

OOP Assignment

by Thorbjorn Ravnaas Kristiansen

FILE	2652_THORBJORN_RAVNAAS_KRISTIANSEN_OOP_ASSIGNMENT_108537_1322374492.DOCX		
TIME SUBMITTED	21-MAR-2018 07:23PM (UTC+0000)	WORD COUNT	581
SUBMISSION ID	85248861	CHARACTER COUNT	4501

University of Wales, Trinity St. David

Icarus Sales Service

BEng Computer Systems and Electronics

Thorbjørn R. Kristiansen (P151226 / 1651093)
March 2018

Contents

Introduction	2
User Manual	3
Features	3
Running the Program	3
Administrator Portal	5
Technical Documentation	7
UML Diagrams	7
Code Examples	13
Overloading	13
Inheritance	13
Object Usage	13

Introduction

This is the technical documentation and user manual for the Icarus Sales System (ISS) software. This software is designed for medium-large businesses who wish to enter the component sales market at a simple level.

User Manual

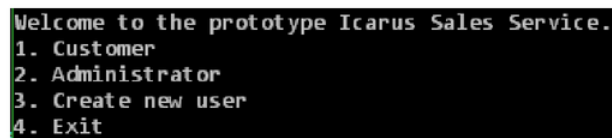
Features

This software features shopping baskets, user lists, inventory control, an administrator portal and discount control. These will be further explained in the document in either the user manual or with included diagrams.

Running the Program

When launching the profile, a prompt with either creating a new user, logging in to the customer portal or logging in to the administrator portal appears (fig. 1).

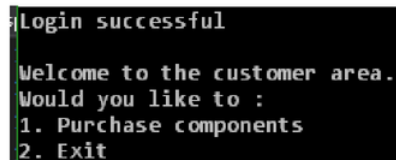
Note: Entering a category number which doesn't exist prompts the user to try again until a valid number is inputted. This recurs through the program UI.



```
Welcome to the prototype Icarus Sales Service.  
1. Customer  
2. Administrator  
3. Create new user  
4. Exit
```

Figure 1 - Initial menu

In the customer menu, the user is given a choice between purchasing components or exiting the program (fig. 2). Exiting the program will give the option of clearing the shopping basket before closing. Choosing option 1 will prompt the user to log in.



```
Login successful  
  
Welcome to the customer area.  
Would you like to :  
1. Purchase components  
2. Exit
```

Figure 2 - Customer portal

If option 1 is chosen, the user is presented with the various components available for purchase and their prices. The current component stock is also shown. The user can now choose to either buy components, view shopping basket, reverting to the customer portal, empty the shopping basket or exiting the program (fig. 3).

```

Welcome to the purchase portal.
Current price list:
ID           Price           Stock
Resistor     0.5             500000
Capacitor    0.5             500000
Diode        0.66            500000
Transistor   31.48           500000
Inductor     12.5            500000

Please choose:
1. Buy resistors
2. Buy capacitors
3. Buy diodes
4. Buy transistors
5. Buy inductors
6. View basket
7. Empty basket
8. Back to customer portal
9. Exit program

```

Figure 5 - Customer purchase menu (note that the prices are to be treated as examples)

If alternatives 1-5 is chosen, a standardised prompt appears asking the user for the desired purchase quantity. Some components are restricted to a minimum purchase amount, which will prompt the user to correct the original requested quantity (fig. 4). After inputting a valid quantity, the component is added to the shopping basket and the price is calculated with any applicable discounts (fig. 5).

```

Please choose:
1. Buy resistors
2. Buy capacitors
3. Buy diodes
4. Buy transistors
5. Buy inductors
6. View basket
7. Empty basket
8. Back to customer portal
9. Exit program
2

How many? (Minimum quantity: 500) 200

How many? (Minimum quantity: 500)

```

Figure 4 - Minimum purchase quantity not satisfied

```

How many? (Minimum quantity: 500) 500
Since you have ordered more than 250 components, you have qualified for a 20% discount.
Old price for 500 Capacitors is 250
New price is 200
Your order of 500 capacitor(s) for 250 Norwegian Kroner is added to your basket.
Buy more? Y/N

```

Figure 3 - Discount applied

Viewing the basket will list the components purchased within the current session (fig. 6).

Component	Amount	Price
Resistor	100	50
Capacitor	500	200
Press any key to continue . . .		

Figure 6 - View basket

The last feature accessible by the end-user is creating a new username and password for new customers. When choosing option 3 from the initial menu, the user is prompted with entering a new username and a new password. Because of software limitations, these cannot contain spaces.

