# CS 255 Model Application Short Paper

Pasco Logan

Pasco.logan@snhu.edu

Southern New Hampshire University

## Process Model Application

[How would you apply a process model to a design for the DriverPass scenario? Remember, you do **not** need to create diagrams for this paper.]

In order to develop a process model for the DriverPass project, we would first need to identify the "types" of each process involved in the development of the software, the maintenance of the database, and the deployment of the service in the cloud. This would be a high-level plan that emphasizes the flow and storage of data rather than specific features or components. We would emphasize the sequential functionality using this method so that the resulting flowchart could be used for business analysis and to establish what other kinds of preparation or planning are needed to finish the program.

To create Driver Pass, the resulting flowchart first considers the needs of the customer, then applies a process model. Let's say I want to modify certain account settings, such as a password or the number of users. To begin, I would ascertain which type of user is needed (secretary, IT administrator, etc.). The procedure would begin with the client, move on to the initial access and login stages, and then branch off to include a password reset if the user has forgotten their credentials. Following this, the user can make appointments, edit their contact information, and view their current status.

Since our system is comprised of numerous processes, we may either model the entire system as a single continuous process or describe each process independently. In my opinion, the best way to understand the system as a whole is to create several different models of the various processes involved. Drawing a flow diagram, which depicts visually how one stage in the process is supposed to lead naturally to the next, is the best way to apply a process model to our DriverPass system. A flow diagram can be used to branch off of a process when steps beyond a specific point must be performed before moving on (employee intervention, a necessary update in information, a required training step, etc.).

## Object Model Application

[How would you apply an object model to a design for the DriverPass scenario? Remember, you do **not** need to create diagrams for this paper.]

The object model for DriverPass would be built around the language and software features of the system. Documentation of an object model system may take the form of state machines, collaboration diagrams, diagrams depicting the relationships between objects and classes, etc. This would be the diagram detailing the required objects/classes, the variables and variable types contained within them, as well as the methods used by those objects/classes and their relationships to one another.

Using a UML diagram, I would implement an object model for the DriverPass use case. To begin, I'd create a generic "user" class from which all other classes would inherit. Users would log in with a unique username and password, and would have varying permissions and duties depending on their user type. Aggregation would be used, and each user type class would be built of a number of different classes. A customer user, for instance, might be made up of a contact record object that the customer can edit, and a training object that displays the client's current training status. There wouldn't be any user objects that make up the secretary's profile, but they would have access to features like appointment scheduling and contact record updates. The IT administrator, like the secretary, would not be made up of any physical things, but would have even more power over user accounts than the secretary does now. Data can be downloaded and the owner will have the same level of access as our IT administrator. A UML diagram is the ideal option here since it provides a graphical representation of the capabilities of different user roles and a road map for how the system should be built by the programmers. The development team would be able to put in place the required controls for the system to function as intended with the help of the UML diagram that lays out the responsibilities of each user type.

## Process and Object Model Comparison

[What are the advantages of each model for the DriverPass scenario? What are the disadvantages of each model for the DriverPass scenario?]

Since the DriverPass process model requires less technical language than the object model, our client can better understand the system architecture. The process model shows how each process in the system works from beginning to end. The DriverPass process model explains how each process is meant to work, but it doesn't reveal the system's technical details. The process model doesn't show where to place classes and methods in the system's hierarchy. DriverPass' object model displays the system's technological aspect. It defines system classes and methods (what actions or functions they contain) The object model helps the team construct the system and deliver an executable to the client. The object model provides a technical roadmap for system development, but presenting it to a client without the technical skills could be frustrating.