Thomas Tran

SNHU CS-255: System Analysis and Design

Professor Denise Washington

04/06/2025

**Module 4-2 Assignment: Evaluate an Object Model**

**Comparison of the Object Model and Process Model for Hamp Crafts' Online Storefront**

**Object Model Interpretation**

**What are the different functions of the online storefront? How are they represented in this type of model?**

The object model diagram displays 12 functions of the online storefront. These functions are represented as methods (operations) inside different classes liker Customer, User, ShoppingCart, OrderDetails, ShippingInfo, and Administrator, and they represent what actions each part of the system can perform

Functions represented:

* Customer Account Management: register(), login(), updateProfile()
* User Authentication: verifyLogin()
* Shopping Cart Actions: addCartItem(), updateQuantity(), viewCartDetails(), checkOut()
* Order Processing: placeOrder() in Order, calcPrice()
* Shipping Details: updateShippingInfo()
* Administrative Functions: updateCatalog()

.

**What are the different classes of “users” represented by this object model? What are the associations between these classes?**

There are three “Users” classes represented by the object model:

* User: general class holding common attributes such as userId, password, etc.
* Customer: contains customer-specific details and actions
* Administrator: hold responsible for managing product catalogs

Both the Customer and Administrator are also inherited from the User class.

The Associations between each class are represented below:

* + Customer and Administrator inherit attributes and function from User.
  + Customer is linked to Order and ShoppingCart association of (1 to 1 or 1 to many) relationships
  + Order is linked to OrderDetails and ShippingInfo association of (1-to-1) relationships

**How would the objects “use” their respective variables and functions? Does this object model capture all of Hamp Crafts’ desired functionality? Why or why not?**

Each object class in the model uses its attributes (variables) and methods (functions) to represent real-world activities. Each object manages its data through attributes and performs tasks through methods:

* **ShoppingCart**: uses cartId, productId, etc., and methods like addCartItem() to manage the items in the cart.
* **Customer**: uses register(), login(), and updateProfile() to manage their account profile.
* **OrderDetails**: calculates prices using calcPrice() based on unitCost and quantity.
* **Administrator**: use updateCatalog() to update the menu catalogs.
* **Order**: placeOrder() finalizes the purchase and link to ShippingInfo.
* **ShippingInfo**: Updateshippinginfo() to display the detail of the purchase and shipment.

The object model covers core functions like customer management, order processing, and catalog updates. However, it lacks features such as product browsing, payment gateway integration, and supplier relationship management, which are important for a complete online storefront.

**The above diagram uses a solid diamond shape to represent a form of aggregation. What type of aggregation does this represent? What does it imply about the relationship between the classes? Why is a solid diamond the appropriate choice here?**

In UML, a Solid Diamond represents composition aggregation, indicating that the "part" cannot exist independently of the "whole." This means there is a tightly bound lifecycle between the two: if the “whole” is deleted, the “part” is also removed. For example, if an “Order” class is composed of multiple “OrderDetails”, those order details cannot exist on their own, they are fully dependent on the Order. This is appropriate in the Hamp Crafts model because an OrderDetail only makes sense within the context of a specific order. This relationship supports data integrity and ensures proper object scoping within the system design.

**Compare Process Model vs. Object Model**

The process model is effective for illustrating sequences and workflows, such as how a customer adds items to a cart, checks out, and receives confirmation. And it is especially helpful for identifying user interactions and system responses. Its visual clarity makes it easier for non-technical stakeholders to understand system behavior. However, it doesn’t show how data is structured or how different components relate, and it can become difficult to manage as system complexity grows.

In contrast, the object model provides a clear view of the system’s structure, detailing data relationships and supporting modular, reusable design, making it well-suited for planning code architecture. Its limitation lies in not showing the order or timing of operations and making it harder to visualize the full user experience. Both models are essential for Hamp Crafts’ system design: process models bring clarity to workflows and external interactions, while object models help define system architecture and functionality. Together, they offer a comprehensive understanding of the system.

To strengthen the current UML class diagram, I think it would be beneficial to add a Notification class, an interface for an external PaymentProcessor, and methods like trackOrder() to fully capture the desired features of the online storefront.