Names: Ronald Macmaster, Michael Marino

UT EID: rpm953, mm75343

PA3: Shopping Cart: OOP

1) Analysis

Problem Statement:

Implement a Shopping-Cart mechanism for an online vendor.

Users will add grocery, electronics, and clothing purchases to the shopping cart.

The program shall compute and report the total price of the shopping cart purchases from:

Base price, tax, and shipping cost for each item.

Input:

Transactions are input from a given file. The filename is specified through the command line.

General Transaction Format:

<operation> <category> <name> <price> <quantity> <weight> <optional field 1> <optional field2>

Valid operations: **insert, search, delete, update, and print.**

Input is not necessarily valid! Report errors using exceptions.

Output:

Print command invokes shopping cart statement.

The statement prints all the shopping cart items in order by name with its attributes.

The total charges for the entire cart will follow.

Questions?

Should we implement the Item price as two ints for dollars and cents? Can we use a double?

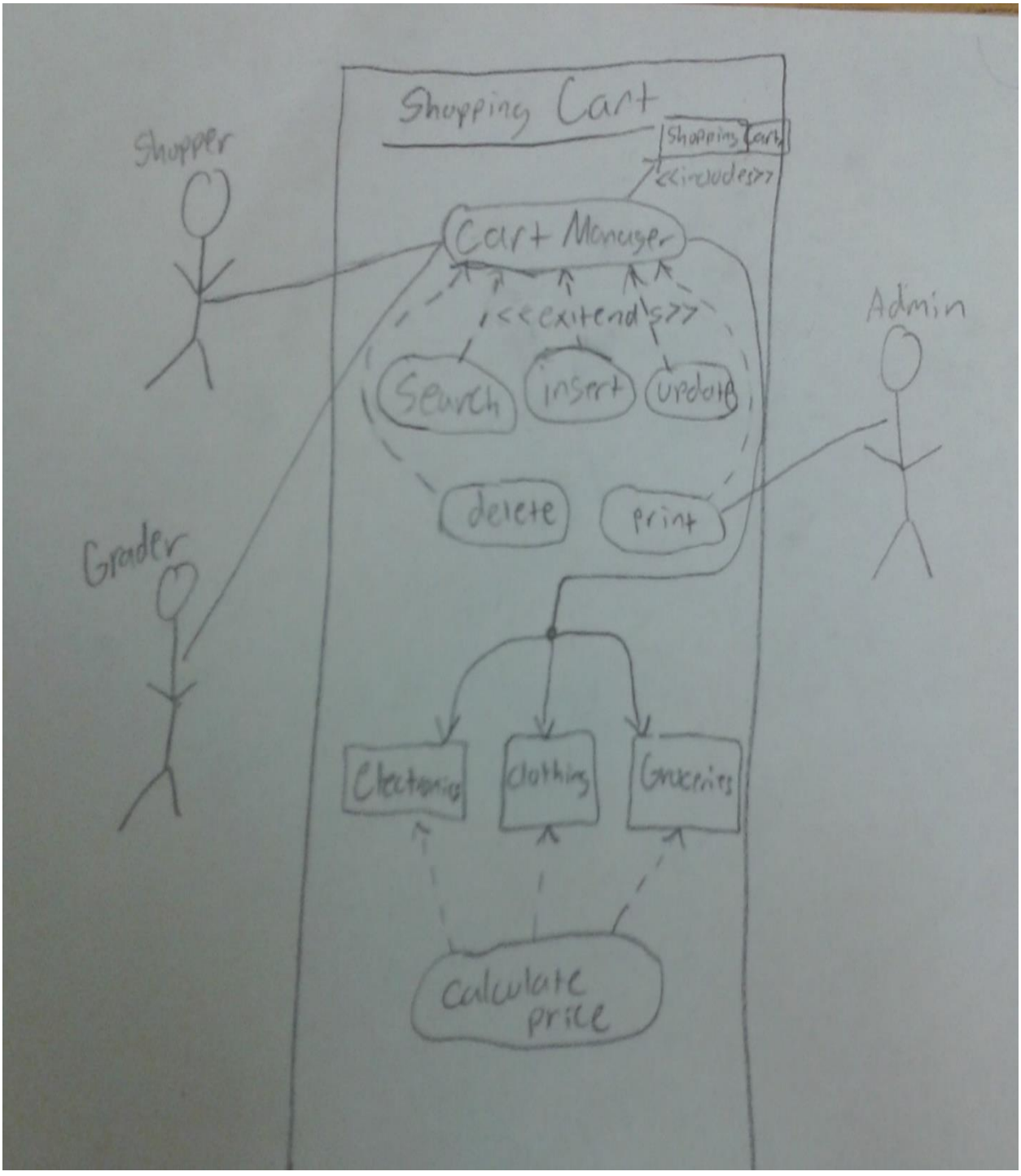
What is a system level use case diagram?

Should we create a separate class or method to read the transactions from a file?

