# CS 255 Model Application Short Paper

Bryan Pirrone

bryan.pirrone@snhu.edu

Southern New Hampshire University

## Process Model Application

“Process models are used to graphically represent the functions or processes that capture, manipulate, store, and distribute data between a system and its environment and between components within a system” (Valacich & George, 2019, p. 179). When analyzing the DriverPass interview several data flows become evident regarding how DriverPass should function and could be captured by using a flow diagram. To create this diagram, I need to analyze the DriverPass interview to dissect key information about the system and its functionality.

First, I would start by decomposing the DriverPass interview to define the sources, sinks, and data sources. An effective way to do this would be to determine who the users will be, what their roles are, and how they need to interact with the system. For instance, the customer has several processes that they will need to interact with the system properly. For a customer to interact with the system, they will need to be able to log in, browse packages, view current packages, schedule, change, and cancel driving lessons with an instructor, view course material applicable to their package, and reset a forgotten password.

The above illustrates one process for a particular user of the system. This would need to be performed for all users of the system such as the customers, instructors, IT manager, and Owner. All individuals will need varying levels of access to the system, which will be strictly controlled via an individual's role in the system. The process model can be easily represented by creating a data flow diagram (DFD). If the DFD is generated with consideration to its guidelines, completeness, consistency, timing considerations, iterative nature, and primitive, then the diagram will depict the system clearly for all parties involved in its implementation and development (Valacich & George, 2019, p. 195).

## Object Model Application

An object model visually depicts a system’s objects, actions, and attributes in a clear and concise manner that is understandable (Adkisson, 2021). Object modeling helps system development by identifying its classes and objects to design a system prior to development (Adkisson, 2021). Clear, discernable objects, classes, actions, and attributes in a visual depiction will significantly increase readability and comprehension for developers. To accomplish this, I would utilize a Unified Modeling Language (UML) diagram.

Since users, in all capacities, are the individuals who interact with this system, a user class would be developed first. This parent user class would be generic and contain functions and variables applicable to all user child classes, such as a user ID and a password. The child classes would be representative of more specific user categories such as IT manager, customer, instructor, secretary, and owner. These child classes would be able to inherit characteristics from the parent class that is applicable to them but would also contain specific characteristics only applicable to that child class. This is critical to ensure the separation of user roles within the system. A customer user does not and should not have the same capabilities as the IT manager.

This diagram would help show the interrelations between different components in the system and how data flow is accomplished between them. Each user has a specific role, and that role has specific data flows that would be essential for the system to function correctly for them. The secretary needs to be able to make, change, and cancel appointments but doesn’t necessarily need the administrative controls that would be applicable to the IT manager or owner. Therefore, the secretary's data flow would be limited in comparison to that of the IT manager or owner.

A UML diagram would be beneficial since UML diagrams contribute to better software development practices, enhanced system understanding, and improved collaboration among team members, leading to a more efficient and reliable system. Individual roles within the system can be clearly defined in the diagram to allow for correct system functionality.

## Process and Object Model Comparison

Process models have several advantages for use in the DriverPass scenario. Since process models are used to graphically represent the functions or processes in a system (Valacich & George, 2019, p. 179), many advantages come to mind. This form of modeling allows for visualization of the system’s workflow which allows for greater understanding at all levels of development. The owner, developers, IT manager, or even secretary could look at this model and have a good understanding of how the system operates. Open communication about system design, operation, and functionality is now achievable by this easy-to-understand model.

The disadvantages of a process model for use in the DriverPass scenario are also evident. If the model does not conform to the guidelines described above, then the model may become misleading and provide an inaccurate or incorrect approach to the system’s functionality. This can result in delays in production, exceeding budgets, and significant rework. This could also lead to confusion about system operation and intent.

Object models have several advantages for use in the DriverPass scenario. Since object models visually depict a system’s objects, actions, and attributes in a clear and concise manner and help by identifying the system’s classes and objects required to design the system (Adkisson, 2021), system development becomes easier to understand and implement. Object models also excel in this scenario because they help identify the relationships between a system’s objects and attributes. This will allow for the integration of some object-oriented programming principles such as encapsulation, inheritance, and polymorphism into the system’s development. This will help promote code maintainability and reuse which will lead to increased scalability of the system.

There are some disadvantages to using an object model for the DriverPass scenario. The object model could become complex which could lead to confusion amongst team members. Not all interested party members may have the level of knowledge required to dissect and understand this kind of model. If the model gets too complex, it can also lead to undesired effects due to the improper integration and utilization of object-oriented principles. This can lead to issues with debugging, which would ultimately affect the timeline and the cost of developing the system. This form of modeling requires a significant time investment for proper utilization and implementation.

## References

Valacich, J. S., & George, J. F. (2019). Modern Systems Analysis and Design (9th ed.).

Pearson Education (US). https://mbsdirect.vitalsource.com/books/9780135172827

Adkisson, H. (2021, October 5). Object Modeling for Designers: An

Introduction. Https://Hpadkisson.medium.com/Object-Modeling-For-Designers-An-Introduction-7871bdcf8baf