Software Modeling Dev with UML
CSP- 586
Assignment #3

Chen Xu A20377739

1. Complete list of classes

- Customer
- PowerMember
- Account
- Salesman
- Transaction
- Payment
- CheckPayment
- CardPayment
- CashPayment
- Order
- Pre-orderOrder
- Trade-inOrder
- RentalOrder
- StoreManager
- Product
- Game
- Console
- WarrantyPlan
- EmailNotification
- SaleEvent
- EmailSubscribe
- ProductMail
- EventMail

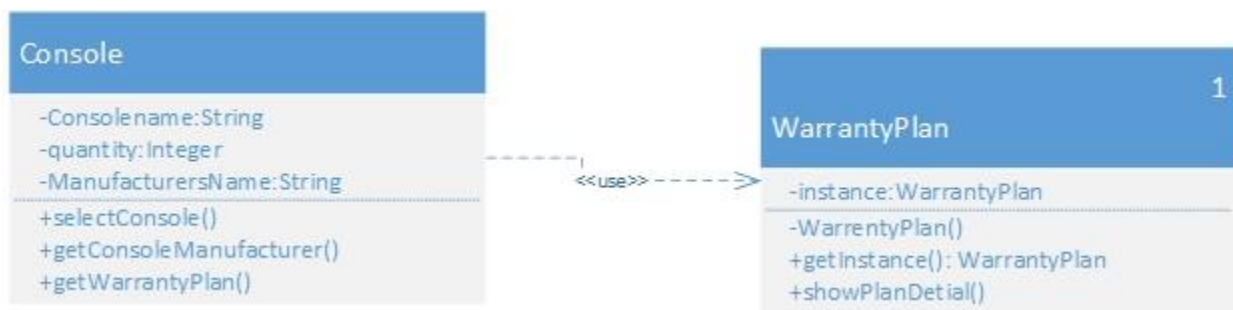# 2. Complete UML Design Model/class diag

PowerMember

-memberID: String
-expireDate: Integer

+membershipEnroll()
+membershipCancel()

Account

-AccountID: String
-Password: String

+login()

Salesmen

-salesmenID:String
-password:String

+createCustomerAccount()
+updateCustomerTransaction()
+updateCustomerOrder()

Transaction

-TransactionID

+getTransID()
+showTransDetial()

<<Interface>>
Customer

-customerID:String
-name:String
-billingAdress:String
-cellPhone:Integer
+e-mai:String

+CreateOrder()
+UpdateOrder()
+CancelOrder()
+UpdatePayment()
+UpdateProfile()
+printTransaction()
+emailNewProducts()
+emailNewEvents()

CheckPayment

-amount: Integer
-CheckNumber:Integer
-RoutingNum: Integer

+confirmPayment()
+reviewPayment()

Payment

-amount:Integer
-billingAdress:String

+printPaymentReceipt()

CardPayment

-amount:Integer
-CardNumber:Integer
-expireDate: time
-SecurityNum: Integer

+confirmPayment()
+reviewPayment()

Order

-OrderID:String
-OrderDate:Date
-amount:Integer
-OrderName:String

getOrderStatus()
selectOrderType()

PaymentFactory

+makePaymentCategory()

CashPayment

-amount: Integer

+confirmPayment()
+reviewPayment()

Pre-orderOrder

-OrderID:String
-OrderDate:Date
-amount:Integer
-releaseDate:Date
-productName:String

+checkAvailable()
+placeOrder()

Trade-inOrder

-OrderID:String
-OrderDate:Date
-amount:Integer
-productName:String

+checkPrice()
+placeOrder()

RentalOrder

-OrderID:String
-OrderDate:Date
-amount:Integer
-expireDate:Date
-productName:String
-rentPlan:String

+chooseRentOption()
+placeOrder()

<<Interface>>
product

-ProductName:String
-quantity:Integer

+getCategory()
+getProductDetail()

StoreManager

-StoreManagerID:String
-password:String

+setSpecialDeal()
+addProduct()
+updateProduct()
+deleteproduct()

