Using UML in Software Development

Issa H.Manita [1], Osama E.Abufanas [2], Aisha Niebu[3]

[1] *Osama E.Abufanas, Faculty of Information Technology, Misurata, Libya*
[2] *Issa H.Manita, Faculty of Information Technology, Misurata, Libya*
[3] Aisha Niebu, *Faculty of Information Technology, Misurata, Libya*

ABSTRACT

This paper talks about UML and its use as a modeling language for the specification of a system at different stages in its development. The overall purpose of this paper is to talk about the dynamic model and its use to express and model the behavior of the system over time. Thus, the focus of this paper is to present the UML behavioral diagrams for the classroom time management system, and to show how those diagrams are used in describing the functionality of the system.

Keyword : - *UML, Software system, System function, Behavioral diagram, Dynamic diagram.*

---

1. INTRODUCTION

UML refers to Unified Modeling Language, which is the primary modeling language used to analyze, specify, and design software systems [1]. Software architects create UML diagrams to help software developers build the software, and to be used by software developers during the initial stages [2]. If you understand the vocabulary of UML (the diagrams pictorial elements and their meanings) you can much more easily understand and specify a system and explain the design of that system to others [3].
UML diagrams can be classified into two groups: structure diagrams, which are used to show the static structure of elements in the system, and behavior diagrams, which show the dynamic nature of the communications between the objects [2].

2. THE UNIFIED PROCESS

The Unified Process is a software development process. It is more than a single process; it is a generic process framework that can be specialized for a very large class of software systems, for different application areas, types of organizations, competence levels, and different project sizes [4]. The Unified Process is component-based, which means that the software system being built is made up of software components interconnected via well-defined interfaces [4].

The Unified Process uses the Unified Modeling Language (UML) when preparing all blueprints of the software system. UML is an integral part of the Unified Process developed. The real distinguishing aspects of the Unified Process are captured in the three keywords: use-case driven, architecture-centric, and iterative and incremental. This is what makes the Unified Process unique [4].

3. SOFTWARE DEVELOPMENT WITH UML

Having a well-defined and expressive notation is important to the process of software development. First, a standard notation makes it possible for an analyst or developer to describe a scenario or formulate architecture and then unambiguously communicate those decisions to others. As in many other disciplines, the UML is used to model (i.e., represent) the system being built. Taken in total, the UML model that you build will represent, to a certain level of fidelity, the real system that will be constructed [2].

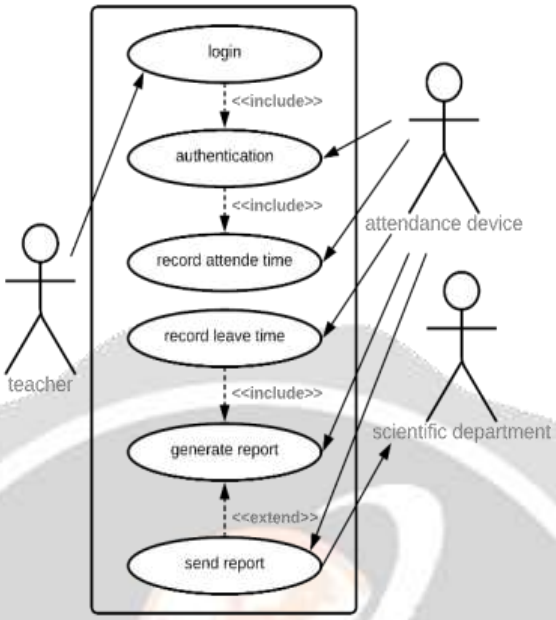

Figure 1. UML use-case diagram for the teacher in classroom time management -system-

4. UML BEHAVIORAL DIAGRAMS

UML behavioral diagrams visualize, specify, construct, and document the dynamic aspects of a system. The behavioral diagrams are categorized as follows: use-case diagrams, interaction diagrams, state–chart diagrams, activity diagrams, timing diagrams, communication diagrams, and sequence diagrams. The next UML models can be useful:

4.1 Use case diagram

The UML use-case diagram helps you determine the functionality and features of the software from the user's perspective. A use-case describes how a user interacts with the system by defining the steps required to accomplish a specific goal, variations in the sequence of steps describe various scenarios. In the use-case diagram, the use cases are displayed as ovals. They show the relationships between the actors that are connected by lines to the use cases that they carry out and none of the details of the use-cases are included in the diagram and instead need to be stored separately [5]. In addition, the use-cases are placed in a rectangle while the actors are not. This rectangle is a visual reminder of the system boundaries and that the actors are outside the system [3].