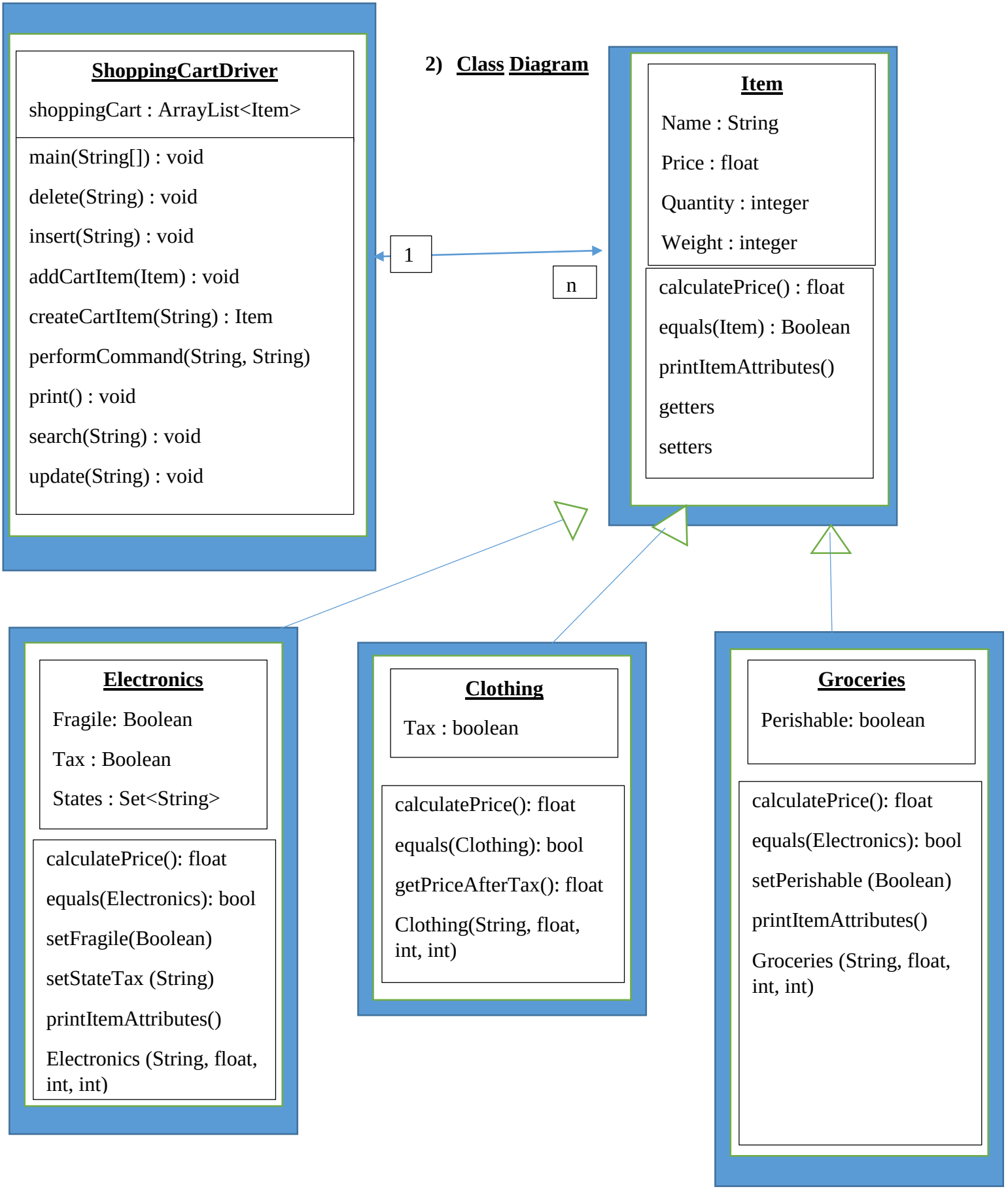
2) Design

Architecture Models:

1) System Use-Case Diagram



2) Class Diagram



3) **ADT class description for each class.**

ShoppingCartDriver

The driver models the shopping cart using an ArrayList of item objects. It can perform 5 commands on the shopping cart: insert, delete, updata, search, and print. Finally, it can create as well as add items to the shopping cart. It should also work if the cart is empty

Item

Item stores the basic common attributes of any shopping cart item. It contains the object's name, price, quantity, and weight. It is able to calculate the total price of the item bundle by multiplying the price * quantity. It can compare two objects by name, price, and weight to test for equivalence. A calling class can retrieve / update the Item fields as well.

Groceries

Groceries are a subset of the Item set. They are perishable but not taxable. Aside from printing the normal item attributes, it can also print a taxable field.

