CS 255 Model Application Short Paper

[Tyler Logue]

[tyler.logue@snhu.edu]

Southern New Hampshire University

**Process Model Application**

I would apply a process model to design the DriverPass application that would first begin with requirement analysis and gathering. The main objective of requirement analysis/gathering is to understand and document DriverPass needs thoroughly. I could perform this by collaborating with the product owner, stakeholders, and customers to gain insight into the systems' functionalities. With techniques of meetings, surveys, and workshops, it will be possible to collect functional and non-functional requirements. Next in the process model, I would be system planning and design due to the fact of having most/all of the functional and non-functional requirements of the DriverPass system. Its objective is to create a blueprint for the system's architecture and functionalities. Also, utilizing tools like Gantt charts, Use Case Diagrams, and Data Flow Diagrams (DFD) clarifies system components and their interactions, offering a visual roadmap for DriverPass development.

We would further continue building our process model through iterative development with a methodology such as Scrum. Scrum is an Agile framework that is flexible and adaptable for software development. This is essential to break down the project into smaller and manageable parts for the development team to focus on incremental development within time limits. Testing of the system will begin after and during sprints to ensure system functionality and reliability. After the system is rigorously tested, deployment of the system can occur to sell or access the product DriverPass. During deployment, we will need to monitor and perform maintenance to keep the system stable and patches/updates as needed.

**Object Model Application**

I would apply an object model to a design for the DriverPass scenario by first identifying objects within the DriverPass system. For our system, the identified objects such as User, Lesson, Car, Trainer, Training Package, and Reservation all play pivotal roles in facilitating the core functionalities and interactions within the Driverpass system. Each object would be defined as a class in the model containing attributes that characterize each class. Also, we would need to identify relationships/ cardinality, and this helps to show how classes interact within the system. After we would want to define methods of each class to represent the behavior performed by specific objects. Lastly, we would want to encapsulate data and methods within classes, to ensure that the DriverPass system is efficient, manageable, and secure.

**Process and Object Model Comparison**

The advantages of the process model for the DriverPass scenario is that the model is structured, visual, flexible, and adaptable. This is important when developing software for a system such as DriverPass because requirements may evolve from customer or owner requests. It also makes it easier to track development progress and collaboration to help better understand the process clearly. Transparency is there as team members should be aware of the processes that exist and why. However, it does have its disadvantages of needing team collaboration frequently to further project development. Also, process models from what I have studied work best for small to medium size projects.

The advantages of the object model for the DriverPass scenario is to break down the DriverPass system into smaller parts. It clearly represents the structure of the system by use of objects, classes, attributes, and how they interact. Encapsulation also occurs within the model to enhance data integrity and secure information that is needed. The object models' disadvantages are that it can be time-consuming to create a proper object model. As well it is not adaptable to adjustments where requirements evolve compared to the process model.

**References**