# CS 255 Model Application Short Paper

Tyler Coplan

tyler.coplan@snhu.edu

Southern New Hampshire University

October 1, 2021

## Process Model Application

A process model outlines the processes involved in a system and how information moves throughout the processes. Data flow diagrams (DFD) are an excellent way to represent a process model. A DFD shows what sources provide information to the system and each step the data takes until it exits the system. The data must be either stored or transformed in some way during each process within the system. The model is not concerned with the implementation of each process, it just shows what actions are taken as data moves along the system.

To create a DFD of the DriverPass system, an analyst would need to consider the different sources of information that enters the system. Customers and employees are sources that can make requests for information from the system or make changes to the information within the system. A customer can create a log-in, book appointments and take tests in the system. These processes would have data flow from the customer, through a process in the system, then stored in a data store within the system. A process in the system can then access that data and send it to another process or a sink, which could be the customer or an employee. The owner of the DriverPass system can make requests for reports from the system, update or revoke employee access and update regulations from the DMV. All the different use cases of the system would be linked to processes in the system where a source starts the information chain, the data runs through the system’s processes and data stores, and leaves the system to a sink, which may or may not be the same as the source.

## Object Model Application

Object models are visual representations of how data is structured and related to other data objects within a system. The unified modeling language (UML) is a well-known tool for creating object models. Data objects such as customer, owner, appointment, exam, etc. are represented on a chart. The primitive data that makes up the object, such as a string for name or an integer for exam score, are shown within the data object. Actions that can be taken on the object such as book appointment or update address are also shown within the data object. Each object has links to other objects that can show how they are all related to each other. For instance, a person class can be a superclass from which customers and the owner can inherit from and contains any relevant data that all people would have. An appointment object can have an aggregate association with a customer, meaning that without the customer there is no appointment. These relationships help identify data dependencies within the system.

An object model the DriverPass system would need certain key classes of data. A person superclass would help build subclasses of owner, customer, employee, and driver. An appointment class would be needed to handle appointment scheduling, and would require a customer, a driver, and a time. A schedule class could be made up of all scheduled appointments and be used to show a calendar for all booked appointments. An exam class is needed for students to take exams online. The exams can hold questions, grade each exam, and show the students their grades. The owner class would need administrative privileges that can handle the backend of the system as well as generate reports about the scheduling. A transaction class would be needed as well to handle customer purchases of driving packages. The transaction could update the customer’s number of driving lessons available, set flags for whether the student has online exam access and produce receipts for both the owner and the customer. Object models help visualize how the data in a system is organized and related to each other.

## Process and Object Model Comparison

A process model is great for understanding how the system works. Process models give brief descriptions (such as a couple of words) of the different information sources and sinks, the processes the data goes through and at what point data gets stored within the system. It is much easier to understand than a written description of the process. There can be multiple levels of process models as well, so each process in a system can have its own process model for just that process. This allows analysts to see the whole system diagrammed out and then refer to other models if they need to understand specific processes in greater detail. Two key components of a system are not addressed in process models. Process models are not concerned with timing of events and they are not concerned with how the data is structured. It is irrelevant to the model if information is requested once a day or once a year. It simply shows who or what is requesting the information and what processes it needs to go through in the system. The model also just shows a brief description of the data, such as customer order, without showing the underlying structure of the data.

Object models are great at showing how data is structured and related within a system. An object model shows a static view of the system and the different links that the data has with each other within the system. Object models are excellent for showing what kind of data is required for each object, such as email addresses or phone numbers, and what the primitive data type is for storing this information. Object models are helpful for understanding how a database should be organized to store the data generated within the system. Object models are also good for showing data dependencies within a system. If a data object is aggregated from another data object, then it cannot exist without that data. The DriverPass system would have an appointment data object that is aggregated from a customer and a driver. The appointment could not exist without either, so deleting the customer or the driver also deletes the appointment due to the data’s relationship. Object models also show the cardinality of objects. An example of this is a customer can have zero to many appointments, but each appointment can only have one customer and one driver. Understanding these types of relationships helps analysts organize the system’s code and database.

Object models do have limitations. They are not concerned with any process involved in execution of the system. It does not show dynamic data either. Object models only give a static view of data in all the states it could exist in. Object models are not good for understanding how data is used within the system or how it is changed. Behavioral models show how the data within each object is changed in a system, and process models show the order of processes within the system. These different models can be used together to create a complete picture of a working system. However, the entire story cannot be told by one type of model. There is good reason for this, too. It is much easier to understand system processes without being concerned with the underlying data or implementation. It is easier to understand the data structure of the system when seen statically. Each type of model has a purpose and is useful for when an analyst needs to understand the part of the system that the model is intended to portray.