As shown in Figure1, the use-case diagram helps in analyzing the functional requirements of the system from the teacher's point of view. The right side shows the actor (teacher) and its acts with the system, while the left side shows the actors who react to the teacher-use cases (the attendance device and the scientific departments) and the system will do the functions due to the teacher's actions.

4.2 Activity diagram

A UML activity diagram depicts the dynamic behavior of a system or part of a system through the flow of control between actions that the system performs, or the roles which execute certain actions [5]. It is similar to a flowchart except that an activity diagram can show concurrent flows. The main component of an activity diagram is an action node, represented by a rounded rectangle, which corresponds to a task performed by the software system. Arrows

from one action node to another indicate the flow of control. That is, an arrow between two action nodes means that after the first action is complete the second act begins. A solid black dot forms the initial node that indicates the starting point of the activity. A black dot surrounded by a black circle is the final node indicating the end of the activity. A fork represents the separation of activities into two or more concurrent activities. It is drawn as a horizontal black bar with one arrow pointing to it and two or more arrows pointing out from it. Each outgoing arrow represents a flow of control that can be executed concurrently with the flows corresponding to the other outgoing arrows [3].

For example, Figure 2 shows the activity diagram for the dynamic functions of an admin in the classroom time management system. It starts with the first action that the admin does for accessing the system (with different cases of validation), then it defines the functions and deliveries that admin can get and finally, the last action will take place after ending all the terms and functions (log out).

4.3 Sequence diagram

Sequence diagram is used to show the dynamic communications between objects

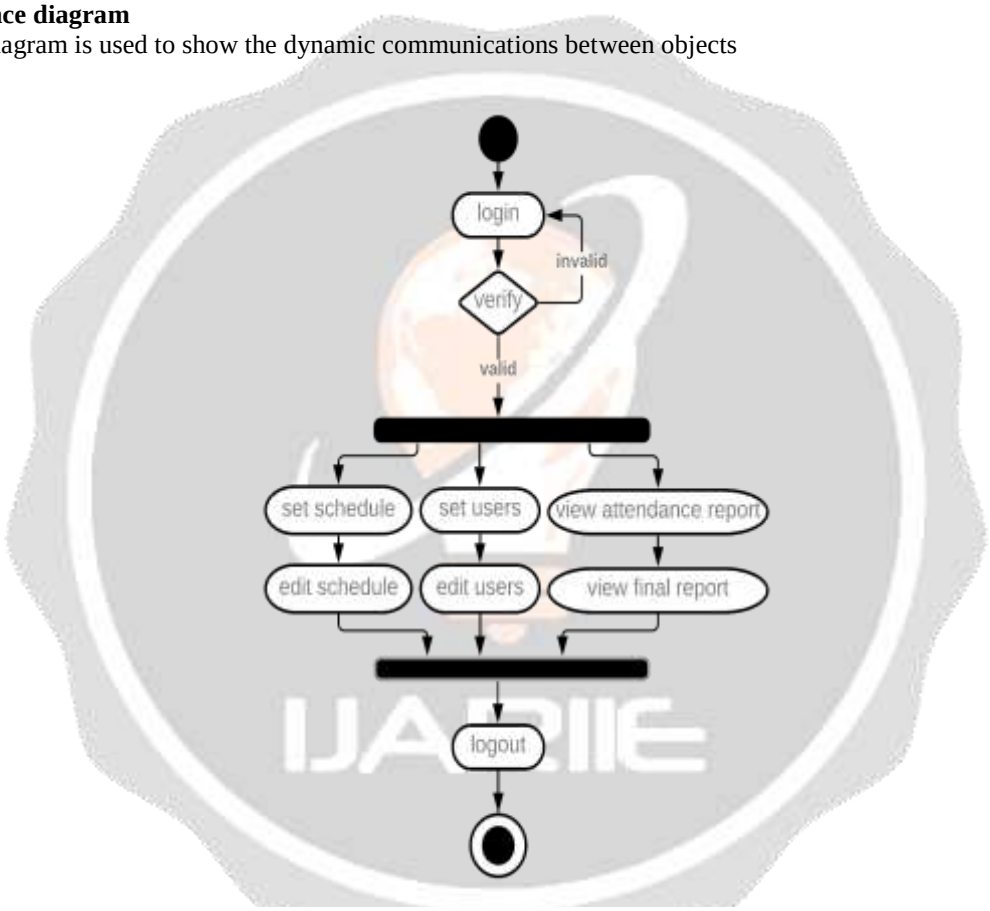

Figure 2. UML activity diagram for the admin in classroom time management -system-

during the execution of a task. It shows the temporal order in which messages are sent between the objects to accomplish that task. One might use a sequence diagram to show the interactions in one use case or one scenario of the software system [2].

