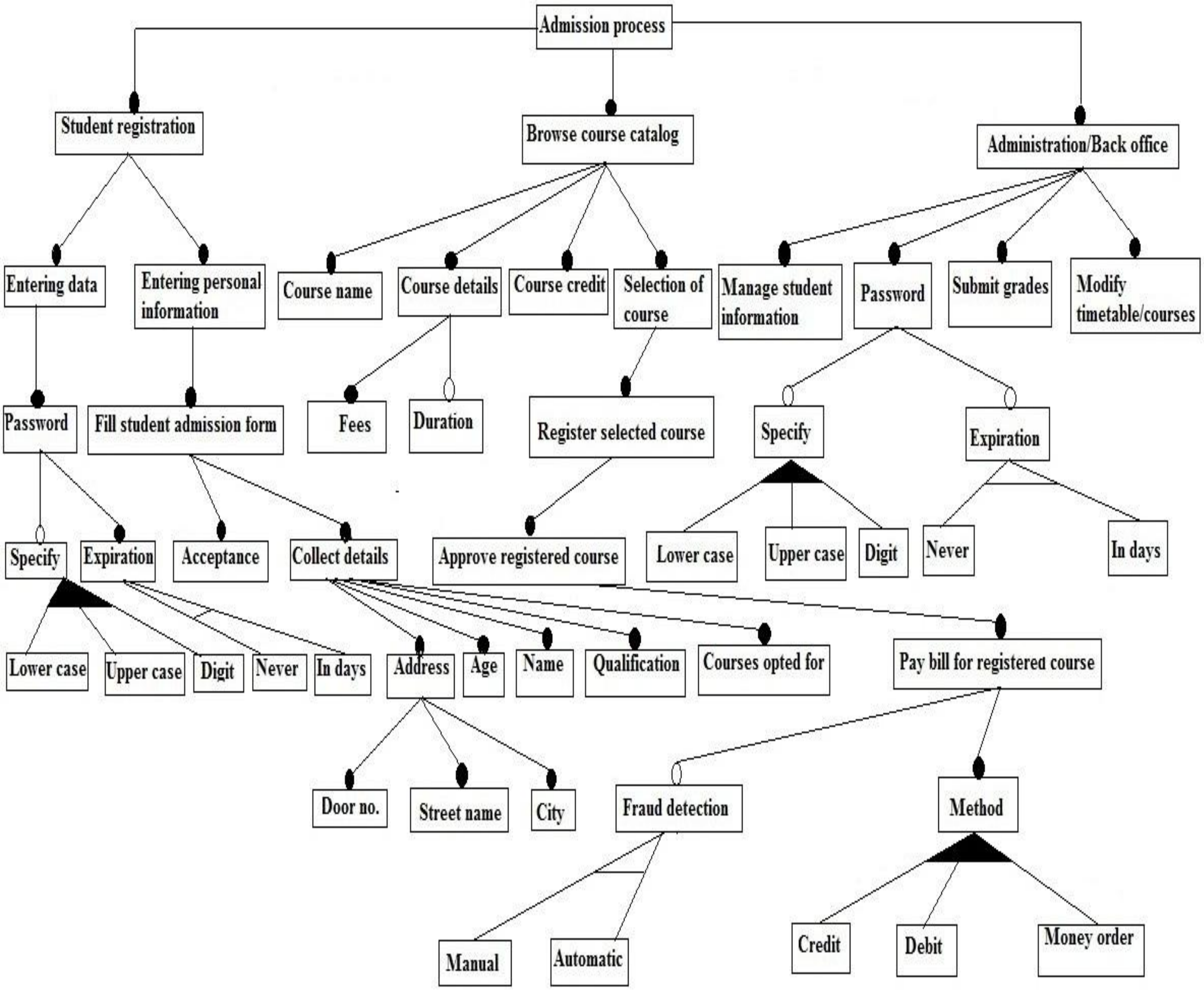

Fig 1: Feature model of admission process

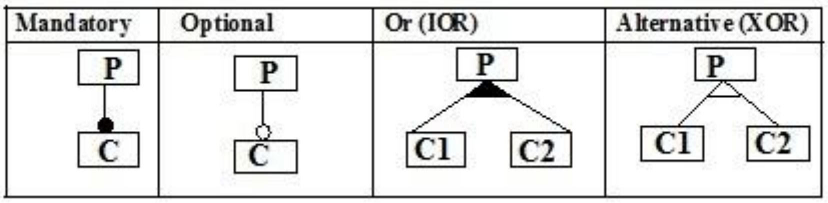

Fig 2: Description of types of features

four types of features in feature modeling: Mandatory, Optional, Alternative, and Or [6].