Administrator Portal

The administrator portal allows for stock control both on the component level and the subcomponent level. It also allows any administrator to change the prices of components sold by the company and makes password management possible. The administrator menu (fig. 7) is shown after the user enters the correct password (default: "2").

```
Welcome to the admin portal. Would you like to:
1. Manage prices
2. Produce components
3. Order raw materials
4. Change password
5. Exit
```

Figure 7 - Administrator menu

In the "Manage prices" option, the administrator can change the price of individual components (fig. 8). A confirmation is printed to screen when the price change is successful (fig. 9).

```

Current price list:
ID           Price           Stock
Resistor     0.5             499900
Capacitor    0.5             499500
Diode        0.66           500000
Transistor   0.43           500000
Inductor     0.54           500000

Please enter which component you would like to change the price of
1. Resistor
2. Capacitor
3. Inductor
4. Transistor
5. Diode
6. Back to admin portal

```

Figure 9 - Price change menu

```

Please enter new transistor price: 0.43
Success! New transistor price is 0.43
To go back to main menu, please type m. To exit, type exit.

```

Figure 8 - Price change complete

The "Produce components" option allows for the administrator to produce individual components or replenish the entire component stock (fig 10). This decrements the subcomponents stock file automatically with the right amounts.

```

Current price list:
ID           Price           Stock
Resistor     0.5             499900
Capacitor    0.5             499500
Diode        0.66           500000
Transistor   0.43           500000
Inductor     0.54           500000

Would you like to produce more of:
1. Resistors
2. Capacitors
3. Diodes
4. Transistors
5. Inductors
6. Replenish all
7. Back to admin portal

