Matthew Muller

Hamp Crafts Object Model

There are a number of different functions of the online storefront shown in this model, all represented as public functions within the different classes in the UML class diagram. The different functions include: addCartItem(), updateQuantity(), viewCartDetails(), checkOut(), register(), login(), updateProfile(), placeOrder(), verifyLogin(), updateShippingInfo(), updateCatalog(), and calcPrice(). There are two different classes of users represented in this object model, the Customer class and the Administrator class. These two classes have an inheritance relationship with the User class, inheriting its elements and adding some of their own (accountBalance, for example).

These three classes use their respective variables and functions to store user information such as shipping and billing information for the customers and adminName for administrators, as well as an ID, password, registration date, and login status for all users. The Customer class' functions allow for the registration, updating, and log in of profiles while the Administrator class' function allows for updating the catalog. All users' log ins are verified by a function in the User class. The shopping cart uses its variables and functions to keep track of items added to the cart, including how many are added and when they are added. The functions allow for adding items, updating their quantity, and allowing users to view their cart and check out. The Order class stores each order's ID, as well as the customer and shipping information for the order. Its function allows for the placing of an order. The Shipping Info class keeps track of a shipment's ID, type, cost, and region. It also has a function that allows for the updating of shipping info. Finally, the Order Details class keeps track of an order's ID, as well as the quantity, price, and ID of each product in the order. It also has a function that calculates the price of an order.

This object model captures most, but not all, of Hamp Crafts' desired functionality. One way that it could be improved would be for the Administrator class to have functions that allow for customer support and the updating of customer information. Also, Hamp Crafts stated that they wanted both themself and the customer to receive a notification when a transaction is completed and this is not present in the model.

The solid diamond shape used in this model represents composition. This is appropriate because it implies that some of the classes "contain" other classes. For example, each customer object contains a shopping cart and an order and each order contains order details as well as shipping information.

Process Model vs Object Model

The process model is very valuable for describing some aspects of this system but it does not show the full picture. It helps the development team to understand how data flows through a system, including the sources of data, how data is processed, and where it is stored. However, it does not provide much detail into how this is accomplished.

Just as the process model does not provide all of the details of this system, the same is true of the object model. It will provide a lot of detail that makes it easier to understand the relationships between different classes and how they interact with each other. This clears up the "how" aspect of a process diagram. However, it mostly describes how the system works statically and does not provide much clarity of how the system works in motion.