EXTENDING THE UNIFIED MODELING LANGUAGE FOR DEVELOPING OR UPGRADING AN APPLICATION

Rahul R. Singh*

Yogesh S. Patil*

Abhijit P. Ingale*

Abstract

The UML is mostly used for visualization, specification, documentation and construction of an application. The same concept can be use when we extend any application just by extending the UML. In this paper we have analyze the different application which uses the concept of extended UML, just like Extending UML for Android Applications, Extending UML for Object-Relational Database Design, Extending UML With Enterprise Architect, Extending UML to Model Navigation and Presentation in Web Applications, Extending UML for Real-Time Component Based Architectures. The advantage of extending UML is that we can easily add the new functionality to the existing system.

Keywords:-

UML, CASE Tools, Database, Stereotype, Constraints.

* Asst. professor, Computer Science & Engineering, SSGBCOET, Bhusawal

I. INTRODUCTION

UML or Unified Modeling Language is a language of graphical description for object-oriented modeling in software development. UML was created to define, visualize, construct and document mainly software systems. Though, the use of UML is not limited to software development. It is also used to model business processes, design systems and visualize organizational structures. [1].

UML Advantages

- UML is object-oriented. As a result analysis results methods and construction methods are semantically close to programming methods using current object-oriented languages;
- UML make is possible to describe the system from virtually all possible points of view and give various aspects of system behavior;
- UML diagrams are quite easy to understand;
- UML extends and allows addition of custom text and graphical types which makes it possible to use not only in software engineering;
- UML is a unified standard and it is dynamically evolving.

Evolution −− not only the ability to adapt, but the ability to develop from adapting −− is the primary means for succeeding in an environment driven by constant change and ever−increasing complexity. Anthropologists and historians study the evolution of societies by analyzing their cultures. Such a study focuses on gaining insight into the social structures that enable a society to evolve. Furthermore, the language used within a culture is critically examined because it provides the means for communication and collaboration among members of the culture. Languages establish the foundation upon which societies are erected and evolve. Languages provide not only means for communication, but also a framework for the knowledge and experiences of the individuals within the society. As a society evolves, its culture and language also evolve via a process of adapting and cultivating the ability to better adapt. It is the ability to adapt and address immediate needs while continuously maturing to anticipate and be better positioned to address immanent needs that enable a society to survive and thrive against various types of adversities.

II. EXTENDING THE UML

UML extended in three main parts as follow:-

A. Stereotypes: A class that defines how an existing metaclass (or other stereotype) may be extended, and enables the use of platform or domain specific terminology or notation in addition to the ones used for the extended metaclass. Certain stereotypes are predefined in the UML, others may be user defined.

B. Constraints*:* Constraints are a semantic condition or restriction. They are standard UML feature. They can be expressed in natural language text, mathematically formal notation, or in a machine-readable language e.g. OCL for the purpose of declaring some of the semantics of a model element. They are used in combination with *tagged* definitions to define the characteristic of stereotypes in profiles. A constraint defines a relationship between model elements that must be true {subset} { xor} (multiplicity not enough for xor). Constraints can be on attributes, derived attributes, associations and classes.

C. Tagged values*:* Many stereotyped model elements have properties that cannot be represented graphically. These can defined in the profile as *tagged definitions* and are shown in the model as tagged values. Tagged values consist of a tag or name and an associated value (e.g. {CoordSystem = LatLong} or Boolean tagged value {abstract} {isPropertyName}). Tagged definitions are the attributes of the stereotypes in the profile and are typically shown in comments attached to the stereotyped model elements.

III EXTENDING UML FOR ANDROID APPLICATION

AndyUML is a new tool for Android that allows you to draw UML diagrams in your Android mobile device. AndyUML relies on the yUML service to render the diagrams. Basically, with AndyUML you can write and store the textual UML definitions to be sent to yUML for its visualization. Of course, this is a drawing tool and not a modeling one but useful nonetheless. Another tool to add to the small but growing list of UML tools for cell phones, ipad and similars.. Figure 1 show a simple class diagram created by using yUML with it code given bellow.

```
[College]<>-orders*>[Order]
[Order]++-0..*>[UPS]
[Order]-[note:Computer System.]
```

The Figure 2 show activity diagram for updating existing system by yUML and code as written below.