```

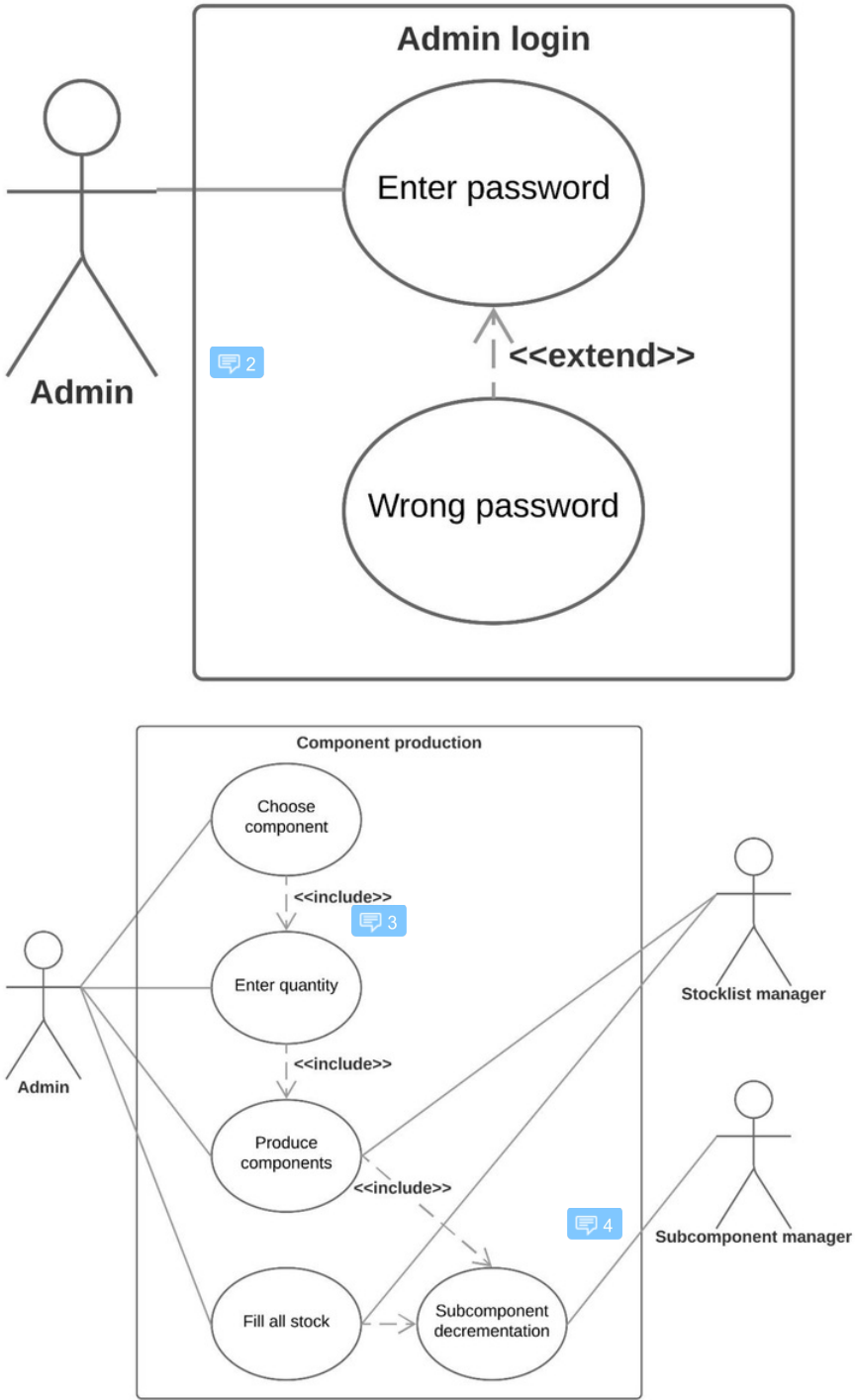
Figure 10 - Production menu

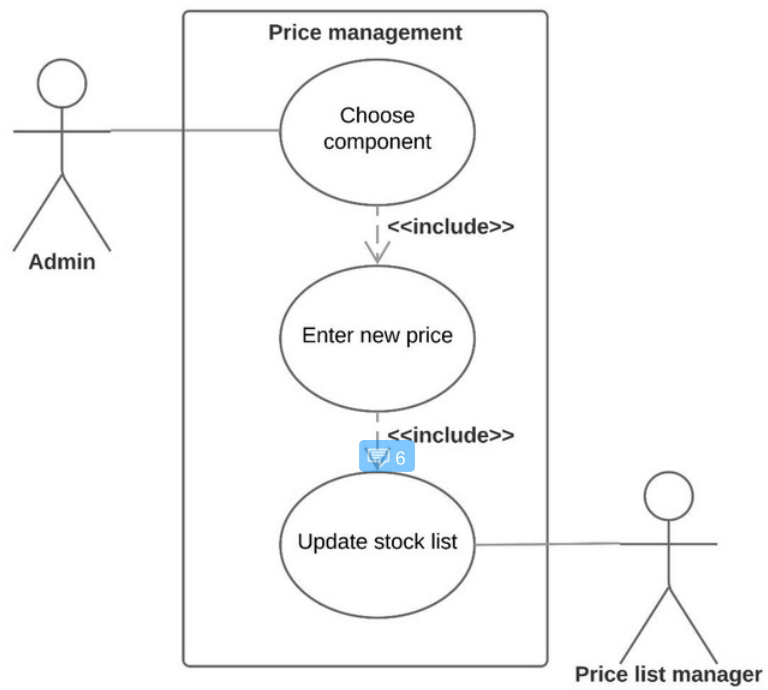
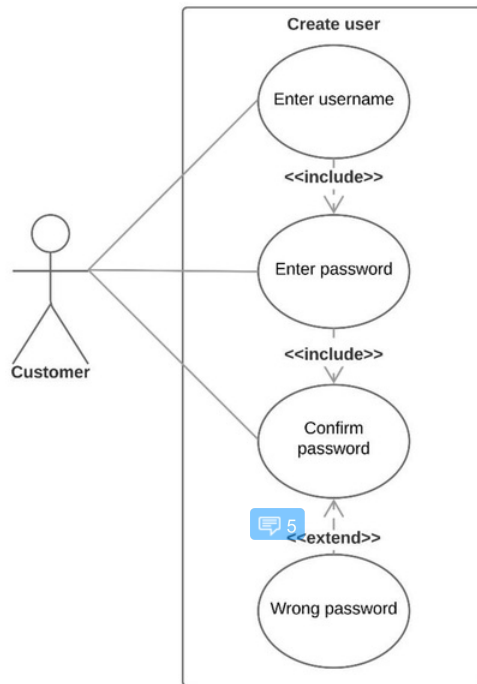
Safeguards are in place, so the stock list cannot be overfilled. If subcomponents are lacking, the administrator will be prompted to reorder subcomponents first.