In sequence diagrams, the entities of interest are written horizontally across the top of the diagram. A dashed vertical line, called the lifeline, is drawn below each object. These indicate the existence of the object. Messages (which may denote events or the invocation of operations) are shown horizontally. The endpoints of the message icons connect with the vertical lines that connect with the entities at the top of the diagram. Messages are drawn from the sender to the receiver. Ordering indicated by vertical position, with the first message shown at the top of the diagram, and the

last message shown at the bottom, the notation used for messages (the line type and arrowhead type) indicates the type line and arrowhead type indicates the type of message being used. Asynchronous message (typically an operation call) is shown as a solid line with a filled arrowhead. An asynchronous message has a solid line with an open arrowhead [1].
A return message uses a dashed line with an open arrowhead [1].

For the student (end-user) in the classroom time management -system, the UML sequence diagram (Figure 3) would be a good choice for the scenarios and actions and the student will get the messages during the system execution. The first diagram shows that the student will do the first action with the mobile app, then the validation messages would get from the courses system (with different cases), and finally, the student would get the last feature defined by the system admin.

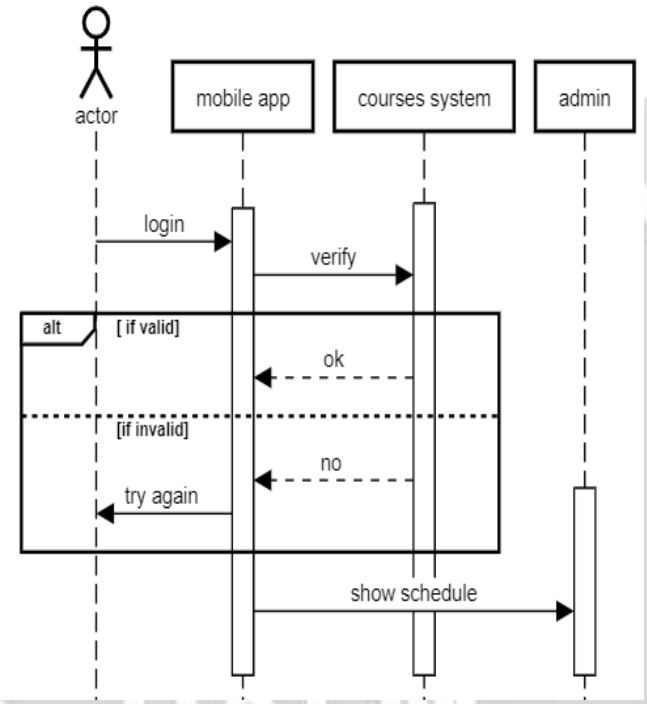

Figure 3. UML sequence diagram for the student in classroom time management -system-

5. CONCLUSIONS

In conclusion, the UML use-case diagram helps in capturing the functional requirements of a system and additionally, it can evolve at each iteration from a method of capturing requirements to develop guidelines for architects. The activity diagram illustrates a business process or workflow between users and the system. The UML sequence charts clearly depict the sequence of events and concurrent operations; thus making them the go-to alternative to explain software analysis, design, and architecture models.

6. REFERENCES

[1]Booch, Grady, et al,Object-oriented analysis and design with applications, ACM SIGSOFT software engineering notes 33.5 (2008): 29-29.
[2]Alhassan Adamu,Wan Mohd Nazmee Wan Zainon, Multiview Similarity Assessment Technique of UML Diagrams, [Available online](24Aug 2021).
[3] Roger S. Pressman, Software engineering: a practitioner's approach, Palgrave Macmillan, 2005.
[4] Jacobson, Ivar, Grady Booch, and James Rumbaugh, The unified process. Ieee Software 16.3 (1999): 96.
[5] Karl Wiegers, Joy Beatty,Software Requirements, Third Edition, 2013.
[6] Lesze A. Maciazek, Bruc Lee Liong, Practical software engineerin", 2005.