A mandatory feature has to be integrated in every member of a product line family, if its parent feature is included (e. g, the relationship between the student registration and entering data features), Optional features characterize choices to choose from, depending on requirements for a particular product(e. g, the relationship between course details and duration features). In case of Alternative feature, exactly one feature from a set of Alternative (XOR) features must be included, if a parent of the set is included(e.g., the set consisting of the Manual and Automatic features of fraud detection in an admission process application can be either manual or automatic, but not both), an (or) feature describes a group of features from which at least one of the features must be included in a product line family (e.g., the group consisting of the Credit Card, Debit Card, and Money Order features so as to enable users to pay for selected courses, at least one payment method must be selected). Commonalities between the product lines are modeled as common features which are named as mandatory features. Variabilities are modeled as variable features which are classified as “alternative features”, “or features” or “optional features”. These features are organized into a feature model as Fig 1, shows a feature diagram of an admission process. An admission process typically consists of three parts, one dedicated to the interface that the student uses for registering and one dedicated to the interface that the student uses for browsing the course catalog and other concerned with administration/back office operations. The student registration enables the entering data and entering personal information. During the process of entering data, users specify their passwords. The password sub-tree on the left hand side specifies that each admission process application has their own user password specification (specify) and expiration policy. A user password specification(specify) policy includes the use of lower case (L. Case) and upper case (U. Case) characters as well as digits. A user password may never expire, or the expiry may be specified in days. During the process of entering personal information it includes fill student admission form. Further fill student admission form might mandate the use of details collection and acceptance. During details collection it mandate the use of name, age, qualification, address and courses opted for. An address might mandate the use of door no., street name and city. During the process of browse course catalog, it enables the course name, course details, course credit and selection of course. The course details sub-tree on the left hand side specifies that each course has their own fees and duration. The selection of course enables register selected course which further enables approve registered course. The approve registered course might mandate the use of pay bill for registered course. In addition, the pay bill for registered course allows for different payment methods and fraud detections. Administration/Back office operations for this admission process are operations of manage student information of the admission process effectiveness, modification of timetable and courses (Modify timetable/courses), submit grades and password. The latter consists of managing information in the administration/back office such as product data (Content management) and managing operational concerns of the admission process such as site search and domain name setup (Store administration). Each administrator/back office of an admission process must have their own password as specified by the password sub-tree on the right hand side. A password policy for administrating an admission process application includes the use of lower case (L. Case) and upper case (U. Case) characters as well as digits and a password may never expire, or the expiry may be specified. At last, feature models also include feature relationships that cannot be captured with a tree structure. Such relationships are called integrity constraints [7]. These constraints are includes and excludes constraints. An includes constraint specify a relationship between two features that ensures that one feature is chosen when another one is. In the feature model in Fig 1, an includes constraints exist between the student registration and the browse course catalog features. The excludes integrity constraint specifies a relationship between two features that ensures that one feature is not chosen when another one is. Such an integrity constraint exists between the credit card and the manual features. In our work we suggest introducing semantics to the feature model of a system, through natural language LEL specifications. These specifications explain constraints and preconditions, also allowing specifying relations for features.

4. LANGUAGE EXTENDED LEXICON

A feature model is not simply comprehensible to stakeholders. During the early stages of software development the interaction with stakeholders is mainly inconvenient. For this explanation, natural language is still broadly used to model requirements information. The language extended lexicon is an illustration of the symbols in the language. The principle of the language extended lexicon (LEL) is to confine the vocabulary used in the universe of discourse (UofD) [8]: “Universe of discourse is the general perspective where the software should be developed and operated. The universe of discourse includes all the sources of information and all recognized people connected to the software”. The LEL aims at registering significant vocabulary in the universe of discourse (UofD). Natural language is the only notation that is commonly readable and understandable by the stakeholders, thus encouraging them to participate dynamically in first steps of software development. The LEL is anchored to an easy idea: the focus of the LEL is on the language, rather than the details of the problem. Each entry in the LEL has a name and is composed of notions and behavioral responses. The notions should aim to capture the meaning of the symbol and its relationships with other entries. The behavioral responses state the results, which come from the use of this symbol in UofD, or any effect caused by another symbol to this one. When describing symbols in the LEL, two principles [9] must be followed: The circularity principle prescribes the maximization of the practice of LEL symbols when describing LEL entries, whereas the minimal vocabulary principle prescribes the minimization of the practice of symbols which are exterior to LEL when describing these symbols. LEL symbols may be classified according to its common use in the UofD.