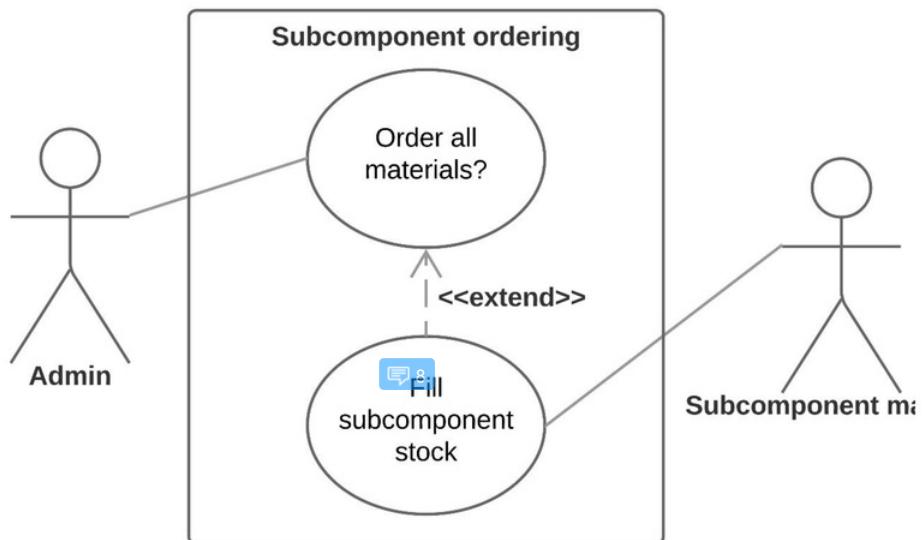
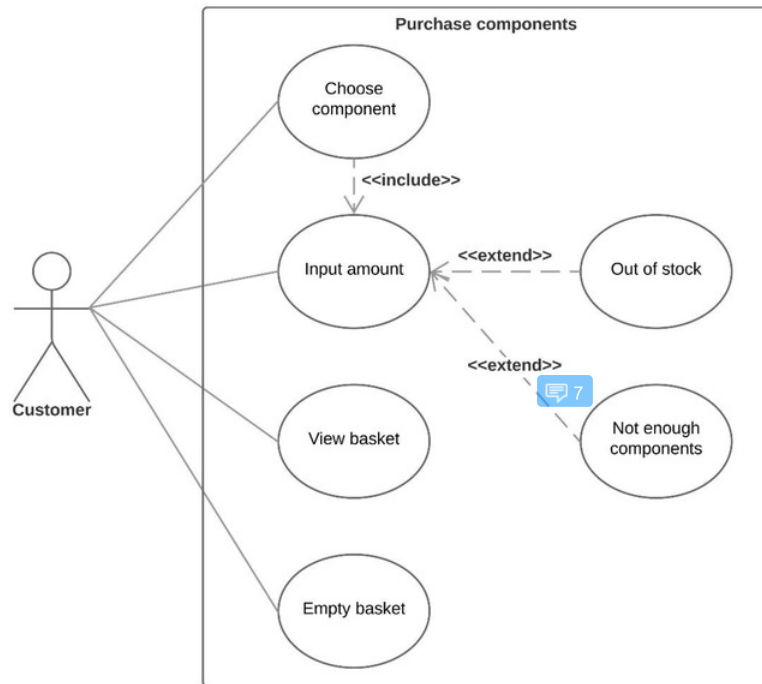
The "Order raw materials" option fills the subcomponent stock to default max.

