# CS 255 Model Application Short Paper

Christian Clark

Christian.Clark5@shnu.edu

Southern New Hampshire Universit

## Process Model Application

A process model is a way to describe how a system or program flows from one process to another. A process model is a graphical representation of a process that includes all step involved, the order in which the occur, and the relationship between them (Vation Ventures, 2024). To that end, describing how the DriverPass system would work would be a little complex as there are a few different branching paths.

The first process would be logging into the system. This is where both the customers and employees would start their entry into the system, and thus, into the process model. The login process would happen, and depending on if the user is a customer, employee, IT admin, or owner, it would branch to different processes to show different information. A customer would be able to see their information and would have another process to change their information if desired. An employee would be able to create new accounts and make appointments for customers. An IT admin would be able to see the information of employees, and the owner would be able to see everything.

When branching to the different processes, those processes would converge into main database to be able to show its relevant information. Customers would be able to see their test scores, personal information, and appointment times. Employees would be able to input new information so they can create new accounts for new customers that call in. IT admins and the owner would be able to see everything so they can change information on employees or change passwords for whoever needs it. There will also be a database of the drivers being hired that the employees can choose from when making appointments.

While this process model is pretty comprehensive and does show how the flow goes from log-in to performing whatever task the user needs, this process model is also very shallow, as it’s a small number of simple processes.

## Object Model Application

On the other hand, an object model would be a much better way to describe what the system is and the relation between the different objects of the system. Since there will be a much larger number of objects, it will be difficult to comprehensively describe what objects there will be and how the interact without a model itself.

We could create a user class that is the parent of all other types of users, those being customers, employees, IT admins, and the owner. These would all have different functions and methods that will work throughout the system. From there we can branch out some objects that may be inside any of the variations of user objects.

A large majority of the data that a customer would have would be directly inside the database we’ll use to store all the data for our users. To access that, we would have methods inside each user variation so they can read from the database to display to whoever is logged in as that user. There will also be variables that the interface can read from so that it can display that information directly to the user on the screen.

Employees would have their own functions to be able to create new accounts and appointments for those who call over the phone. This will be similar to how the IT admins and the owner will be able to see all data and change passwords or block employee accounts from accessing the system.

There could be an appointment object that an employee would use to create that appointment and put it into the database. This would reference a certain customer in the database, but the object would be attached to the employee making the appointment. This appointment object would have all the methods required to input information into the object so that when it’s submitted, all that data can be put directly into whatever database is used to keep track of the appointments.

The interface might also be considered an object. It would be able to take from the current logged-in user and display the relevant information to that user. It would change form depending on which type of user is logged in as well. We could display the interface object as a parent of the different types of interfaces for the different types of users. This would also keep the interface safe from unwanted tampering as customers wouldn’t even have the chance to access the admin panel or employee panel.

## Process and Object Model Comparison

Comparing the two models is important, as we can see the differences they produce. Our process model shows us how data flows between the different processes. It also shows us where the entry and ending points of the system are. It shows us where branching points lie, how we get to those branching points, and what that means for our data flow.

Unfortunately, a process model doesn’t show how those processes work. A process model really only shows the big picture, but not how data is transformed when going through its travel. We’re unable to see the objects that work in those processes. A process model essentially shows us a bunch of linked black boxes that take in data and transfer that between other processes.

The object model shows us those specifics. It shows us the exact relations between each object and how those objects function. What it doesn’t do, however, is tell us when or why said objects are used. That is its strength, the ability to describe how a series of objects works, without the flow. It gives us the raw view of the inner function of the system without any external influence. We want to know how a system works in an object model.

But that’s also a big downside to an object model as well. We see how the objects work and the relationships between them, but we don’t know when or where they’re used. That’s why we need both models working together to create an overview of the system. Each strength complements the other’s weakness.

# Works Cited

Vation Ventures. (2024). *Process Model: Definition, Explanation, and Use Cases*. Retrieved from vationventures.com: https://www.vationventures.com/glossary/process-model-definition-explanation-and-use-cases