**Week 2 Assignment**

Justin Stewart

CST499: Capstone for Computer Software Technology

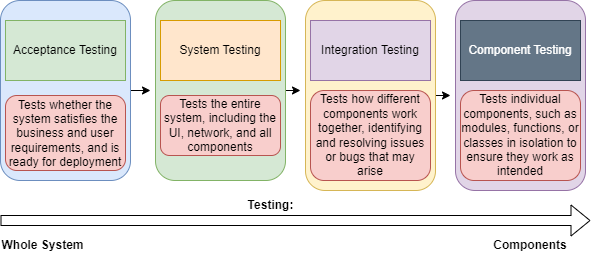
Prof. Elchouemi

3/13/2023

**Introduction**

Software testing is a critical part of the software development process, ensuring that the final product meets the specified requirements and is free of errors. There are different types of testing that are performed at different stages of the software development life cycle (SDLC). In this paper, we will describe four types of testing - component testing, integration testing, system testing, and acceptance testing, and explain their relationships with various UML diagrams, including class diagrams, sequence diagrams, activity diagrams, state diagrams, and use case diagrams. This paper will also include figures to illustrate the four types of testing and each type of UML diagram created for a proposed online course enrollment system.

**Figure 1: 4 Types of Testing**

****

**Types of Testing**: **Component Testing**

Component testing, also known as unit testing, is a type of testing that focuses on testing individual components or modules of a software application in isolation (Spillner, Linz, & Schaefer, 2014, p.42). The objective of this testing is to verify the correctness and functionality of each component by testing them in isolation. Component testing can be performed using different techniques such as integration testing, and white box testing, and may help in identifying defects at an early stage, as well as enabling faster debugging.   
 When it comes to useful diagrams for component testing, class diagrams are used to model the components, representing the classes, attributes, and methods. They help to identify the components and their relationships, making it easier to write test cases that cover all the different scenarios (Tsui, Karam, & Bernal, 2016, p.152). Below, Figure 2.1 provides an example class diagram created for a proposed online course enrollment system.

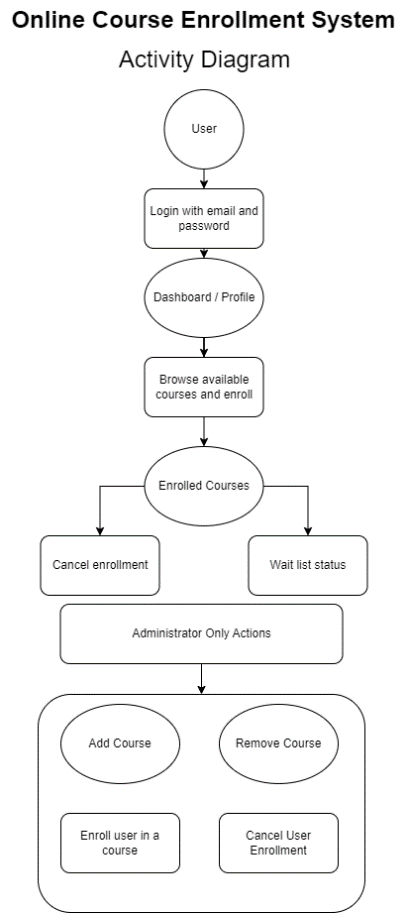
**Figure 2.1: Online Course Enrollment System Class Diagram**

**Diagram

Description automatically generated**

Sequence diagrams are also useful in component testing, representing the interactions between objects and components. They show the order in which methods are called, making it easier to identify and isolate any issues. Activity diagrams are useful for modeling the workflow of a component, helping testers to understand the sequence of steps required to execute a particular function. Figure 2.2 provided below, gives an example of an activity diagram created for a proposed online course enrollment system.

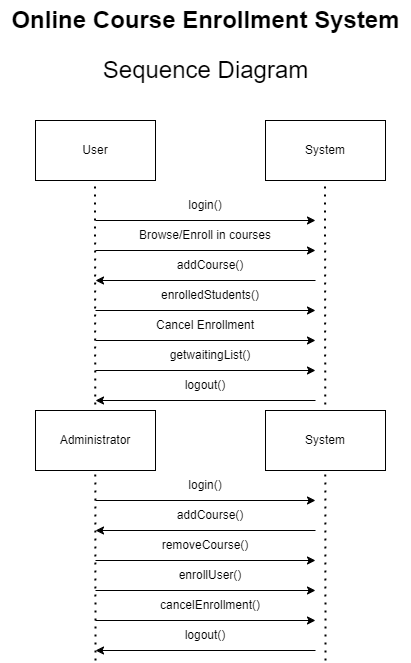
**Figure 2.2 Online Course Enrollment System Activity Diagram**

****

**Types of Testing**: **Integration Testing**

Integration testing is a type of testing that tests how different components of a software application work together as a single system (Awati, 2022, para.1). It verifies that the components integrate seamlessly and function as a single unit. Integration testing requires a more comprehensive approach to testing, and test cases are designed to verify the system's functionality as a whole. Sequence diagrams are useful in integration testing to show the interactions between the components and how they communicate with each other. We can see this in practice for the same online course enrollment system through Figure 2.3 provided below.

