Tyler Thomas

CS-255 System Analysis and Design

Southern New Hampshire University

April 5th, 2020

Model Application

In order to support the design of the Driver pass system, we can develop some very useful applications in order to visualize and organize how the system should work based on the requirements. First, we can consider the process model. For this we can design a Data Flow Diagram of DFD. Using a DFD we can outline the different processes of the system, the data stores, and sources. One of the main processes of the system is going to be the scheduling of appointments within the driver pass system.

A customer will interact with the system with process 1.0 which will be scheduling an appointment date & time. This process will then use process 2.0 to validate if the date and time is available based on the number of drivers on that date & time. Once validated the schedule process can then notify the customer with a confirmation and send a report to administration (secretary & IT admin). Another process within the system will be tracking user progress on based on the driving package they purchased, this will be requiring sending data to the management source and back to the customer so they can check their progress. It will give them update such not started, in progress, and completed on their respective examinations and trainings.

Finally, we can consider the process for creating a user, it will be requiring collecting information from the user and storing that information in the authentication server. This will provide the necessary security for the system ensuring users are being validated upon login. Overall, we have outlined the process of creating a user, scheduling appointments, and tracking user progress.

In order to implement and object model, we will want to break the design into classes, objects, and attributes. For classes we will want to use a few main classes for the different parts of the system in order to create the necessary objects. We will want a base package class that will create all the package objects offered by Driver Pass. Some attributes would packageName, packageId, price, and content. Then from this we could derive an object for each of the different packages that will be offered to customers.

Next, we would need a User class, the user class will have attributes such as name, & name. From this we could derive the different user objects that will inherit all the User Class attributes. There could be a customer object, which will have attributes such as name, phone number, email, address. The admin object will just have a username attribute for authentication, and then we have a secretary object as well.

Another class we would need is an appointment class, which would have attributes such as name, driver name, date & time, address, and appoint number. This could create appointment objects which are assigned to the customer upon schedule, this will allow the customer and the system to have a record of the appointments made in the system. This would also allow for reporting for the admin.

If we look at the advantages of using the process model, it provides a high-level overview of how data will flow throughout the system. It helps the designers analyze how the system will interact throughout the process of scheduling appointments, tracking information, storing information, and where information is coming from. Some disadvantages to process modeling are that it does not really aid in coding of the design, a developer could not design an application based on a DFD. Also, if not done correctly, it could slow down the systems design process, if multiples DFD’s are made and say one does not identify a process from another it could mess with the design of the system due to poor documentation.

Looking at object-models, they are most advantageous in that they can be turned from design to code. Object models break down each process, and display their attributes, and show the relationship between the different processes in the form of object-oriented design. These designs help reduce redundancy in code, it modular in design, and helps with security by using encapsulation, and polymorphism. Some disadvantages of object models are that object-oriented programming is a difficult concept with a steep learning curve. It requires the use of object oriented languages, and frameworks, and can become overly complex in a large enough system with lots of libraries.