Object

Notion: It defines the object and identifies additional objects with which the object has a relationship.

Behavioral response: It describes the actions that might be applied to the object.

Subject

Notion: It defines who the subject is.

Behavioral response: It registers actions executed by the subject.

State

Notion: It describes what it means and the actions which might trigger the state.

Behavioral response: It describes other situations and actions related to it.

Verb phrase

Notion: It describes who executes the action, procedures involved in the action and when it happens.

Behavioral response: It describes the constraints on the occurrence of the action, identifies the action triggered in the environment and new situations that occur as the consequence.

The above proposed to suggest what to include in the notion and behavioral response of a symbol according to what the symbol define. The entire elements of the model must pursue the same vocabulary as used in the lexicon. The words that are highlighted are signs of the LEL, used in accordance with the principle of circularity. LEL symbols in the admission process case study are given below:

Table 1. Admission process (Subject) LEL symbol

Notion	Behavioral Response
• It is a software system for the admission of students. • It is composed of **student registration**; **browse course catalog** and **administration/back office**. • It has a starting date. • It has an application form. • It has an approximated period of duration in days. • It may have one or more employee.	• It may enable **student registration**. • It may enable **browse course catalog**. • It may enable **administration/ back office**.

Table 2. Administration/back office (Subject) LEL symbol

Notion	Behavioral Response
• It is the interface that administrator uses to access **admission process**. • It is a component of the **admission process**. • It is composed **of manage student information, modify timetable/courses, submit grades** and **password**. • It may have one or more employee. • It has a unique identification.	• It **manages student information**. • It **modifies timetable/courses**. • It **submits grades** of student. • It manages **password**.

Similarly LEL symbols for the subject's student registration and browse course catalog can be obtained.

Table 3. Entering data (Verb) LEL symbol

Notion	Behavioral Response
• It is the process of **entering data** of a student. • Performed by students. • It contains **password**.	• It accepts the **entered data**. • It manages the **password**.

Similarly LEL symbols for the verbs entering personal information, selection of course, manage student information, modify timetable/courses and submit grades can be obtained.

Table 4. Course name (Object) LEL symbol

Notion	Behavioral Response
• It describes the **name** of the courses. • It is a component of **browse course catalog**. • It has a list. • It may be in alphabets.	• Each **course name** must have unique id. • Student choose course according to **course name**. • It displays the list of courses.

Similarly LEL symbols for the objects course details, course credit and password can be obtained.

Table 5. Fill student admission form (Verb) LEL symbol

Notion	Behavioral Response
• It is the process of **filling student admission form**. • Performed by student. • It contains **details collection** and **acceptance.**	• It may accept student **admission form**. • It gives **details collection**.

Similarly LEL symbols for the verbs register selected course, specify and expiration can be obtained.

Table 6. Password (Object) LEL symbol

Notion	Behavioral Response
• It is the process of providing security to the data entered by the student. • It is a component of **entering data**. • It contains **specify** and **expiration**. • It has a size. • It has unique identification. • It has an approximated period of duration in days.	• It provides authorization to students to access data related to **password.** • It determines **expiration**. • It maintains **specify**.

Similarly LEL symbols for the objects fees and duration can be obtained.

Table 7. Approve registered course (Verb) LEL symbol

Notion	Behavioral Response
• It is the process where **administration/back office approves registered course**. • Performed by **administration/back office**. • It contains **pay bill for registered courses**.	• The student can access to registered courses.

Similarly LEL symbols for the verbs specify, expiration and acceptance can be obtained.

Table 8. Collect details (Subject) LEL symbol

Notion	Behavioral Response
• It is the process of collecting student details. • It contains **name, age, qualification, address** and **courses opted for**. • It is a component of **fill student admission form.**	• It provides the details of student.

