# CS 255 Model Application Short Paper

Matthew Rios

Matthew.rios1@snhu.edu

Southern New Hampshire University

## Process Model Application

## In the DriverPass scenario, the process model offers a clear, organized path for designing and developing the new system that will handle driving lessons, online testing, and DMV compliance. It provides a roadmap that guides the project from start to finish. A model such as the Systems Development Life Cycle (SDLC) works well for a project like this because the client’s needs are clearly defined and can be broken down into structured phases.

## The process starts with gathering requirements. This is where Liam, the owner of DriverPass, and his team share what they want the system to do, such as managing user roles, scheduling lessons, and maintaining security. Once these needs are fully understood, the project moves into system design, where those ideas are transformed into technical blueprints. For DriverPass, that includes creating the layout for how students, instructors, and administrators interact with the system.

## Next comes implementation, where the design becomes a real working web-based system that connects scheduling, payment, and online learning features. The testing phase ensures that every function works correctly and that the system performs smoothly and securely. Finally, in the deployment and maintenance phases, the finished system is launched and maintained over time to keep it updated and reliable.

## This process model helps the team stay organized, ensures accountability, and provides checkpoints for quality assurance. Since DriverPass will store sensitive information like payment details and personal data, having these structured steps is essential for keeping the system accurate, secure, and compliant with regulations.

## Object Model Application

The object model takes a different approach by focusing on the real-world elements that make up the DriverPass system, such as students, instructors, cars, and lessons, and how they interact with each other. Each of these elements becomes an object in the design, with specific characteristics and functions. For instance, a Student object might include personal information like name, address, and phone number, along with actions such as register, login, and scheduleLesson. A Lesson object would include details like date, time, instructor, and vehicle, with actions like bookLesson and cancelLesson.

This model makes the system more modular and easier to manage. If DriverPass ever decides to add new lesson packages or update the DMV connection, developers can make changes to specific objects without rebuilding the entire system. That is because the object model uses encapsulation, which means each object controls its own data and functions independently.

It also supports inheritance, so shared features can be reused across multiple objects. For example, a User class might define basic login and password features, while subclasses like Student, Instructor, and Admin each have their own specialized actions. This structure makes the system more flexible and easier to expand in the future.

For DriverPass, the object model mirrors how the business actually operates. It organizes the system in a way that is easy for both developers and stakeholders to understand, creating a more intuitive design that can grow alongside the company’s needs.

## Process and Object Model Comparison

Both models are important for building a system like DriverPass, but they serve different purposes. The process model focuses on how the system is built, including the steps, timelines, and structure. It is especially useful for keeping projects on track and ensuring that each phase is completed before moving forward. This is great for quality control and accountability. However, one drawback is that it is not very flexible. If Liam or his team wanted to make big changes halfway through the project, such as adding new lesson packages, it could cause delays or require extra resources to adjust the plan.

The object model, on the other hand, focuses on what the system is made of. It is more adaptable because it is based on real-world components that can be easily modified or extended. This approach makes future updates, such as adding mobile access or new features, simpler and faster. The main challenge is that it requires skilled developers who can accurately design object relationships. If the model becomes too complex, it can make the system harder to maintain.

In short, the process model provides structure and stability, while the object model offers flexibility and long-term adaptability. For DriverPass, the best approach would be to use both together by starting with a process-driven plan for development and using object-oriented design principles during implementation. This hybrid method ensures that the project stays organized while allowing the system to evolve as the company grows.

## References

Dennis, A., Wixom, B. H., & Roth, R. M. (2021). Systems analysis and design (8th ed.). Wiley.

Shelly, G. B., Cashman, T. J., & Rosenblatt, H. J. (2018). Systems analysis and design (11th ed.). Cengage Learning.