Week 2 Assignment: UML Design Modeling

Joshua Murray

CST499: Capstone for Computer Software Technology

Professor Joseph Rangitsch

Due May 26, 2025

# UML Diagrams

## A diagram of a computer AI-generated content may be incorrect.

A diagram of a user registration

AI-generated content may be incorrect.

## Course Registration

## A diagram of a course AI-generated content may be incorrect.

# Testing Methods

Component testing, which is also known as unit testing, is the first level of testing in the software testing hierarchy. It focuses on verifying the smallest units of software in isolation typically individual functions, classes, or modules. The primary goal is to ensure that each component performs as expected, according to its design and logic. This type of testing is generally conducted at the developer level, using automated testing frameworks such as JUnit for Java or PyTest for Python. During testing the developers validate logic, handle edge cases, and test conditions like input validation and exception handling. Since each unit is tested independently of the rest of the system, component testing helps identify issues early in development, making them easier and cheaper to fix.

Usually performed after component testing, the next level of testing aims to examine how multiple components or modules interact when combined. While individual modules may work correctly on their own, integration testing verifies that they function properly together as a group. This includes ensuring that data passed between modules is accurate and that the system behaves as intended across component boundaries. Integration testing is typically performed by developers or testers and can be approached in different ways. This level of testing is especially important in systems where modules heavily depend on one another, such as in an online course enrollment platform where user registration, course availability, and enrollment features must operate seamlessly.

System testing is the a very thorough layer, which evaluates the entire software system as a complete and integrated application. This level of testing ensures that the software meets the functional and non-functional requirements outlined in the Software Requirements Specification. This level of testing is mostly conducted at the QA or testing team level outside of the development team. System testing includes a variety of test types such as functional testing, performance testing, load testing, and security testing. For example, in a student course enrollment system, system testing would validate end-to-end scenarios like registering a user, logging in, enrolling in a course, and being placed on a waitlist when the course is full.

Acceptance testing is the final level of testing before the software is delivered to the client or end users. The primary objective is to determine whether the software system satisfies business requirements and is ready for deployment. This type of testing is often performed by stakeholders such as clients, end users, or a client representative. It validates real-world usage scenarios and ensures the system delivers the intended value to users. Acceptance testing is usually divided into two types: alpha testing, which is conducted in a controlled environment by internal staff, and beta testing, which is performed in a real-world environment by actual users. Only after the system passes this stage is it considered ready for release.

# References

Rumpe, B. (2016). *Modeling with UML* (Vol. 98). Berlin/Heidelberg, Germany: Springer.

Ould, M. A., & Unwin, C. (Eds.). (1986). *Testing in software development*. Cambridge University Press.

Baresi, L., & Pezze, M. (2006). An introduction to software testing. *Electronic Notes in Theoretical Computer Science*, *148*(1), 89-111.