Table 9. In days (Object) LEL symbol

Notion	Behavioral Response
• It defines that **password** may expires **in days**. • It has number of days. • It may be a week. • It has a starting date.	• It may provide some fix days.

Similarly LEL symbols for the objects lower case, upper case, never and digit can be obtained.

Table 10. Pay bill for registered course (Verb) LEL symbol

Notion	Behavioral Response
• It is the process of **paying bill for registered course**. • Performed by the student. • It contains **fraud detection** and **method.**	• The student has access to registered course. • May have **fraud detection**. • May have **methods** to **pay bill for registered course.**

Table 11. Name (Subject) LEL symbol

Notion	Behavioral Response
• It is the **name** of student. • It is a component of **details collection**. • It has assigned a unique identification. • It has assigned a unique roll no.	• It provides the details of student **name**.

Table 12. Courses opted for (Object) LEL symbol

Notion	Behavioral Response
• It is where the student selects the **courses opted for.** • It is a component of **details collection**. • It has a unique identification number. • It has an author. • It has a title.	• Student can opt one or more courses.

Similarly LEL symbols for the objects age, qualification, address, never, in days, lower case, upper case and digit can be obtained.

Table 13. Fraud detection (Subject) LEL symbol

Notion	Behavioral Response
• It is the process of detecting fraud in case of **paying bill for registered courses**. • It is composed of **automatic** and **manua**l. • It has a transaction id. • It is a component of **pay bill for registered course.**	• It may enable **manual fraud detection**. • It may enable **automatic fraud detection**.

Similarly LEL symbols for the subject method can be obtained.

Table 14. Door no. (Object) LEL symbol

Notion	Behavioral Response
• It defines the **door no**. in the **address** of a student. • It is a component of **address**. • It may have number.	• It must be in numeral form. • Must have unique number.

Similarly LEL symbols for the objects street name and city can be obtained.

Table 15. Credit (Object) LEL symbol

Notion	Behavioral Response
• It is the process through which a student can **pay bill for registered course.** • It is a component of **method**. • It has a card. • It has a bank account. • It has an account holder. • It has card number.	• Student has to enter pin number.

Similarly LEL symbols for the objects manual, automatic, debit and money order can be obtained.

5. THE TRANSFORMATION PROCESS USING A CASE STUDY

In this section we describe a transformation process [10] to derive a UML class diagram of admission process from a natural language oriented requirements models (LEL). The process consists in the application is a set of transformation rules which are applied to natural language oriented models(LEL) to derive classes, attributes, methods and relationships taking into account the structure and semantic of LEL. The above LEL symbols are used as source to derive the UML class diagram. The process takes as the source model a LEL model from a case study of admission process and follows the steps described below to organize the application of the transformation rules:

5.1 The Transformation Rules

5.1.1 Rule 1: Transformation of Subject to Class

• Each LEL subject becomes a UML class.

• For each entry in the notion of this LEL symbol that does not contain a LEL, the transformation identifies each noun and defines them as attributes.

By applying the transformation Rule 1 to the subject LEL symbol admission process that is Table 1, the class shown in Fig 3 is defined.

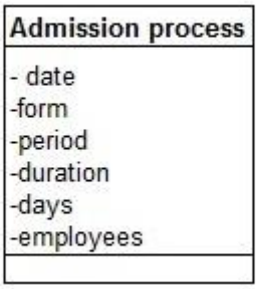

Fig 3: Admission process class

As Rule1 indicates, second entry in the notion is discarded because it contains another LEL symbol from the other five entries; the rule identifies nouns and defines them as attributes. One of the major problems of this transformation is that it misses noun groups. As the current implementation only detects separate nouns, every noun is a possible attribute, thus generating more and sometimes inappropriate attributes. By following linguistic approaches, noun group's detection may be included [11].

5.1.2 Rule 2: Transformation of Object to Class

• Each LEL object becomes a UML class.

• For each entry in the notion of this LEL symbol that does not contain a LEL symbol, the transformation identifies each noun and defines them as attribute.

• Methods to access and modify each attribute are defined by adding SET and GET prefixes for each attribute.