(start)->|a|,

|a|->(existing functionality)->|b|,

|a|->(Update functionality)->|b|,

|b|-><c>[want more functionality]->(Make updated functionality),

<c>[satisfied]->(end)

The simple Use-Case diagram drawn by using yUML tool is as shown in figure 3 and code written below.

Use Case Diagram

[staff]-(write a assignment)

(write a assignment)<(Take paper)

(write a assignment)>(Take Pen)

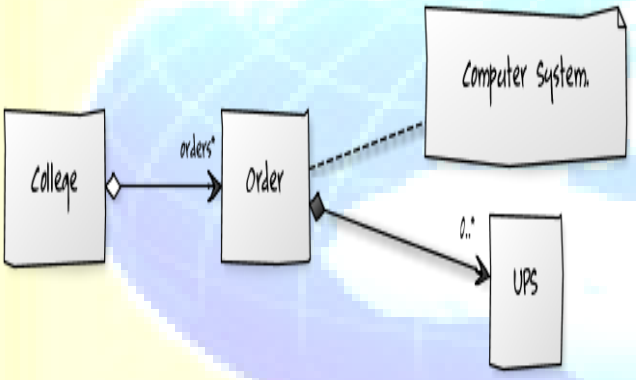


Figure 1 Class Diagram for Order System by yUML.

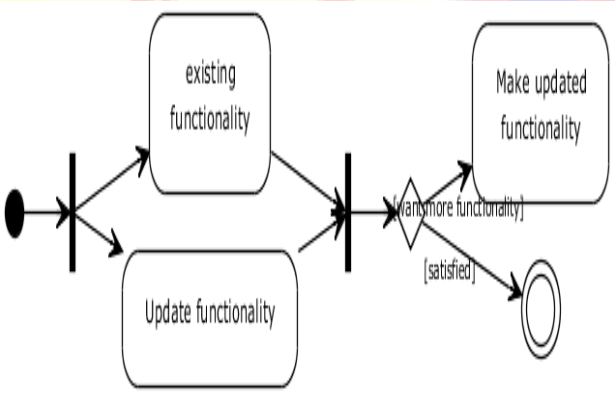


Figure 2. Activity Diagram for updating system by yUML

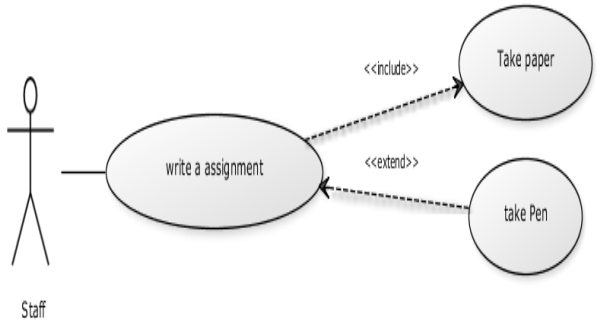


Figure 3. Simple Use-Case Diagram by using yUML.

IV UML EXTENSIONS FOR OBJECT-RELATIONAL DATABASE DESIGN

The Metamodel had been developed to extend UML for object oriented concept. Metamodeling, or meta-modeling in software engineering and systems engineering among other disciplines, is the analysis, construction and development of the frames, rules, constraints, models and theories applicable and useful for modeling a predefined class of problems. As its name implies, this concept applies the notions of meta- and modeling.

