Project One: Model Application Short Paper

Ryan Hubbuck

CS 255

Prof. Washington

02/10/2024

Process Model Application

Process modeling involves graphically representing the processes that capture, manipulate, store, and distribute data between a system and other components within a system. During the analysis phase of systems development, process modeling helps organize information into a representation of the system that currently exists, and the requirements desired in the new system. A process model is one of three major views of an information system which are process, logic, and data models. These tools outline the specifications of an information system.

A Data Flow Diagram could be created for the DriverPass system to better describe the flow of information through the system. Throughout this process, the key attributes to keep in mind will be completeness, consistency, timing considerations, and iteration. The overall DriverPass system must first be broken down into several processes. First, data sources need to be identified. The data sources of the DriverPass system are the students/users, secretaries, drivers, and IT admin. Each of these data sources send information to or receive transformed information from processes.

The processes which comprise the DriverPass system are creating accounts, logging in, updating user information, downloading reports, choosing a package, studying course materials, taking practice exams, contacting support, contacting students, making payments, making reservations, modifying reservations, deleting reservations, reviewing DMV guidelines, and recording driving lesson notes. Each of these processes can be defined by where information is received from and where the data travels to. Beginning with a focus on the student/user as the data source or actor, we would need to create branches for each of the processes the user can complete. Creating an account would be a branch that can lead to logging in, but the user can also go directly to logging in if they already have an account. From there, data can branch in many directions such as creating, modifying, or deleting reservations, studying materials, taking practice exams, reviewing DMV guidelines, choosing a package, or contacting IT support. Creating, modifying, or deleting reservations can also receive data from secretaries and send data to the driver instructors. Admin will receive data from user contact, password reset, and system errors. Drivers will send data to the process of leaving notes during lessons. This data is then sent to the users for review.

Object Model Application

An object model visually represents a system’s objects, actions, and associated attributes. An object has a well-defined role in the application and has data, behavior, and identity characteristics. Object models, therefore, are important in defining the various objects within a system and how they interact. Essentially, object models use object-oriented programming to form a model. To create an object model of the DriverPass system, we will need to define the classes that are part of the system and the relationships between these classes. Each class will contain attributes to store information and methods to access or transform information.

The classes required to create the DriverPass system will be User, Student, Driver, Administrator, Secretary, Reservation, Package, Driving Session, and Course. We will have the User superclass that contains attributes such as a user ID, password, date of creation, and log-in status. Methods available to this class will be log in verification, logging out, and updating account details. The Student, Driver, Admin, and Secretary classes will inherit from User, which means they will also have the attributes and methods available to Users. In addition, Drivers will have attributes for associating themselves with a vehicle, logging time worked, downloading reports, and notes taken on specific students. Drivers will also need methods to access reservations, view and contact students, and log notes. Secretaries will need methods to create, modify, and delete reservations. Admins will possess methods allowing them to access the database of course materials, to modify reservations, and to modify or unlock accounts. Finally, students shall require methods allowing them to access course material, take practice exams, provide payment information, select packages, and create, modify, or delete reservations.

In addition to the classes associated with user types, the Reservation class shall contain attributes to store a reservation type, duration, time of reservation, associated driver, and associated student. Next, a Package will need attributes to store cost and included materials. Lastly, a Course will contain information regarding a course name, course duration, the learning modules which comprise the course, a practice exam, and methods to allow students to access material and record results to the database. Regarding class associations, a Reservation is a Composition of a Student, which means that Reservations cannot exist without Students. Furthermore, a course is a composition of a Package because each learning module belongs to a package purchased by a student.

Process and Object Model Comparison

Object models differ from process models by focusing more on the objects and classes that comprise a system. This focus on the elements that compile to create a system does include some information about relationships between elements. Still, there is not as much detail on data flow as present in a process model. Process models focus more on the processes that take place to make the system function and go in-depth on how data is transferred throughout the system, while object models focus more on the overall structure of the system and how various elements relate to one another.

Process models are most effective in describing data flow and the basic way a system will operate. It provides a high-level view of what a system should do. Process models do not provide much insight into where data will specifically be stored and specific methods that will transform said data. A primary advantage of a process model in relation to the DriverPass system is that it would likely be most effective in portraying the overall functionality of the system. We can clearly observe who inputs information into the system and who receives that data. This helps us visualize the functionality of the application clearly. On the other hand, a disadvantage of the process model is that it is lacking a level the level of detail required to code aspects of the system. While we know how data flows, we cannot accurately assess which objects need to be created or specific variables or functions within the system. A process model would be most effective as a high-level overview.

While object models may not specifically name and lay out the overarching ‘processes’ within the system, they do show how data will be shared or transferred between classes. The details describing attributes and methods make it much easier to understand how objects within the system will be implemented by software developers. A key advantage of object models is that developers can easily see which methods or attributes should be private to the specific object and which can be public. This can give us a better understanding of which modules of the system can interact with other parts. Additionally, the options for representing relationships between objects are also useful in providing insight into how the classes will be coded. This can be related to object-oriented programming, as encapsulation is the idea of keeping data segregated from the rest of the system, inheritance is the sharing of attributes from parent classes, polymorphism is the reusing of methods by associated classes, and abstraction is the hiding of complexity from the outside system. The primary disadvantage is that it may be difficult to visualize the ‘big picture’ from an object model. While it is clear to see specific methods, it may not be clear the chain of events that lead to system functionality. The overall goal and key features of the system are more difficult to understand from an object model alone.