The application of the transformation Rule 2 to the object LEL symbol password that is Table 6, gives as result the class and attributes shown in Fig 4. By applying the manual heuristics from [12, 13]**,** we would have obtained the following attributes: identification, size, period of duration (Rule 2 takes each of them separately). In the last case, the problem is that the dictionary does not recognize noun groups, as we have mentioned before. Moreover, the attribute days obtained after applying Rule 2 would not be an attribute following the manual approach because human judgment would have realized they are the way in which periods are considered.

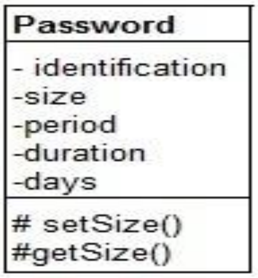

Fig 4: Password class

5.1.3 Rule 3: Transformation of Subject Behavioral Response to Method

• Each entry in the behavioral response of a LEL subject that was modeled as a class by Rule 1 becomes a method of this class.

Applying the transformation Rule 3 to the LEL symbol admission process shown above in Table 1, the methods described in Fig 5 are obtained.

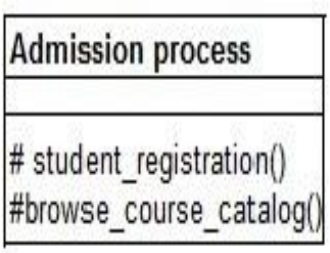

Fig 5: Methods of admission process class

5.1.4 Rule 4: Transformation of Subject Information to Method Parameter

• Each scenario comes from an entry in the behavioral response of a LEL subject that was modeled as a UML class.

• The rule models actors and resources of each scenario as parameters of the method obtained by Rule 3 from the entry in the behavioral response that originated the scenario. The actor referring to the subject LEL symbol in consideration is excluded.

For example, for each method previously defined by Rule 3 (Fig 5), parameters are identified considering the scenarios involved. As parameters come from resources and actors, they are modeled as classes when the corresponding resource and actor is a subject or object LEL symbol (Rule 1 and Rule 2). When the resource or the actor does not belong to the LEL, two things may occur. It may be a word that does not need a LEL entry because it belongs to the minimum vocabulary or it may represent a set. In the former case, it is modeled with a primitive class or type, and in the later one no new classes are needed because the parameter is a set of a class which is already defined.

5.1.5 Rule 5: Transformation of LEL Relationships to Class Relationships

This transformation applies to subject as well as object LEL symbols.

• The entries in the notion of each symbol in the LEL (called L1) modeled as a class is analyzed in order to detect other symbols in the LEL modeled as classes.

• For every detected LEL symbol (L2), this rule defines a relationship between the corresponding classes, analyzing the verb involved to determine the type of relationship, taking into account the following issues**:**

Inheritance relationships: L1 and L2 have the same classification (object or subject). As well, L1 appears in one of the entries of the notion of L2. The concerned entries of L1 and L2 contain, two kinds of verbs, in a complementary way [14]: bottom-up verbs (is a, is a type of, is a class of) or top-down verbs (is, may be, may be classified as, classifies as). The rule does not consider if the methods of the subclasses are refinements of the corresponding super class methods. The software engineer is the one who must take the appropriate decision depending on the semantic of every case.

Aggregation relationships: In the entries of the notion of the LEL symbol considered as container, verbs of the type "component_composition_verb" must show [14]: "to consist / to contain / to include / to form, to compose, to divide" (these three last in passive voice). In the entries of the notion of the “component” symbol, verbs of the type content_composition_verb must appear [14]: "it is part, it belongs, it is a component, it is included", among others. Since it is not likely to automatically distinguish between an aggregation and a composition relationship, the transformation rule defines the relationship as an aggregation.

Association relationships: Any relationship between LEL symbols that does not represent a relationship of the previous types, represents an association. The verb in the entry of the notion (classified as general verb in [14]) is taken as the name of the association.

By applying the transformation Rule 5 to the LEL symbols given below we obtain a Hierarchy, with method as the super class and credit, debit and money order as subclasses.

