CS 255 Model Application Short Paper

Joshua Wozny

joshua.wozny@snhu.edu

Southern New Hampshire University

Process Model Application

The process model describes the flow of data through the system and the discrete processes that exist within the system. Valacich, J. and George, J. (2019) say that “Process modeling involves graphically representing the functions, or processes that capture, manipulate, store, and distribute data between a system and its environment and between components within a system” (p. 179).

The DriverPass system process model defines sources, sinks, data flows, processes, and data stores. In this system sources include Student, Administartor, and Driver. Each of these act as sinks within our system as well. Several data stores are required for the system to function and include Registered Students, Driver Availability, Course Content, and Instruction Packages. The data flows through the system between sources, sinks, and data stores as managed by the various processes:

1. Register New Student
2. Select Package
3. Process Payment
4. Login
5. Display Landing Page
6. View Content
7. Take Practice Test
8. Score Practice Test
9. Retreive Student Records
10. Update Student Records
11. View Driver Availability

12. Reserve Driver Instruction
13. Schedule Driver
14. Update Packages
15. Log Instruction Time
16. Log Driver Notes
17. View DMV updates

Data flows for the DriverPass system can be divided into three primary types: online content, system administration, and driver instruction. The process model describes in detail how these dataflows travel through the system.

Object Model Application

The object model describes the relationship between objects within a system. Objects are single instances of classes with properties and attributes with values known as state. This state cannot be changed from outside the object but can only be accessed through the object's behaviors which describe how an object acts and reacts, expressed through its methods. An operation occurs when one object acts on another to get a response (Valacich, J. S., & George, J. F., 2019).

The system object model defines the classes, attributes, methods, and relationships between classes. The DriverPass class diagram is summarized n the table below.

Class Name	**Methods**	**Relationships/Comments**
User	login(), retrieveStudentRecord(), accessLevel(), viewContent(), viewCurrentScore()	Parent of Administrator, Driver, and Student

Class Name	**Methods**	**Relationships/Comments**
Driver	updateAvailability(), viewCalendar(), logInstructionTime(), logNotes()	Driver has an aggregate relationship with Student - one driver can have multiple students Child of User
Student	register(), takeTest()	Child of User
Administrator	updateStudentRecord(), updateReservation(), updateContent(), viewDMVUpdates()	Parent of Super User
Super User	updatePackages(), createUser(), viewLogs(), viewStatistics()	Child of Administrator
Content	getContent(), isContentCompleted()	Composite relationship with Student, content cannot exist without an associated student Parent of Test
Test	scoreTest(), getScore(), isRetestAvailable()	Child of Content
Package	getPackageDescription(), getPackagePrice(), availableDriverInstrSessions()	Composite relationship with Student, a package cannot exist without its associated student.
Driver Log	getDriverLog()	Composite relationship with Driver, a driver log cannot exist without its associated driver.
DMV Repo	viewCurrentDriverManual(), viewDMVTestingLocations(), viewDMVTetingCalendar(), hasUpdatedMaterial()	A repository class to access the DMV API to access current DMV materials and track their current version

Process and Object Model Comparison

The process model provides a high-level view but is more difficult to design and implement concrete system components. Some questions that are not well defined in a process model include:

- What component of the system is responsible for a particular process?
- What is the desired state of the system before, during, and after a process is executed?
- Does a process have a local or global state?

The object model simplifies the development of the business logic by using object instances to manage state and functionality. However, object models have a more limited view of how each process flows through the system, and the core functionality is less visible. Each class must be reviewed to determine what functionalities exist, and the order that they occur cannot be immediately discerned.

The strengths of each type of model complement the weaknesses of the other. Using both allows for a more complete description of the system being analyzed and will result in a more complete set of system requirements.

References

Valacich, J. S., & George, J. F. (2019). Modern Systems Analysis and Design (9th ed.). Pearson Education (US). https://mbsdirect.vitalsource.com/books/9780135172827