UML DESIGN MODELING

Darren Tramell

CST 499 Capstone for Computer Software Technology

Instructor: Joseph Rangitsch

March 17, 2025

**Introduction**

In developing a Learning Management System (LMS) website, rigorous testing is essential to ensure the system meets functional, performance, and user requirements. Testing is conducted at multiple levels, each addressing specific aspects of the system's functionality and integration. This paper refers to the four major levels of testing—component testing, integration testing, system testing, and acceptance testing—in terms of developing an LMS. The paper refers to pictorial support from the diagrams provided, describing the important processes and interactions between them within the system. This article is intended to cover all the stages involved in testing within the development of an LMS, highlighting their significance in providing a stable and user-friendly platform.

**Component Testing**

Component testing, or unit testing, tests the correctness of one component or module of the LMS. For example, the user registration process, represented in the sequence diagram (Figure 1), is a process of operations involving input validation, uniqueness validation of ID, and database storage. Component testing ensures that every operation is correct as an individual entity. According to Tsui, Karam, and Bernal (2018), component testing is necessary to detect defects early during development in order to reduce the cost and effort of fixing issues at a later time. In the LMS case, this would involve writing test cases for features such as checking user input and ID uniqueness checks against the database.

*Figure 1: Sequence Diagram (sequence\_diagram.png)*

A screenshot of a computer screen

AI-generated content may be incorrect.

Figure 1 illustrates the sequence of steps involved in the user registration process, including input validation, ID uniqueness checks, and database storage.

**Integration Testing**

Integration testing evaluates the interactions between different components or modules of the LMS. For instance, after the user registration module is tested, it must be integrated with the course enrollment module. The activity diagram (Figure 2) illustrates the way the system checks course capacity and admits users or places them on a waiting list. Integration testing is tasked with ensuring that modules communicate and data passes from one to the other appropriately. Connolly and Hoar (2022) highlight that integration testing plays a critical role in determining interface defects and ensuring integrated parts perform as intended. Testing user scenarios such as registering and then enrolling in a course to verify end-to-end behavior would be included in the LMS.

*Figure 2: Activity Diagram (activity.png)*

*A diagram of a course

AI-generated content may be incorrect.*

Figure 2 demonstrates the process of checking course capacity and enrolling users or adding them to a waiting list, highlighting the interactions between system components.

**System Testing**

System testing verifies the LMS as a system in entirety, whereby all components added function as if within an actual environment. Testing for performance, security, and usability falls under this category of testing. The state diagram (Figure 3) shows various user states, for instance, "Enrolled" and "Waitlisted", and transitions between these states. System testing would make sure that such transitions are dealt with appropriately and that the system meets the specified requirements (Tsui et al., 2018). Note that system testing plays a vital role in verifying the general behavior of the system and ensuring that the system meets the users' expectations. For the LMS, this would involve user login testing, course registration testing, and waiting list management testing under actual conditions.

*Figure 3: State Diagram (state\_diagram.png)*

*A screen shot of a computer screen

AI-generated content may be incorrect.*

Figure 3 outlines the various states a user can be in, such as "Enrolled" or "Wait listed," and the transitions between these states, which are critical for system testing.

**Acceptance Testing**

The acceptance test is the final phase, where the LMS is tested to see if it satisfies business needs and can be launched. This typically includes user acceptance testing (UAT), where the system is tested by the users to see if it is appropriate for their needs. The use case diagram (Figure 4) defines key features such as course management, viewing enrollment status, and user registration. Acceptance testing verifies that this kind of functionality performs as needed and that the system is user-friendly. Connolly and Hoar (2022) write that acceptance testing is critical in achieving stakeholder acceptance and that the system delivers the intended value. In the LMS context, this would involve testing the system with real users to gather feedback and make the last-minute changes before going live.

*Figure 4: Use Case Diagram (Use\_case\_1.png)*

*A diagram of a learning process

AI-generated content may be incorrect.*

Figure 4 provides an overview of key functionalities such as managing courses and viewing enrollment status, which are validated during acceptance testing.

**Conclusion**

Developing an LMS site requires a well-planned test cycle that encompasses component, integration, system, and acceptance testing. There are significant roles for each level of testing to make the system execute, function, and be usable. Adhering to a total testing strategy allows developers to identify and correct problems at each stage of development, ultimately delivering a solid and stable LMS. These mentioned diagrams provided informative information regarding processes and interactions that need to be tested at any level. Following Tsui et al. (2018) and Connolly and Hoar (2022), effective testing is important for achieving a system of high quality that meets users' needs as well as business goals.

*References*

Connolly, R., & Hoar, R. (2022). \*Fundamentals of web development\* (3rd ed.). Pearson.

Tsui, F., Karam, O., & Bernal, B. (2018). \*Essentials of software engineering\* (4th ed.). Jones & Bartlett Learning.

*Figures*

Figure 1: Sequence Diagram (sequence\_diagram.png)

Figure 2: Activity Diagram (activity.png)

Figure 3: State Diagram (state\_diagram.png)

Figure 4: Use Case Diagram (Use\_case\_1.png)