Assignment 2 Reviewed

Israel Balderas

CST499 Capstone for Computer Software Technology

University Of Arizona Global Campus

Professor: Amjad Alkilani

November 16, 2021

The SRS document contained the complete proposed information for the software that is to be developed. It covered all the system requirements that high-level system requirements that were identified as well as lower-level requirements. The next phase of this project is to develop Unified Modeling Language(UML) models to show a better workflow of the work progress. These models will make it much easier for our developers as well as stakeholders to see a greater picture of the software that is to be developed. “Because of its wide reach, UML is the perfect visual language to communicate detailed information about the architecture to the largest number of users.” (Oliver, 2017) A benefit of this is that it allows stakeholders to request changes that can be made before development has started. Last of all, once everyone has agreed on the development plan, we can move on to the next phase, which is the planning for the different levels of testing. Testing is an important phase in the development; having poorly tested software can lead to product failure and lost revenue.

The first UML model that we developed was the Use Case Diagram and is developed to show the different uses that this system may be used for. Examples of uses that were identified were such things as a student dropping or registering for an upcoming class. A professor looking at their current classes and current students enrolled or removing a student from the class. Lastly, a counselor looks at making changes to a student's schedule, either dropping a class or adding a class. Below, we can see an example of the different levels of users and their use cases for the system software.

Diagram

Description automatically generated

For the second ULM model, an Activity diagram was developed that provides another system overview. Like the model sequence, it provides a general system overview, but it shows the workflow that happens when a user tries to use the system. This workflow only shows the sequence of events that occur when a user selects searches for new classes. The user searches for the class scheduled and selects the class they would like. This system checks to see if there is availability if so, adds the user to the class if not, adds the user to the waiting list, and sends the information back to the user. This process can be the scene in the diagram below. Diagram

Description automatically generated

The next phase of the process is the planning for the system test that will be done. Conducting system tests is crucial for the development process; skipping any part of these tests can be a critical mistake. For example, "For staying in business, having a software testing is imperative. A critical defect left undetected can cause losses in business." (Lozančić, 2018) Many mistakes can be uncovered this way, from simple system bugs to major issues that may be used to exploit the system. Regardless of the type of issues, having too many systems bugs can cause degradation and the source of negative user feedback. To ensure the best quality software, different levels of software tests need to be done to ensure functionality from testing at the component level, which tests the smallest modules to integration test and system test ensure groups of components work well together in the system. The last test, the acceptance test, is used to ensure that system has met all the system requirements that's were determined.

One thing the developers will be doing is continuous testing of the products. This means that each individual component developed will be tested to ensure no errors occurs. Some of the components Identified that will be tested are the user accounts, the SQL database, and the payment processing page. The user account will be tested to ensure any new clients can accurately create accounts. The SQL database will be tested to ensure tables are appropriately created and new items can be added or deleted from the webpage. Following the component testing, we can move on to the integration testing to ensure the tested components work well in groups of components. For example, the new user registration page and the new class registration page. We will test that when a new student registers on the site, the information inputted is successfully stored into the student database. Or, for the new class registration page, when a student tries to enroll, the system will check for open spaces, if so, it will assign it to the user and mark the spot as taken. Otherwise, the user will be added to the waitlist and display the class as pending.

Once the component testing and integration testing have been completed, we can move on to the final test. We can put all the tested groups of components together to ensure the system functions without any issues with the system test. This means when a student registers, they should be able to log in with the created login info or manage their class schedule. Multiple items will be tested, attempting to register while logged in, choosing to drop a class, and registering for a full and empty class. If any issues are found, they can be addressed in this testing phase. Otherwise, we can move on to the final test. The final test is the acceptance test, and this test will ensure the system has met all the functional and non-functional requirements. It will also ensure that the system executes as it was intended to do. We will conduct an in-house test to obtain a confirmation from the developers before having the stakeholders perform the last test acceptance test on their onsite hardware. If all these tests are successful, then the software is ready for delivery.

The models presented in this report are only used to give a general overview of the system workings that were determined in this first week's functional requirements. The system can still be changed if there is an issue with one of the presented models. Once these models have been approved, then the official development of and testing can begin. Testing is something that will be done through the entire life cycle of the development ot the software. As mentioned before, it is impossible to address every bug in the system, but with continual testing, we can ensure the highest quality system is delivered.

References:

Gebretsadik, K. K. (2020, May 22). *Challenges and Opportunity of UML Diagram for Software Project development as a complete Modeling Tool*. http://www.iosrjournals.org/. http://www.iosrjournals.org/iosr-jmca/papers/Vol7-Issue3/Series-1/H07034648.pdf.

Lozančić, A. N. A. (2018, March 20). *Benefits of software testing*. Gauss Development. https://gauss-development.com/benefits-software-testing/.

Oliver, R. (2017, October 19). *Advantages and disadvantages of uml every developer should know*. Creately Blog. https://creately.com/blog/diagrams/advantages-and-disadvantages-of-uml/.