IOSR Journal of Mobile Computing & Application (IOSR-JMCA)
e- ISSN: 2394-0050, P-ISSN: 2394-0042.Volume 7, Issue 3 (May - June 2020), PP 46-48
www.iosrjournals.org

Challenges and Opportunity of UML Diagram for Software Project development as a complete Modeling Tool

Ketema Kifle Gebretsadik[1]
Debre Markos University, School of Computing, Institute of Technology, Debre Markos, Ethiopia

Abstract: *UML(Unified Modeling Language) is a most useful method of visualization and documenting software systems design.UML uses object oriented design concepts and it is independent of specific programming language. Unified Modeling Language is a popular technique for documenting and modeling system. The UML uses set of symbols to represent graphically the various components and relationships within the system and UML can be used for business processing modeling and requirements modeling, it mainly is used to support object oriented system analysis and to develop the object models. Many articles describe UML features, but only very few of them discuss its downside in software design. This article discusses the downside of UML as a complete modeling tool for software design. Some of the disadvantages of UML areno specification for modeling of user interfaces, business rule specification a group exists for this within theObject Management Group(OMG), so we should see something in UML and Poor for distributed systems are no way to formally specify serialization and object persistence.Even though UML have many advantages it has also their owndownside for software design.*

Keyword: *UML, Challenge of UML, Software design, UML diagrams*

Date of Submission: 22-05-2020 Date of Acceptance: 09-06-2020

I. Introduction

Unified Modeling Language (UML) is a software modeling language with an emphasis on graphics and motion. It is the industry standard language for software modeling and design, according to Sparx Systems. However, some developers and software design companies may experience issues using UML.

World development and lifestyle increasingly depend on software. Software-intensive systems, as technological achievements, as well as social demands, are growing in size, complexity, distribution and importance. However expansion of these systems changes the limits of software industry know-how. As a result, building and maintenance of software becoming increasingly complex. Various software development projects fail in different ways but they all share common symptoms. Some of them are: inaccurate understanding of end user needs; inability to deal with changing requirements; software that is not easy to maintain or extend or late discovery of serious project flaws.Some of the problems and their causes can be avoided by implementing more rigorous developmentprocess. Deployment of notation language, like UML, might facilitate communication between allparticipants in the development process(1). These are some of the reasons for UML utilization in softwaredevelopment. Present article analyses their downside of UML use in the software design.

Currently the object-oriented paradigm is the most popular paradigm in software engineering. Together with object orientation, the Unified Modeling Language (UML) has become the most popular modeling language for analysis and design(2). In fact, UML has some obvious shortcomings: -Tool support is not satisfactory. There are still various tools that do not support all UML elements. Only few tools are suitable from analysis up to implementation. The reporting functions of most of the tools are not adequate. Models are not interchangeable between tools.Apart from the obvious shortcomings of the UML we found another one: many of the real-world problems are faced with interfaces to up/downstream systems, look and feel of the user interface, are not addressed by the UML diagrams.

UML includes nine diagram for describing system(3):

- Class diagram describes set of classes, interfaces and their relationships. It shows static design view of a system .This diagram is very useful in modeling object-oriented systems.
- Object diagram shows set of object and snapshots of instances of instances of the things found in class diagram.
- Use case diagram shows a set of use cases, actors and their relationships. This diagram especially important in organizing and modeling the behavior of a system.
- Sequence and collaboration diagrams are interaction diagrams, which describe interaction between objects.
- State chart diagram shows a state machine consisting of states, transitions events and activities.

- Activity diagram is special kind of state chart diagram. It emphasizes a flow from activity to activity within system.
- Component diagram describes organization and dependencies among a set of components. It is related to class diagram among a set of components. It is related to class diagrams in a way of mapping component to one or more classes, interfaces, or collaborations.

II. Challenge of UML Diagram for software Project modeling

2.1 Major disadvantage of UML design in code generation

System design with UML is very dependent on the type of the system under development. Theexperiments support the conclusions arrived at. UML is much more appropriate for designing the systems having complex static architecture (e.g. data and GUI-related systems)(4).The applied UML tool has code generators for several most interesting programming languages.Preconditions for code generation are completeness and correctness of the model. Hence, prior to code generation there is always an automatic check of the system design.

Major disadvantage of UML design in code generation is the loss of much information. Codegenerators use only class diagrams. Complete behavior has to be implemented manually. In somecases, most of the work is done through sequence and statechart diagrams. Class diagrams are very simple, representing static relations rather than dynamic behavior. Using UML include and adding tasks to a project's work scope and relying on UML diagrams too heavily.

One disadvantage some developers might find when using UML is the time it takes to manage and maintain UML diagrams. To work properly, UML diagrams must be synchronized with the software code, which requires time to set up and maintain, and adds work to a software development project. Small companies and independent developers might not be able to handle the added amount of work required to synchronize the code.

When creating a UML diagram in conjunction with software development, the diagram might become overwhelming or overcomplicated, which can be confusing and frustrating for developers. Developers cannot possibly map out every single scenario for a software tool in the diagram, and even if they try to, the diagram gets messy. One way developers can combat this issue is to only include basic facts and high-level information in UML diagrams, according to a post on Stack Overflow by Stefano Borini, a quantum chemist and UML developer.