Game

-ProductName:String
-quantity:Integer
-Game MakersName:String

+selectGame()
+getGameCategory()
+getGameMaker()

Console

-Consolename:String
-quantity:Integer
-ManufacturersName:String

+selectConsole()
+getConsoleManufacturer()
+getWarrantyPlan()

WarrantyPlan
1

-instance:WarrantyPlan

-WarrentyPlan()
+getInstance(): WarrantyPlan
+showPlanDetial()

<<Interface>>
EmailSubscribe

+subscribeEmail()
+unsubscribeEmail()
+notify()

EmailNotification

-Notification: String
-ProductName: String
-Price: Integer
-EventName: String
-EventTime: time

+getSaleEventState()
+getProductState()

SaleEvent

-EventName: String
-Eventime: time

+setEventContent()

ProductMail

+emailNewProducts()

EventsMail

+emailNewEvents()

observers

0*:1
1:1
1:1
-1:0*
0*:1
0*:1
1:0*
1:1*
1:1*
1:1*
1:1*
0*:1
1:1*

3. List of the Design pattern(s)

1) Singleton Design Pattern

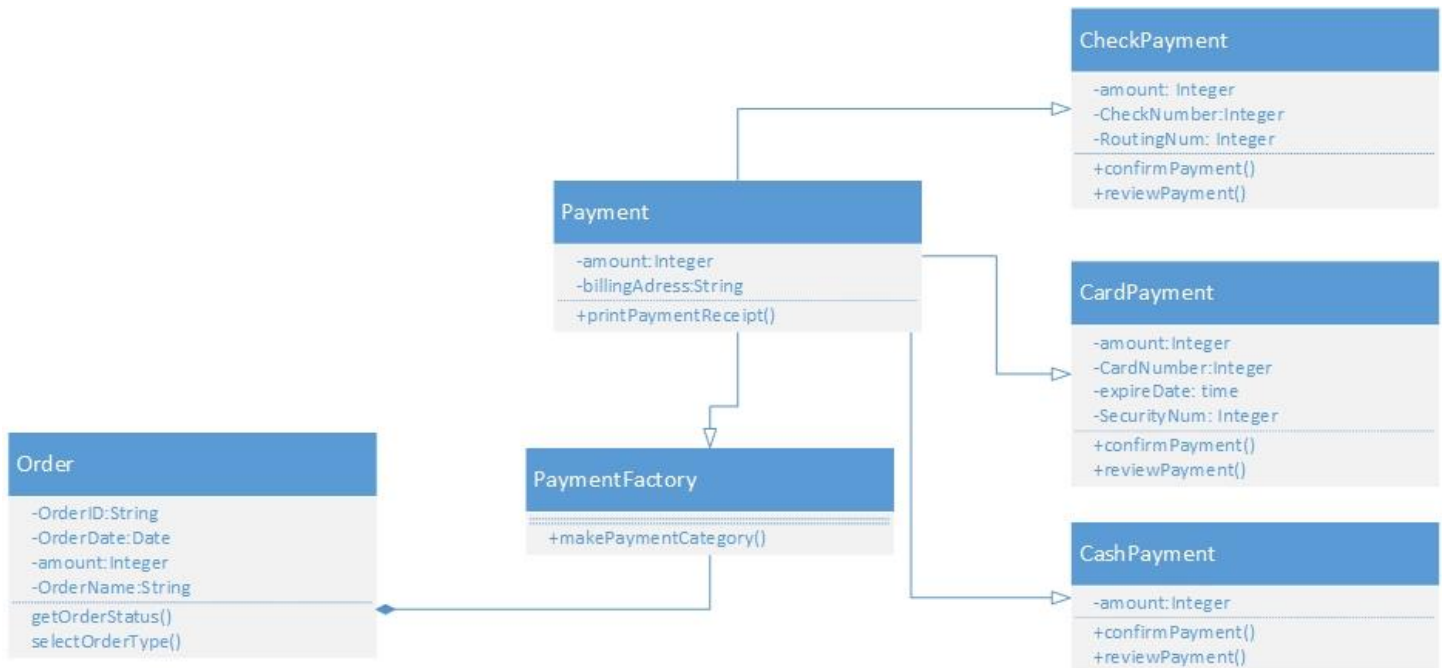2) Factory Method Design Pattern

3) Observer Design Pattern

4. Documentation of used design patterns

1) Singleton Design Pattern: Implement select console Warranty plan:

