# CS 255 Model Application Short Paper

Cristian Alvarez

Cristian.Alvarez@snhu.edu

Southern New Hampshire University

## Process Model Application

A process model for the DriverPass application would start with Users attempting to login to the online system, with them being successful or requiring a password reset or account creation. When using process models its better to not make then too complex so they can describe the whole system without going into insane amounts of details so let’s get a bit less technical.

Users are either customers, administrators or employees. Customers can create, view, modify and delete reservations, they can access online courses from their account, pay for services/packages and update their contact information. So, depending on the users’ inputs, which generally would start with creating a reservation, then contact information, then payment and finally confirmation. If a user does not want to make an appointment, then they can access online courses. Functionally they have three decisions to make in regard to appointments, courses or account information and finally log off.

Administrators process flow begins with login, then from there they can choose if they want to view logs, manipulate or view reservations, update course work, reset customer account information, and log off.

Basically, we have a process model that would have branching decisions that start with login, then the use of certain processes and finally log off for all types of users. Personally, I believe each type of user should have their own process model since they use different parts of the system. A flow diagram allows for an accurate but a less precise explanation of what is happening, what decisions lead to branches or other processes. This is perfect for this type of system since it relies on users interacting with the system to make things occur and users have multiple options on what they can do and its not very complex to begin with.

## Object Model Application

An object model essentially starts with a UML diagram. An object model shows how objects interact, inherit, associations and how data potentially flows. I would begin with an abstract user class that has attributes for First Name, Last Name, Email, Username, Password with methods to manipulate these attributes. The Customer class would inherit from the User class, and I would include attributes for address, phone number, role, and API payment process integration and methods to modify address and phone number.

Admins would also inherit from the User class and would include attribute for roles and methods for modifying course information, other users’ profiles and processing payment refunds. A course class would need to be implemented with an aggregation towards users many to many since many users may be enrolled in many classes. Same with an appointment class with an aggregation of many to many since many users can make many appointments. These classes would have their respective modifiers for their attributes such as packages, payment confirmation and completion data.

UML diagrams are great for understanding how the components of a system are defined and interact with each other. UML is prefect for this project considering there are many interactions among classes.

## Process and Object Model Comparison

The Process Model’s greatest strength is the fact that they are very simple, easy to understand and give a good overview of how a user may interact with a system. However, these models tend to group processes together and that ends up hiding interactions and encapsulating data that may be necessary to the proper deign and implementation of said system. Object models are complex detailed oriented models that allows how the classes should be implemented and how they communicate/manipulate data amongst each other. Object models are a technical document that is very difficult to understand if you do not know what you’re looking at. Essentially object models are great for developers but horrible for customers since it shows them information, they don’t need nor know how to interpret.

Process models are great for customers because they can see how the system may be used and what decisions can be made with said system, but they are horrible for developers because they lack critical information about how the system is implemented. They both have their place within system analysis/design and should be used together to get a high/low scope explanation that can be used for a variety of people on the team and customers.