**Week 2 Assignment**

**UML Design Modeling**

John Home

The University of Arizona Global Campus

CST 499: Capstone for Computer Software Technology

Dr. Charmelia Butler

6/19/2023

When it comes to designing software it is important to ensure that the system does what it is supposed to from a technical standpoint as well as a functional standpoint. In order to do this testing must be done throughout he lifecycle of the software by different stakeholders in the project to ensure that the software has been properly scrutinized from multiple viewpoints.

The first step to ensuring proper testing can take place is fully understanding what the system is supposed to be accomplishing. Programmers and testers need to understand how the system is going to be utilized in order to determine what they need to test and figure out specific important components to test. To do this the programmers and testers can refer back to the use case documents that would have been gathered when the conversations with stakeholders was happening. For a project to build a class registration system use case scenarios would be like the examples below.

***Use Case 1: Student***

*Student will connect the website that hosts the class registration options for their selected school. The student needs to log into the website utilizing their username and password to ensure the classes they choose are identified with them. Once successfully logged in the student will want to be able to decide what semester they are going to be signing up for. It is possible to sign up for multiple semesters so they need to be able to clearly see what they are working with. It would be good if the student could see any classes or semesters they may have already signed up for before selecting the semester they are going to look at. Once the student has selected the semester they are going for they are going to be presented with a list of available classes for that semester. They should be able to see how many seats are available in the classes. Once they have a class they would like to sign up for they should click on it and see more information about that class and be given the option to register. If the class is full they will get a prompt to sign up for the overflow with details on how the overflow works. Once the student has selected all the classes the have they will log out of the registration website.*

***Use Case 2: Registration Staff***

*In order to check the status and availability of classes the faculty supporting students looking to register will need to be able to sign into the website and see the different courses available for each semester, the number of available seats, and review the number of people on any particular class wait list. The supporting faculty should be able to select any student that is currently registered for a class and validate that the student meets the pre-requisites of the class in order to be accepted into the class and have it added to their schedule. If a student does not meet the pre-requisites the faculty member should be able to send them an email from the system notifying them that they need additional classes and that they have been removed from that class. If students are removed their seat should be made available for other students to register for.*

Once an understanding of how the system is going to be utilized has been reached the use case diagrams can be converted into class diagrams to understand the different objects that will be needed for the system to function. Knowing what objects should exist in the program allows the testers to ensure those objects exist and perform the functions they were intended for. From the use case examples class diagrams can be set up for any objects that will be needed.

A picture containing text, screenshot, diagram, design

Description automatically generated

*Image 1. Class diagram for student registration.*

Once the class diagram is complete an activity diagram can be created showing how the flow of the system should go. The activity diagram shows how the system should pass information and what should happen when certain conditions are met. This allows the testers to know what order the system should operate in and allows them to know what different outcomes they should test for.

A picture containing text, diagram, screenshot, plan

Description automatically generated

*Image 2. Activity Diagram.*

Once the testers fully understand what they should be testing for they can then start to test the different levels of tests to ensure the program meets the requirements and functionality that is desired. The first level of testing that is done is component testing. Component testing is when each component of the system is tested individually to ensure it is able to perform its task without any input or interaction with any other components. Component testing is done early to help identify any issues that may exist before the whole system is put together and reduce the amount of trouble shooting needed at later stages (Devi, 2023). From the class diagram in figure 1 each class would be tested to make sure that the class stored the information that it needed and was formatted so that it could be retrieved and passed on to other components. The testers would also test individual processes like drop down menus and the wait list to ensure they stored information and presented the correct options. This way once those components were integrated if there were issues they would know it was something in the integration and not with the component itself.

The second level of testing is integration testing. Integration testing is the process of combining all the individual components and testing their interactions with one another (Carpenter, 2021). For integration testing the testers are gong to take each component and determine which components are going to need to interact with one another. Once they know what components interact with one another they link the components together to ensure they pass the information they need successfully and do not end up passing incorrect information or fail to perform any needed calculations. Integration testing can be tricky because it is hard to know if the interactions of different module will interfere with one another or if something is missing until the whole program is assembled.

The third level of testing is system testing. System testing is when all the different components have been tested with each other and finally the whole system has been assembled and can be tested to see if it meets requirements (Bernal, Karam, & Tsui, 2018). System testing focuses on ensuring the technical requirements for the system have been met. The testers will go through all the operations and make sure that they are outputting the correct information to the GUI and the end users.

The last level of testing is acceptance testing. Acceptance testing is when the testers are made up of stakeholders and employees that are actually going to be utilizing the software. This is important because the testers will be less educated on the internal working of the software and focus more on how it feels to utilize the software and if it will fit their job role. This testing helps to ensure the software meets the functionality requirements and will fit in the intended environment. Generally for acceptance testing the end users will utilize the software in production space and then provide feedback to the developers of any changes they feel needs to be made or if there are any major issues they found that prevents them from using the software in the manner the needed.

**References**

Bernal, B., Karam, O., & Tsui, F. (2018). Essentials of software engineering (4th ed.). Jones & Bartlett Learning. <https://platform.virdocs.com/r/s/0/doc/2348054/sp/294215359/mi/864710041?cfi=%2F4%2F2%5Bch06%5D%2F6%2F4%5Bch06-sect1-002%5D%2F4%2C%2F1%3A0%2C%2F1%3A0&menu=table-of-contents>

Carpenter, A. (2021). What is Integration Testing? <https://www.codecademy.com/resources/blog/what-is-integration-testing/>

Devi, K. (2023). What is Component Testing? <https://www.browserstack.com/guide/what-is-component-testing>