SENG-300 Iteration 3 Team - Q

George Abouseta, Carlo De Guzman, Reeshad Faiyaz, Eric Gantz, Aaron Gao, Amna Hassan, Collin Ho, Deivydas Johnson, Peter Kuchel, Christopher Lee, Aoqi Li, Joanna Lin, Rachel Ralph, Haoyang Shi, Peter Duc Tran, Rei Tsunemi, Evan Wong, Dingkai Wu

GUI Design: For GUI design, we used the Window Builder extension in eclipse to make the design process more effective, this decision was approved by Dr. Walker in a discussion board post. The DataPasser class is used to pass data collected between the UI and use cases.

Design Choices: We decided to use several interrelated classes for use cases, such as the use case for Receipt Printer has several related classes. We made this decision in order to optimize how many people can be working on the project at a time, as well as to increase cohesion of each class. While this does mean that certain classes are highly coupled, we believe that this was offset by the organizational benefits, and the ability to maximize parallelizability of development.

Item Adding and Placing Design: Items of different types are added to the users cart by their own specific adder classes (ItemAdder for barcodes, PLUAdder for PLUs), and a general container/interface for these is given by the checkout interface class. Items are added from the visual catalogue by these as well, depending on what code type that item uses. When items are scanned the scanners are disabled so the customer cannot continue adding items, and a **5 second timer is started in a new thread**, the BaggingTimeout class is the timer task for this, and if the user has not placed their items in the bagging area after 5 seconds they are notified to do so. When an item is placed in the bagging area the Item Placer class is notified and the item is confirmed to be the expected item, which will be the most recently scanned item.

Payment Design: Any use cases involving payment are split into separate classes that handle Coins, Banknotes, Card payments & PaysWithCash that consolidates Coin & Banknotes and deals change back to the customer.

PaysWithCash class: This payment class consolidates BigDecimal values returned from CoinRunner & BanknoteRunner in order to determine the total amount that the customer has paid. If the customer is owed any change, highest to lowest currency denominations are deducted first in a loop, until the value of change owed is zero.

Hardware Bugs: One thing that our payment team noticed was that CoinStorageUnit/ CoinStorageUnitObserver hardware was not able to detect the value of the coin, which may be due to a hardware bug. As a result we used CoinValidatorObserver to determine if a coin has been added to the checkout machine and its value.