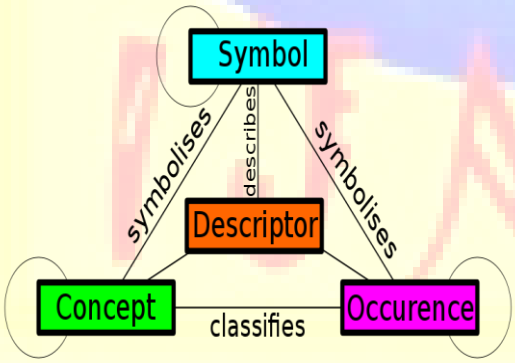


Figure 4. Meta Model to extend UML.

"Metamodeling" is the construction of a collection of "concepts" (things, terms, etc.) within a certain domain. The figure 4 shows a metamodel to extend UML. A model is an abstraction of phenomena in the real world; a metamodel is yet another abstraction, highlighting properties of the model itself. A model conforms to its metamodel in the way that a computer program conforms to the grammar of the programming language in which it is written.

Common uses for metamodels are:

- As a schema for semantic data that needs to be exchanged or stored
- As a language that supports a particular method or process
- As a language to express additional semantics of existing information
- As a mechanism to create tools that work with a broad class of models at run time
- As a schema for modeling and automatically exploring sentences of a language with applications to automated test synthesis

Because of the "meta" character of metamodeling, both the praxis and theory of metamodels are of relevance to metascience [*disambiguation needed*], metaphilosophy, metatheories and systemics, and meta-consciousness. The concept can be useful in mathematics, and has practical applications in computer science and computer engineering/software engineering, which are the main focus of this article.

Metadata modeling is a type of metamodeling used in software engineering and systems engineering for the analysis and construction of models applicable and useful some predefined class of problems. Meta-modeling is the analysis, construction and development of the frames, rules, constraints, models and theories applicable and useful for the modeling in a predefined class of problems. The meta-data side of the diagram consists of a concept diagram. This is basically an adjusted class diagram as described in Booch, Rumbaugh and Jacobson (1999). Important notions are concept, generalization, association, multiplicity and aggregation.

V EXTENDING UML WITH ENTERPRISE ARCHITECT

List of stereotypes are supported by UML, which are an inbuilt mechanism for logically extending or altering the meaning, display and syntax of a model element. Variation model elements have different logic stereotypes associated with them. A stereotype is generally displayed as shown in Figure 5.Applying Stereotypes in an Enterprise Architect enables you to apply one or more stereotypes to any UML construct, including:

- Elements (such as Classes and Objects)
- Relationships (such as Dependencies and Associations)
- Association Ends
- Attributes and Operations
- Operation Parameters.

To apply a stereotype to any UML construct, using the Properties *dialog*, select any one of the following steps:

1. In the Stereotype field, type the stereotype(s) to apply as a comma-separated list.
2. Click on the drop-down arrow and select the required stereotype from the list.
3. Click on the [...] button to use the Stereotype Selector dialog.

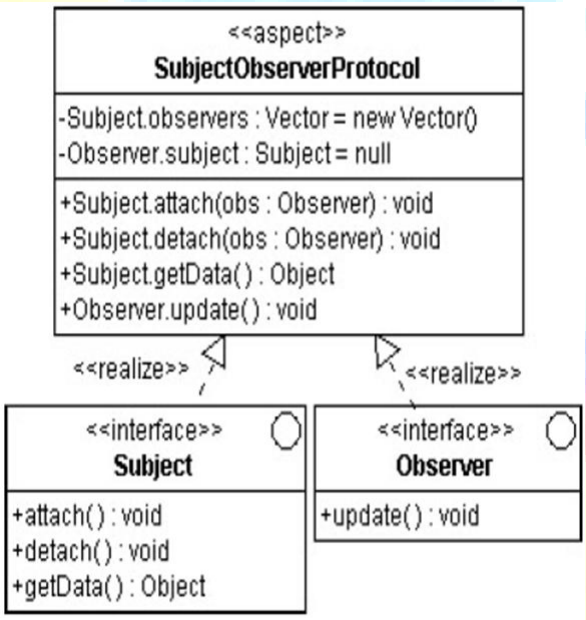
Figure 5. Stereotype.