I. Use singleton Pattern to make every console use the instance that instantiated by WarrantyPlan Class.

II. WarrantyPlan has Instance in the Class and Console can get it by using getInstance() method.

2) Factory Method Design Pattern: Implement Order make payment:

CheckPayment

-amount: Integer
-CheckNumber:Integer
-RoutingNum: Integer

+confirmPayment()
+reviewPayment()

Payment

-amount:Integer
-billingAdress:String

+printPaymentReceipt()

CardPayment

-amount:Integer
-CardNumber:Integer
-expireDate: time
-SecurityNum: Integer

+confirmPayment()
+reviewPayment()

Order

-OrderID:String
-OrderDate:Date
-amount:Integer
-OrderName:String

getOrderStatus()
selectOrderType()

PaymentFactory

+makePaymentCategory()

CashPayment

-amount:Integer

+confirmPayment()
+reviewPayment()

I. Order get payment instance through PaymentFactory Class and use the
    payment category of the Order to instance the payment in right way.

II. CheckPayment , CardPayment and CashPayment Class are the different
    category of payment that inherit from the Payment Class.

3) Observer Design Pattern: Implement email subscribtion:
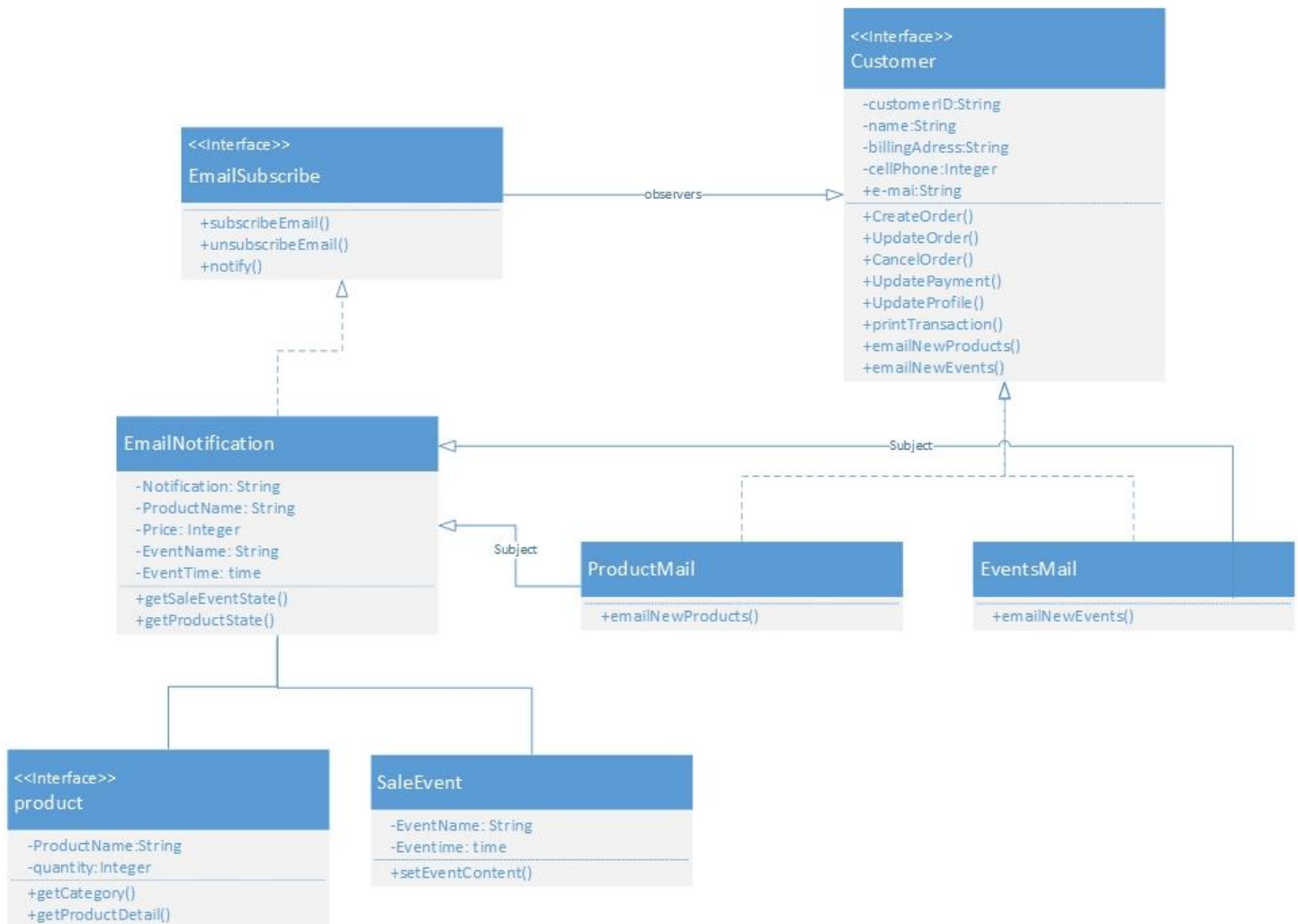
I. the customer will subscribe or unsubscribe to get the notification of new products and events .

