CS 255 Model Application Short Paper

Kyrstie Jackson

kyrstie.jackson@snhu.edu

Southern New Hampshire University

[**Note:** Complete each section below. Each response should be a few paragraphs in length, and your paper should be between 3 to 5 pages. Use 12-point Times New Roman font, double spacing, and one-inch margins. Be sure to use proper grammar and APA format to cite any sources you use, including the textbook. Remove this note before you submit your paper.]

**Process Model Application**

To apply a process model to the DriverPass scenario, I would utilize a Waterfall approach for initial planning, paired with some Agile flexibility during development. The Waterfall model is beneficial during the early design and requirements phases, especially for clearly outlining the expectations of stakeholders such as DriverPass leadership, instructors, and students. This model breaks the system design down into sequential phases, allowing the team to focus first on gathering requirements, then moving into system design, development, testing, and deployment.

In the DriverPass scenario, this process model helps ensure that all necessary student functionalities—like lesson scheduling, progress tracking, and automated reminders—are captured and defined before development begins. Since the project involves multiple user types (students, instructors, and administrators), a structured and linear model helps prevent scope creep and ensures that stakeholder expectations are met before advancing to the next stage.

However, a strictly linear Waterfall model might not be ideal due to the potential need for feedback and iteration. Therefore, incorporating Agile sprints within each development phase allows for smaller, incremental updates, user testing, and flexibility. For example, building the scheduling feature could happen in one sprint, while building the student dashboard could happen in another, allowing for easier adaptation based on user feedback.

**Object Model Application**

An object model would be extremely helpful in designing the DriverPass system by organizing the system into interacting objects that represent real-world entities. This model focuses on objects like Student, Instructor, Administrator, Lesson, and ProgressReport. Each of these objects would have specific attributes and behaviors that define their role within the system.

For example, a Student object may include attributes such as name, contact information, progress status, and a list of scheduled lessons. It would also include methods like “scheduleLesson()” or “viewProgress()”. The Instructor object could have attributes like availability schedule and a list of assigned students, with methods to “updateAvailability()” or “markProgress()”.

Using this approach provides a clear and modular structure to the system. The object model promotes code reuse and easier maintenance. Each object encapsulates its data and logic, allowing the system to scale and evolve over time. As new features or users are added, the system can be updated by modifying or extending specific objects without needing a complete overhaul.

**Process and Object Model Comparison**

Both the process model and object model bring valuable tools to the design of the DriverPass system. The process model is useful for creating a structured, phase-driven timeline for development. It ensures that the team adheres to a clear plan, tracks progress, and meets deliverables on time. This is particularly helpful for DriverPass since it is a business with specific operational needs, including scheduling and communication improvements. The structured nature of the process model helps prevent costly rework by identifying system needs early.

The object model, on the other hand, excels in managing the complexity of the system through abstraction. By modeling system components as real-world objects, developers can easily understand and manage the relationships between different parts of the system. It supports scalability and maintainability, making it ideal for the growing nature of DriverPass’s student base.

The main disadvantage of the process model is its rigidity—if DriverPass’s needs change mid-project, adapting a Waterfall model can be difficult. Conversely, while the object model promotes flexibility and modularity, it doesn’t address the overall timeline or structure of the development process. Combining both models allows the team to benefit from the structured flow of the process model and the clarity and adaptability of the object model.