UML places much emphasis on design, which can be problematic for some developers and companies. Looking at a software scope in a UML diagram can lead to software project stakeholders over-analyzing problems, as well as cause people to lose focus by spending too much time and attention on software features. Companies cannot solve every problem with a software tool using a UML diagram -- eventually, they just have to start coding and testing. Brody Gooch, a co-creator of UML, said that the original vision for UML was a "graphical language to help reason about the design of a system as it unfolds." If people get hung up using a diagram to identify and solve issues, it can delay the actual work that needs to be done to fix the issues.

III. Opportunity of UML Diagram

3.1. Advantage of UML Diagram for Software Project modeling

UML is a highly recognized and understood platform for software design. It is a standard notation among software developers. You can safely assume that most software professionals will be at least acquainted with, if not well-versed in, UML diagrams, thus making it the go-to alternative to explain software design models.

What makes UML well-suited to and much-needed for software development is its flexibility. You can customize your modeling elements and interactions in a UML diagram specifically to suit the domain or technologies you are using (5).

The main weakness of UML, from our point of view, is its lack of formality. Clearly to build an executable model we need a formal language. This leaves the question of why we would want to do this. There are two possible reasons. One is that UML is used in a development process and we wish to use it for our stage of this process for consistency and to avoid the user translating work into a new language. The other related reason is simply that if users are familiar with UML it might be worth keeping it, as a “front end” to a formal language(4).

The other Limitations of UML is it brings a set of notations and concepts that meet the needs of typical software modeling projects but some users have found UML unable to express their modeling needs.

- Flexibility should be added to construct and document more heterogeneous and complex systems.
- UML lacks features that would allow to attach non-semantic information to models.
- Component models and architectural frameworks (JavaBeans, CORBA Component Model and COM+ cannot be modeled easily with UML.

- Formalization hard to deal with for non-expert users
- Too detailed formalization could create confusion between project stakeholders

IV. Weakness Of UML Diagram

4.1 Weaknesses of Use Cases and Use Case Diagrams

Use cases themselves have no formal documentation standard. This leads to confusion and debates on what comprises a good use case. Most projects proceed on the basis of a predetermined standard for documenting use cases. However, this lack of standards creates an opportunity for modelers and project managers to develop their own standards, as well as their own interpretation of what needs to be documented.Disadvantages of use cases and scenarios(6)Poor identification of structure and flow, Geometric and temporal information hard to describe, Unsystematiccraft, Time-consuming to generate, Limited software tool support, Handling unforeseen combinations of abnormal events, Require the co-existence of prototypes, Still poor integration with established methods, Difficult to generalize from scenarios – abstract use cases and Scenario management is difficult.

4.2 Weaknesses of Activity Diagrams

Activity diagrams have no structural characteristics, and they do not provide direct information on how the system or the requirements are organized and prioritized. Activity diagrams represent use case behavior pictorially. However, they do not usually give a complete picture of the system. For large and complex use cases, multiple activity diagrams are required. The inability of activity diagram to view the full requirements of the system at a glance is their weakness(4).

4.3 Weaknesses of Classes and Class Diagrams

Class diagrams do not have any dynamics. They have no concept of time. Therefore, they are only able to represent how the system is structured and what its relationships are. There is no opportunity to model an if-then-else scenario on a class diagram(4). Thus, class diagrams are extremely weak when it comes to modeling the dynamic-behavioral aspect of the system.

4.4 Weaknesses of Sequence Diagrams

Sequence diagrams are not structural in nature and therefore are unable to show any structural relationships between classes. Thus, although sequence diagrams can be used to identify missing classes, they cannot be used to identify precise structural relationships between classes.Although they show behavior in a temporal sequence, sequence diagrams are not pure dynamic diagrams. This is because they show messages exchanged during a particular time or period but are unable to show changes to the state of the object or changes to the values of the attributes of an object(4). And it is difficult to show an if-then-else or a for-next condition on sequence diagrams.

V. Conclusion

Software design is a process of problem solving and planning for a software solution. After the purpose and specifications of software are determined, software developers will design or employ designers to develop a plan for a solution(7). For software design developers use UML tools, the main advantage of UML is Readability and Usability, however it their lack of formality and tools complexity different UML tools have created confusion between experts.

References

[1]. [Online] http://www.c-sharpcorner.com /UploadFile /nipuntomar/uml-diagrams-part-1/..
[2]. Analysis beyound UML. Christiane Stutz, Johannes Siedersleben, Dorthe Kretschmer, Wolfgang Krug. s.l. : sd & m.
[3]. Sasa Desic, Darko Gvozdanovic,Mario Kusek,Darko Huljenic. Advantages of UML-based Object-oriented system development .
[4]. BHUVAN UNHELKAR, PHD. VERIFICATION AND. United States of America : John Wiley & Sons, Inc., Hoboken, New Jersey, 2005. 0-471-72783-0.
[5]. [Online] https://creately.com/blog/diagrams/advantages-and-disadvantages-of-uml/.
[6]. [Online] https://uxapprentice.wordpress.com/2011/11/29/disadvantages-of-use-cases-and-scenarios/.
[7]. UML, By Adwait Moghe, CPSC 606 Presentation, CPSC606.ppt.
[8]. [Online] [Cited: 1 12, 2013.] http://www.c-sharpcorner.com/UploadFile/nipuntomar/uml-diagrams-part-1/.
[9]. Analysis beyound UML. Christiane Stutz, Johannes Siedersleben, Dorthe Kretschmer, Wolfgang Krug. s.l. : sd & m Research.