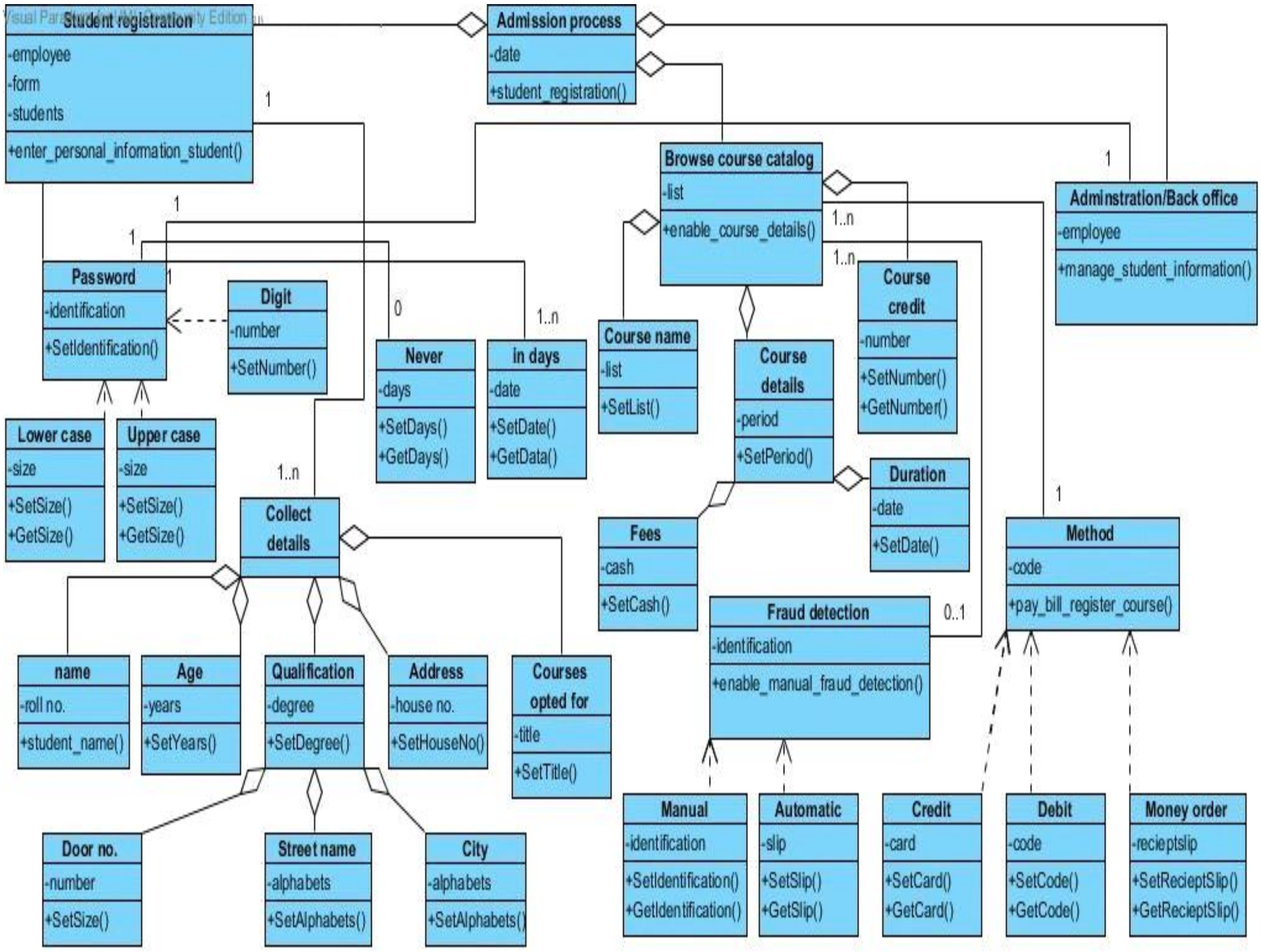

Fig 6: UML class diagram of admission process

Method (Subject)
Notion

- It is the process by which a student can **pay bill for registered course.**
- It contains **credit, debit** and **money order**.

Credit (Object)
Notion

- It is the process through which a student can **pay bill for registered course.**
- It is a component of **method**.

Debit (Object)
Notion

- It is the process through which a student can **pay bill for registered course.**
- It is a component of **method**.
- It has a card.

Money order (Object)
Notion

- It is the method through which a student can **pay bill for registered course.**
- It is a component of **method**.
- It has a account.

Fig 6 shows the UML class diagram which was defined by considering the structure and the construction process of LEL model. It is important to mention that this strategy must be complemented with the participation of software engineers who will adjust the results obtained after the application of the transformation rules.