II. The Customer is the observer interface, it can use ProductMail and EventsMail Class to get new product and event notification.

III. The EmailNotification Class implement the subject interface EmailSubscribe.

IV. ProductMail and EventMail Class are subclass of EmailNotification Class.

V. EmailNotification Class can get state of information from the product and

SaleEvent Class.

5. Capture design model class diagram(s)

Visio Professional window showing a UML class diagram with the following classes:

(Left top class)
- -OrderID:String
- -OrderDate:Date
- -amount:Integer
- -releaseDate:Date
- -productName:String
- +checkAvailable()
- +placeOrder()

(Middle top class)
- -OrderID:String
- -OrderDate:Date
- -amount:Integer
- -productName:String
- +checkPrice()
- +placeOrder()

(Right top class)
- -OrderID:String
- -OrderDate:Date
- -amount:Integer
- -expireDate:Date
- -productName:String
- -rentPlan:String
- +chooseRentOption()
- +placeOrder()

<<Interface>> product
- -ProductName:String
- -quantity:Integer
- +getCategory()
- +getProductDetail()

Game
- -ProductName:String
- -quantity:Integer
- -Game MakersName:String
- +selectGame()
- +getGameCategory()
- +getGameMaker()

Console
- -Consolename:String
- -quantity:Integer
- -ManufacturersName:String
- +selectConsole()
- +getConsoleManufacturer()
- +getWarrantyPlan()

StoreManager
- -StoreManagerID:String
- -password:String
- +setSpecialDeal()
- +addProduct()
- +updateProduct()
- +deleteproduct()

WarrantyPlan (1)
- -instance:WarrantyPlan
- -WarrentyPlan()
- +getInstance(): WarrantyPlan
- +showPlanDetial()

<<Interface>> EmailSubscribe
- +subscribeEmail()
- +unsubscribeEmail()
- +notify()

EmailNotification
- -Notification: String
- -ProductName: String
- -Price: Integer
- -EventName: String
- -EventTime: time
- +getSaleEventState()
- +getProductState()

SaleEvent
- -EventName: String
- -Eventime: time
- +setEventContent()

ProductMail
- +emailNewProducts()

EventsMail
- +emailNewEvents()

Callout: Insert 'Member' Shape

Shapes panel — UML Class:
Class, Member, Separator, Interface, Enumeration, Package (expanded), Package (collapsed), Note, Inheritance, Association, Directed Association, Aggregation, Composition, Interface Realization, Dependency