**UML Design**

Tony Carolla

The University of Arizona Global Campus

CST 499 Capstone for Computer Software Technology

Amr Elchouemi

January 23, 2023

**UML Design**

Design and testing are the keys to a successful project. System architects must plan with the customers and stakeholders using mechanisms that convey an understanding of the functionality to people with various viewpoints. Graphical representations can convey many types of information without using complicated language or terminology, and they are valuable for planning and modifying systems. The Universal Modeling Language (UML) is a formalized system of representing many types of functionalities with graphics and symbols (Tsui et al., 2018). UML is perfect for representing functionality to customers and stakeholders.

**Class Diagram**

UML class diagrams are used when developing systems with object-oriented development environments. For example, the MetricuL8 system will utilize object classes including Class, Student, Enrollment, and Waitlist. Class diagrams illustrate the attributes and functions that each class performs (Al-Refai et al., 2019). Figure 1 below, represents the object classes in UML class diagrams.

**Figure 1**

*Class Diagram*

Diagram

Description automatically generated

**Sequence Diagrams**

UML sequence diagrams help illustrate actions taken by users and the systems involved (Tsui et al., 2018). For example, when a student registers, they might forget to fill out the entire form. Figure 2 below, shows the interaction between students and the MetricuL8 system when it happens.

**Figure 2**

*Sequence Diagram*

Diagram

Description automatically generated

**Activity Diagrams**

Some actions involve decision points where the logic will branch based on conditions. The best way to demonstrate this is with a UML activity diagram. For example, when students register for a class, it may be full. The UML activity diagram in Figure 3 below details the interaction.

**Figure 3**

*Activity Diagram*

Diagram

Description automatically generated

**State Diagrams**

When modeling real-world objects using object-oriented environments, it is helpful to illustrate the object attributes as states. For example, in the MetricuL8 system, classes may become full. When discussing this with customers and stakeholders, a UML state diagram is helpful to illustrate object states and the events that cause them to transition between states (Tsui et al., 2018). Figure 4 below shows a UML state diagram for the class object’s full/available state.

**Figure 4**

*State Diagram*

Diagram

Description automatically generated

**Use Case Diagrams**

When detailing the ways users will interact with systems, software architects can use graphics to show the interactions from the user’s standpoint. For example, one way to detail the way a student interacts with the MetricuL8 system when registering is with a UML use case diagram. Figure 5 below shows use cases for students registering online.

**Figure 5**

*Use Case Diagram*

Diagram

Description automatically generated

**Testing**

When creating the MetricuL8 system, we will test it at many levels. The lowest level of testing is at the component level. Software components are the smallest unit of logic and perform one action (Wang & Zhou, 2012). Tests at the component level verify the low-level functionality of the components with test values and scenarios, depending on the type of functionality.

Components work together to provide higher-level functionality. As components are built, they must be tested together so that the higher-level functionality performs as expected (Tsui et al., 2018). Integration testing is performed by testing the functionality of components working together.

The next level of testing is system testing. System testing is performed to verify that functions work end-to-end (Tsui et al., 2018). It is where the system is verified to perform all functionality before turning it over to the customers, although the customers may be involved in the testing. Systems testing for the MetricuL8 system will be done on items such as online student registration and class enrollment.

Once a system release candidate is verified, it should be tested by the customer. Acceptance testing is done by the customer and stakeholders to verify that everything checks out with their expectations. It may be a repeat of the system testing, but the acceptance testing is done by the customers, or it has high visibility.

Conclusion

UML is perfect for representing functionality to customers and stakeholders. It can be used to demonstrate detail, actions, and scenarios in many ways to bring understanding and consensus. Testing at each level of development complements the detailed design with results. Together, design and testing allow a software architect to build a system with end-to-end visibility.

**References**

Al-Refai, M., Ghosh, S., & Cazzola, W. (2019). *Supporting inheritance hierarchy changes in model-based regression test selection. Software & Systems Modeling, 18*(2), 937–958. <https://doi.org/10.1007/s10270-017-0636-3>

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

Wang Haoyu, & Zhou Haili. (2012). *Basic Design Principles in Software Engineering*. 2012 Fourth International Conference on Computational and Information Sciences, Computational and Information Sciences (ICCIS), 2012 Fourth International Conference on, Computational and Information Sciences, International Conference On, 1251–1254. <https://doi.org/10.1109/ICCIS.2012.91>