6. CONCLUSION

Feature Model can be used as the source for software development in the context of domain analysis. It describes commonalities and variabilities of systems within a domain. It would be essential to include linguistic approaches to achieve a better processing of the information. Natural language is used to describe the requirements of a software system during the initial stages of software development because it contributes to elicit, model, and communicate requirements in an easy and friendly way. LEL provides a complete description of an application. LEL helps to set and merge the domain concepts and also encourages and facilitates stakeholders contribution and on the other side, the accurate structure and well-defined construction process has eases the manipulation of the information it models, and thus this information may be reinterpreted into more formal descriptions for example, a UML class diagram. The use of LEL and UML class diagram is motivated by the truth that stakeholders frequently speak of product characteristics in terms of “the product has and/or delivers” using them in order to communicate their problems, ideas and needs. In this paper we have described a transformation process to obtain a UML class diagram from natural language oriented requirements models (LEL) in order to obtain design level of a system as the elements in class diagram are in abstract form which represents the blue print of a system. Transformation rules involve fix decisions about certain modeling issues. It produces a better and more precise model definition, transformation rules are a starting point to deal with the requirements information. We must improve some transformation rules in order to attain a better definition of the relationships between UML classes. By defining another corresponding and independent model to capture and signify the relationships produced by the application of the transformation rules we can improve this transformation process.

7. REFERENCES

[1] Bowman, M., Debray, S. K. and Peterson, L. L. 1993. Reasoning about naming systems.

[2] Mannion, M., Savolainen, J. and Asikainen, T., 2009, Viewpoint-oriented variability modeling. In proceedings of the IEEE International Computer Software and Applications Conference.

[3] Czarnecki, K. , Helsen, S. and Eisenecker, U., 2004, Staged configuration using feature models. In proceedings of the SPLC on Software Product Line, pp. 266–283.

[4] K. Kang, S. Kim, J. Lee and K. Kim, “FORM: a feature-oriented reuse method with domain-specific reference architectures”, Annals of Software Engineering, 1998.

[5] Lee, K., Kang, K. and Lee, J., 2002, Concepts and guidelines of feature modeling for product line software engineering. In proceedings of the ICSR-7 on Software Reuse.

[6] Czarnecki, K. and Eisenecker, U., 2000, Generative programming: methods, tools, and applications, Addison-Wesley, Reading, MA, USA, June 2000.

[7] Benavides, D., Segura, S. and Ruiz-Cortes, A., 2010, Automated analysis of feature models 20 years later: A lit. review. Inf. Systems, Vol. 35(6), pp. 615 - 636.

[8] Leite, J.C.S.P. and Franco, A.P.M. , 1993 , A Strategy for Conceptual Model Acquisition. In Proceedings of the First IEEE International Symposium on Requirements Engineering, pp. 243-246.

[9] Leonardi, M.C., Mauco, M.V., Felice, L., Montejano, G., Riesco, D. and Debnath, N. 2009. Integrating Natural Language Oriented Models with Feature Model. IEEE. 185-190.

[10] Leonardi, M.C. and Mauco. M.V., 2009, Integrating Natural Language Requirements Models with MDA, Encyclopedia of Information Science and Technology, Second Edition,Khosrow-Pour M. (editor). IGI Global, USA. ISBN 978-1-60566-026-4, pp. 2091-2102.

[11] Diaz, I., Pastor, O., Moreno, L. and Matteo, A., 2004, Una Aproximación Lingüística de Ingeniería de Requisitos para OO-Method, Proc. IDEAS'2004: IIV Workshop Iberoamericano de Ingeniería de Requisitos y Desarrollo de Ambientes de Software, Perú, May 2004, pp.270-281.

[12] Leonardi, M.C., 2001, Una Estrategia de Modelado Conceptual de Objetos basada en Modelos de Requisitos en Lenguaje Natural. Master Thesis. Facultad de Informática, Universidad Nacional de La Plata, La Plata, Argentina.

[13] Mauco, M.V., 2004, A Technique for an Initial Specification in RSL. Master thesis. Facultad de Informática, Universidad Nacional de La Plata, La Plata, Argentina.

[14] Juristo, N., Moreno, A. and López, M., 2000, How to Use Linguistic Instruments for Object-Oriented Analysis, IEEE Software, Vol. 17(3), pp. 80-89.