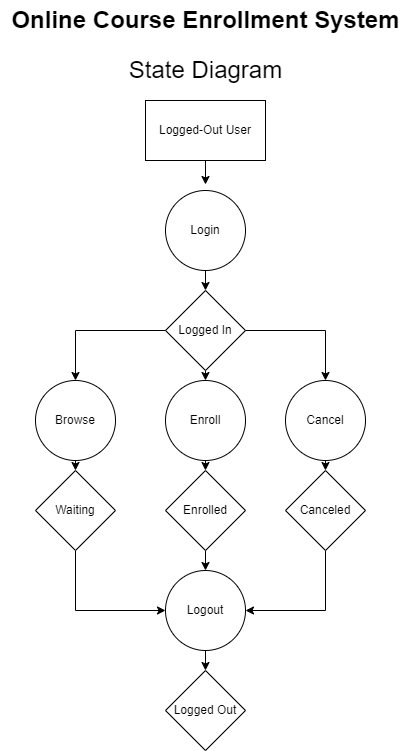
**Figure 2.3 Online Course Enrollment System Sequence Diagram**

****

**Types of Testing**: **System Testing**

System testing involves testing the entire software system as a whole, including all components, integrations, and external dependencies. The objective of system testing is to ensure that the software system meets the specified requirements and functions as intended (Spillner, Linz, & Schaefer, 2014, p.60). System testing may be conducted through a variety of means such as functional testing, performance testing, usability testing, and security testing. State diagrams are useful in system testing as they show the different states that the system can be in and the transitions between them (Tsui, Karam, & Bernal, 2016, p.155). They help to identify and isolate any issues that may occur during the testing process. Below, an example state diagram created for an online course enrollment system is provided in Figure 2.4.

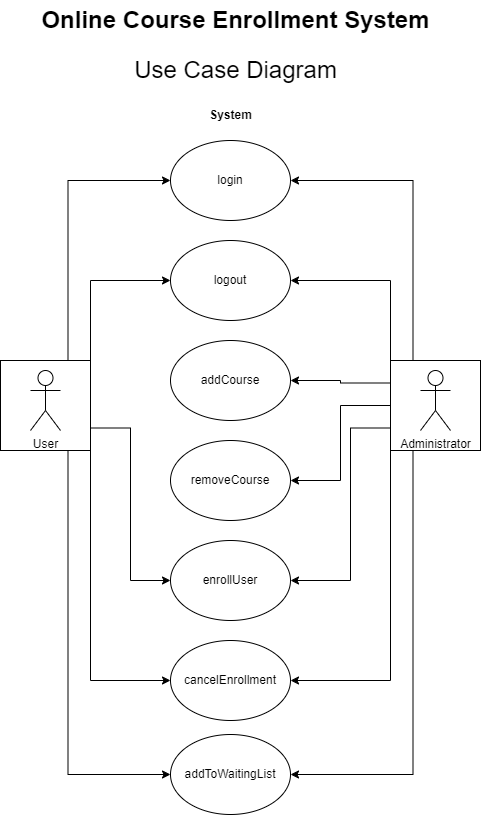
**Figure 2.4 Online Course Enrollment System State Diagram**

****

**Types of Testing**: **Acceptance Testing**

Acceptance testing is the final level of testing that involves testing the software system's functionality against the stakeholders' requirements. It is usually performed by the end-users or the stakeholders themselves to ensure that the software system meets their needs and expectations, and is ready for deployment (GeeksforGeeks, 2022, para.2). Use case diagrams are useful in acceptance testing as they show the different scenarios in which the software system will be used. They help to ensure that the software system meets the stakeholders' requirements and that all the necessary functionality is present. Figure 2.5 provides us with an example of just how that is accomplished, using the same example of a proposed online course enrollment system.

**Figure 2.5 Online Course Enrollment System Use Case Diagram**



**Conclusion**

In conclusion, software testing is a vital phase in the software development lifecycle, and each level of testing requires a specific approach to ensure that the software system meets the requirements and specifications defined by the stakeholders. UML diagrams, including class diagrams, sequence diagrams, activity diagrams, state diagrams, and use case diagrams, are useful in each level of testing, helping testers to identify and isolate issues so that they are certain the software system meets the stakeholders' requirements.

**References**

Awati, R. (2022, January 13). What is integration testing (I&T)? Software Quality.

Retrieved March 13, 2023, from https://www.techtarget.com/searchsoftwarequality/definition/integration-testing#:~:text=Integration%20testing%20%2D%2D%20also%20known,be%20coded%20by%20different%20programmers.

GeeksforGeeks. (2022, June 15). *Acceptance Testing: Software Testing*. GeeksforGeeks.

Retrieved March 13, 2023, from https://www.geeksforgeeks.org/acceptance-testing-software-testing/

Master Diagram Link:

https://drive.google.com/file/d/1czj8TvHKJHTpkC9QBJT\_Qp62g6hK8WEu/view?usp=sharing

Spillner, A., Linz, T., & Schaefer, H. (2014). Software testing foundations: A study guide for

the certified tester exam (4th ed.). Rocky Nook.

Tsui, F., Karam, O., & Bernal, B. (2016). Essentials Of Software Engineering (4th ed.). Jones \

and Bartlett.