**Seng 301 – Assignment 4 justification**

My project design consists of three facades on top of the Hardware façade, and a business class. The three facades do not communicate with each other but instead communicate downward with the hardware facade, and upward to the business class. The business class can use and retrieve information from these three facades in whatever way it pleases allowing it to make calls in one façade based on the fields or events of another. This allows for a connected system without inter-communication between the three facades. The business class operates exclusively on the three facades and does not directly influence the hardware façade.

I have designed the communication façade so that a business class can easily find information from the hardware facade and use it in whichever way it pleases. The communication façade listens to events fired by the hardware façade and based on those events, handles them by firing other events that the business class can then handle. For example, the communication façade is designed to fire events for when a selection made, out of stock, or product vended event is fired in the hardware façade. Most of the communication façade consists of output of information which can be used to implement a display for the customer, wherever needed. For input, the user can communicate with the machine only by pressing a button letting the communication façade know the order they would like to purchase. For this I have created a press method that determines the product selected and fires a selection made event. The communication façade stores some information as global variables as well, including the button pressed, and some information on the item selected. These can also be used by the business class and for extensibility purposes.

The payment façade is designed to handle transactions, keep track of money, and manage funds. To communicate with the business class, it can fire two events, a funds sufficient and a funds insufficient event which indicate whether the physical funds inserted in to the vending machine are sufficient for the selected purchase. These however do not take into account the credit stored, this is left for the business class to interpret. The payment façade keeps track of many things related to the money inserted and contained in the hardware. The available funds keeps track of all physical coins inserted into the hardware. SufficientFunds is a Boolean that much like the events can be used by the business class to check payment. The productCost represents the price of the item selected, changeNeeded the change required after purchase, and insertedCoins keeps track of all the coins inserted by the customer. The credit variable allows for further extensibility giving the business class the ability to create other forms of payment. The credit also keeps track of insufficient returns of change made by the hardware. The payment façade allows the business class to insert physical coins into the hardware as well as credit with the InsertFunds methods. The insertSuccessful method handles the coinAccepted event ensuring the coin inserted is done so properly. CheckFunds checks if the funds are sufficient for an inputted product, and returnChange returns whatever coins it can based on the returnChange variable, and stores whatever it cannot as credit. This is done with the ProcessWithCredit method.

The product façade is intended to keep track of the products inside the vending machine. The deliverProduct method can be called by the business class to deliver a product to the delivery chute. The product delivered is determined by the button pressed by the user which is obtained through the communication façade, and used by the business class. The loadProduct method allows the business class to load the vending machine.

For my implementation of the business rule, the business rule responds to events fired by the other facades and handles them with the methods SelectionMade, ReturnToDelivery and FunsInsufficiantHandler. Selection made checks if the funds are sufficient and if they are returnToDelivery is called, if not then the Insufficient handler checks the credit.