# CS 255 Model Application Short Paper

Bobby Eigenfeld

robert.eigenfeld@snhu.edu

Southern New Hampshire University

## Process Model Application

Applying a UML process model to the design of the DriverPass system involves using class, component, and deployment diagrams to create a comprehensive and clear representation of the system’s structure and behavior, ensuring it meets the client’s requirements.

Initially, class diagrams will be created to model the static structure of the system. This involves defining the main classes such as User, Appointment, Vehicle, Driver, and TrainingPackage, along with their attributes, methods, and relationships. The class diagrams will establish a clear data model and illustrate how different parts of the system interact with each other. This will help ensure that all necessary data elements and relationships are accurately captured and understood.

Following the class diagrams, component diagrams will be used to show the organization and dependencies of the system’s components. These diagrams will break down the system into manageable parts, such as the user interface, business logic, and data access layers. By detailing the interactions and dependencies between components, these diagrams help in understanding how different parts of the system work together, ensuring a modular and maintainable design.

Finally, deployment diagrams will be developed to illustrate the physical deployment of the system’s components on hardware nodes. These diagrams will show how the system will be distributed across different servers and devices, specifying the hardware requirements and network configurations. This will ensure that the system is designed with a clear understanding of its deployment environment, addressing issues like scalability, performance, and security.

## Object Model Application

Applying an object model to the DriverPass system involves defining key objects, their attributes, behaviors, and interactions. Begin by identifying essential objects such as User, Appointment, Vehicle, Driver, TrainingPackage, and Lesson. For instance, the User object might include attributes like firstName, lastName, and creditCardInfo, with methods such as register, login, and resetPassword. The Appointment object might have attributes like date, time, and driverAssigned, with methods like schedule, cancel, and modify.

Next, establish relationships between these objects. For example, an Appointment object could be linked to one User and one Driver, while a TrainingPackage might include multiple Lessons. These relationships help illustrate how the components of the system interact.

Finally, use UML class diagrams to visually represent these objects and their relationships. This structured approach ensures that the system is well-organized, with clear definitions and interactions, making the development process more efficient and aligned with the client’s needs.

## Process and Object Model Comparison

The DriverPass scenario, each UML model provides distinct advantages. Class diagrams define the system's structure by detailing objects, attributes, methods, and their relationships, offering a clear data blueprint essential for a robust design. Component diagrams help visualize the organization and dependencies of system components, facilitating a modular and maintainable design. Deployment diagrams illustrate the physical distribution of components across hardware, addressing scalability, performance, and security needs. Together, these models offer a comprehensive view of the system from different perspectives, aiding in effective design and implementation.