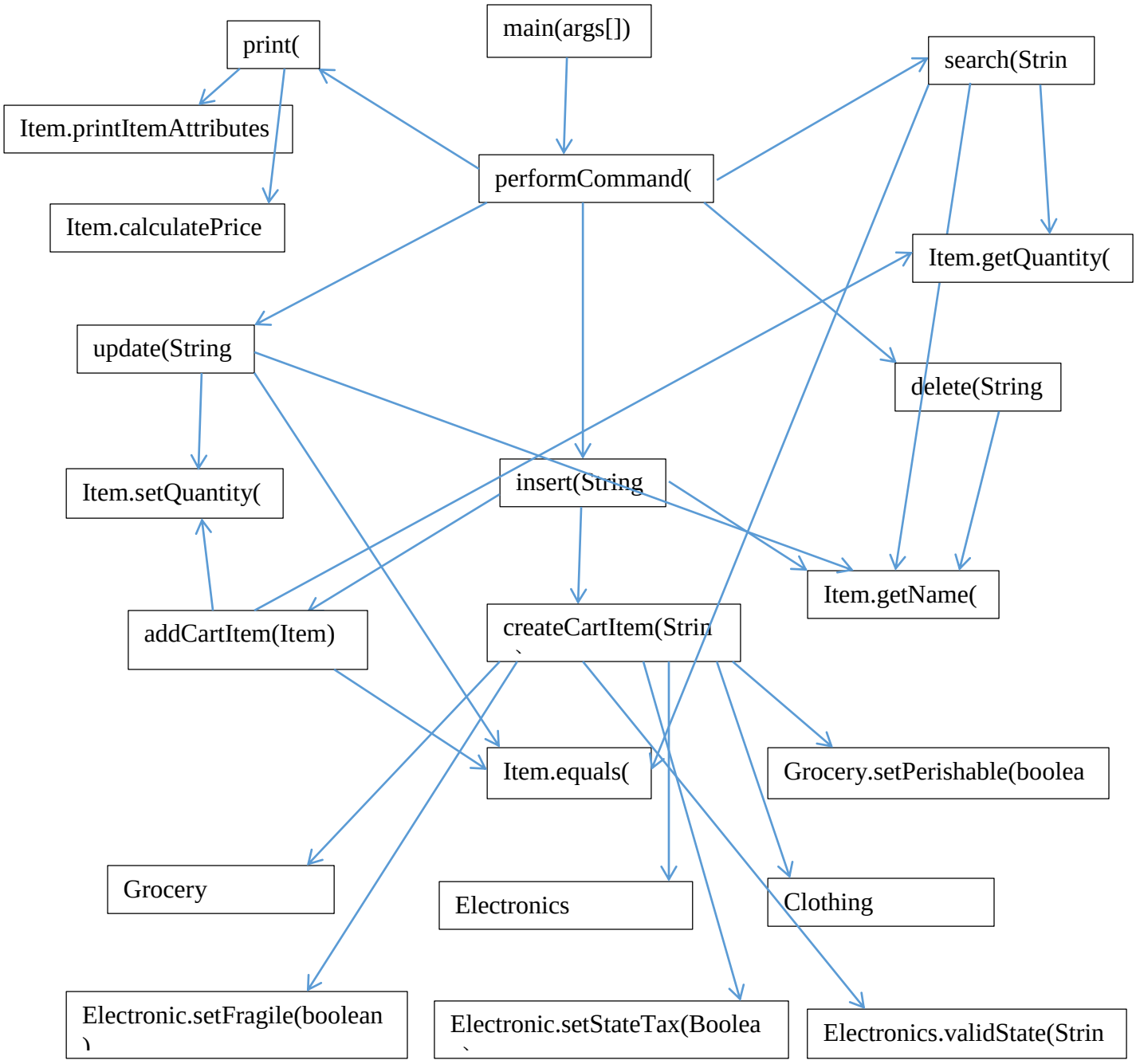
Electronics

Electronics are a subset of the Item set. They are both fragile and sometimes taxable. Aside from printing the normal item attributes, it can also print a taxable field and a fragile field. Some destination states do not include a tax rate, so these are checked for in the set of taxable state codes.

Clothing

Clothing is a subset of the Item set. It is a taxable item, and the tax is always applied. The class can be updated to find a way to avoid taxation in the future.

4) **Functional Block Diagram**



Algorithms

Driver Algorithm: (ShoppingCartManager)

1) Open the transaction file
2) Parse the transactions
3) Perform the Transaction (Insert, Search, Delete, Update, Print)
4) Repeat from 2) for the entire Transaction file.

Insert Algorithm:

1) Create an Item Object
2) Add the Item to the Shopping Cart List

Delete Algorithm:

1) Delete Item from Shopping Cart if exists
2) Otherwise, throw an error

Search Algorithm:

1) Look up the Item in the Shopping Cart
2) Print the number of hits for the item in the Shopping Cart

Update Algorithm:

1) Search for the first occurrence of the item in the Shopping Cart
2) If the Item exists, update the quantity at the first occurrence

3) Log the item name and new amount. If it doesn't exist, log that it was not found

Print Algorithm:

1) Print the Item Attributes to the screen