VI EXTENDING UML FOR REAL-TIME COMPONENT BASED ARCHITECTURES

Separately or together extension mechanisms can be used in UML. The building element of UML may have only single stereotype, but one or more constraints and tagged values. A stereotype is a semantic feature attached to an existing model element. The Stereotypes not only classify model elements at the object model level, but they facilitate the addition of "virtual" UML metaclasses with new metaattributes and semantics. The other extension mechanisms allow users to attach additional properties and semantics directly to individual model elements, as well as to model elements classified by a stereotype. It is important to make clear that stereotypes may extend the semantics, but not the structure of preexisting elements. Any modeling element may have arbitrary information attached to it, such as management information, code generation information or additional semantic information, in the form of a property list consisting of tag-value pairs. A tag represents the name of an arbitrary property with the given value. For example, preconditions and post conditions attached to operations in a class are examples of standard tagged values. Constraints, as well as tagged values, are used to extend the semantics of model elements classified by that stereotype. Constraints can be attached to classes or objects, restricting in this way the semantics of these elements. The constraints must be observed by all model elements marked with that stereotype [2].

VII EXTENDING UML WITH ASPECT SUPPORT IN THE DESIGN PHASE

In the First step aspect should be added to the UML metamodel. The aspect is a construct derived from the Classifier element, which describes behavioral and structural features. All the Class, Interface and Component are kinds of Classifier. Aspect can have attributes, operations and relationships. Attributes of an aspect is used by its set of weave definition. Operations of an aspect are considered as its weave declarations. Relationships of an aspect include generalization, association and dependency. If the aspect weaver uses the kind of weaves, it is specified as a constraint for the corresponding weave (operation) declaration. An aspect is shown as a class rectangle with stereotype <<aspect>> as depicted in Figure 6. The operation list compartment of the rectangle means the list of weave declarations. Each weave is displayed as an operation with the stereotype <<weave>>.



A signature of weave declaration shows a designator; which elements (e.g. classes, methods and variables) are affected by the aspect. Figure 6 does not show the stereotype <<weave>> and the kind of weaves in the operation compartment, because the CASE tool they are using does not display the stereotypes and constraints for operations. Other CASE tool such as Rational Rose can display a weave declaration like: <<weave>> {introduce} Subject.attach():void The attribute list of a rectangle symbol maps into the list of attributes of the aspect [7].

VIII CONCLUSIONS

Finally we conclude that extending UML is efficient way when we are adding a new functionality to the existing system also useful when we are developing any application which required extension in future according to need.

REFERENCES

[1] Sinan Si Alhir. "UML in a Nutshell: A Desktop Quick Reference". O'Reilly Associates, Inc., 1998.

[2] José L. Fernández-Sánchez and Antonio Monzón, "Extending UML for Real-Time Component Based Architectures", 14th International Conference Software & Systems Engineering and their Applications, Paris, France, December 2001.PP 1-10.

[3] Garzotto F and Paoloni P, "Extending UML for modeling Web applications", 'Proceedings of the 34th Annual Hawaii International Conference',2001.

[4] Sauer, "Extending UML for modeling of multimedia applications", 'Visual Languages, 1999.Proceedings.1999 IEEE Symposium', 1999, PP 80-87.

[5] Morision M, "Extending UML to support domain analysis", 'Automated Software engineering', 2000, PP 321-324.

[6] Grady Booch , James Rambaugh and Ivar Jacobson, "The Unified Modeling Language user Guide", Pearson eduction, Low Price Edition,2006,PP159-163.

[7] Junichi Suzuki, Yoshikazu Yamamoto, "Extending UML with Aspects: Aspect Support in the Design Phase", 'Submitted to the 3rd Aspect-Oriented Programming (AOP) Workshop at ECOOP', 1999, PP 1-6.

Figure 6. An aspect as a class rectangle with stereotype