**UML Design Modeling**

Adrian Lopez Carreon

The University of Arizona Global Campus

CST499: Capstone for Computer Software Technology

Instructor Amr Elchouemi

June 13, 2022

Before software can be implemented into the user system, the developer needs to ensure that everything is working. Sometimes they will create a line code or erase a line code, which will cause issues with another thing in the software. TheThe way they make sure that everything is working correctly is by putting the software they have created to the test. Different types of testing are done on the software before everyone involved with creating the software feels satisfied with implementing the software into a production environment. Testing Components, integration, system, and acceptance testing are some of the testing levels that need to be done.

Testing the component of a system is a crucial part of the process because that will provide an understanding of the quality of the components. According to Jerry Gao (2000), an ideal testable software component is not only deployable and executable but also testable with the support of standardized components test facilities (p. 4). When component testing, each system component must be tested individually to verify that they are working as they should. Unit testing and component testing are very similar, but developers will test during the unit testing phase, and testers will test during the component testing phase. Component testing plays a significant role in finding any issues that the system might have, and once component testing is done, integration testing should be started afterward.

Integration testing is when different components are grouped and examined to see how they work together. According to Frank Tsui, Orlando Karam, and Barbara Bernal (2018), when dealing with large software systems, functions may be integrated into a component (p. 213). Integration testing plays a vital role in discovering any issues that could occur with different components being incorporated to work as a whole system. Different types of integration testing can be used. One of them is the big bang approach, while the other is the incremental approach, top-down, bottom-up, and hybrid. Integration testing is a crucial testing practice in the software development life cycle to catch any bugs in the software.

System testing is done once the integration testing has been completed to assess the obedience of the system and the requirements. According to Muhammad Abid Jamil, Muhammad Arif, Normi Sham Awang Abubakar, and Akhlaq Ahmad (2016), system testing tests the whole software from every perspective (p. 177). System testing is to verify that the software is working as one significant component and that everything is working as a whole unit. Two categories within system testing consist of black box and white box testing. Black box testing consists of testing the outer workings of the software from the perspective of the user. At the same time, white-box testing consists of the opposite of the black box, testing the software's code. System testing is a necessary act that needs to occur if the developer intends to present a working system that has a minimal amount of issues.

Acceptance testing ensures that the software works as the user expected the software to function. According to Frank Tsui, Orlando Karam, and Barbara Bernal (2018), acceptance testing is the explicit and formal testing conducted by the customers before officially accepting the software product and paying for it (p. 214). If the software meets all the requirements during the acceptance testing, the customer will accept the software as it has been designed. The customer will usually test the software during the acceptance testing to verify that the software will work for them the way it has been created. During the acceptance testing phase, the testers verify that the software has met the needs of the stakeholders according to the requirements given to them during the requirements process. Different types of acceptance testing are distinguished by their functions. Acceptance testing has excellent benefits and should be done when creating software to verify that the software was produced as the requirements were given so that it could be helpful to the customer.

There are many steps in creating the software, and testing the software can be considered one, if not the most important, phase of developing accurate software. Without testing the components, the integration of the components, the system as a whole, and the acceptance testing, many bugs can be that software. It is up to the project manager to ensure that all the software is being tested to ensure that good quality software is delivered to the customer. Testing the software could prevent significant problems in the software and prevent developers from having to fix the issues later. Testing is created to find most of the defects in the software that would not have been discovered until the user has gone and done work on the software.

Figure1. Class Diagram

Diagram

Description automatically generated

Figure 2. Sequence Diagram

Diagram

Description automatically generated

Figure 3. Activity Diagram

Chart, diagram

Description automatically generated

Figure 4. Use Case Diagram

Diagram

Description automatically generated

**References**

Gao, J. (2000, June). Component testability and component testing challenges. In *Proceedings of International Workshop on Component-based Software Engineering (CBSE2000, held in conjunction with the 22nd International Conference on Software Engineering (ICSE2000)*. Retrieved from <https://www.researchgate.net/publication/274080200_gao00>

Jamil, M. A., Arif, M., Abubakar, N. S. A., & Ahmad, A. (2016, November). Software testing techniques: A literature review. In *2016 6th international conference on information and communication technology for the Muslim world (ICT4M)* (pp. 177-182). IEEE. Retrieved from https://www.researchgate.net/profile/Muhammad-Arif-75/publication/312484469\_Software\_Testing\_Techniques\_A\_Literature\_Review/links/5a003444aca272347a2b77f5/Software-Testing-Techniques-A-Literature-Review.pdf

Tsui, F., Karam, O., & Bernal, B. (2018). [Essentials of software engineering](https://ashford.instructure.com/courses/101819/modules/items/5165678)(4th ed.). Jones & Bartlett Learning.