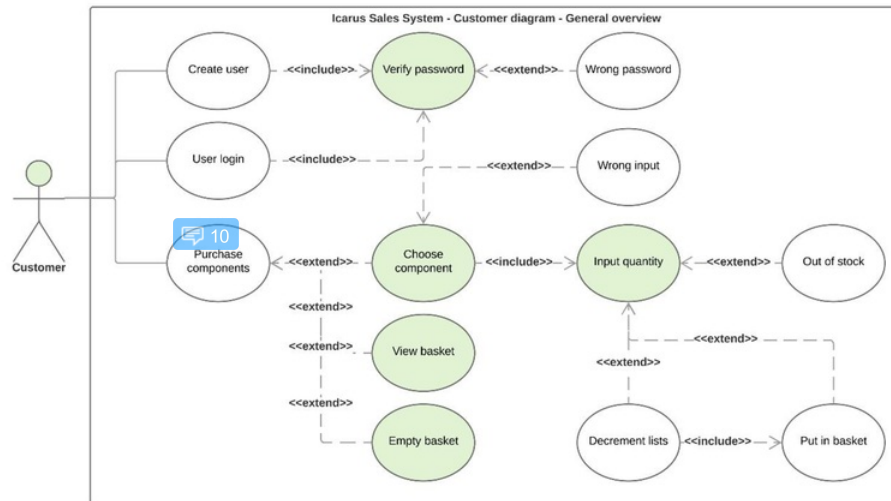
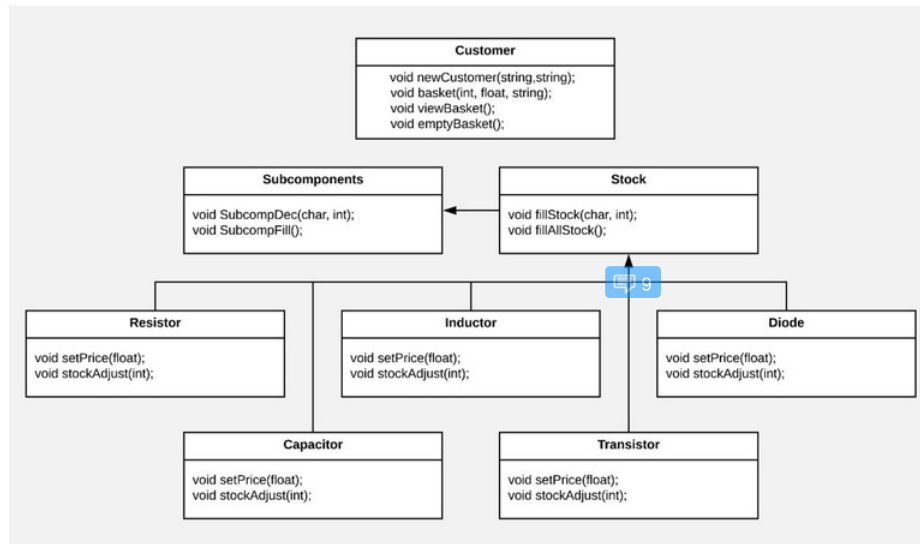
Technical Documentation

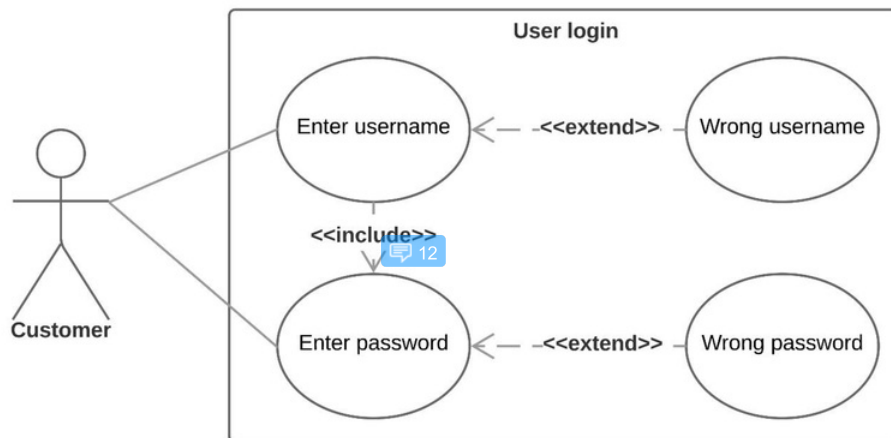
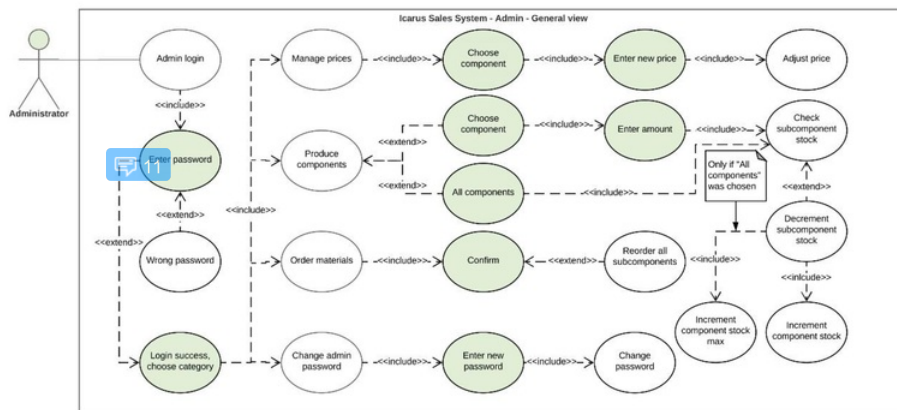
UML Diagrams

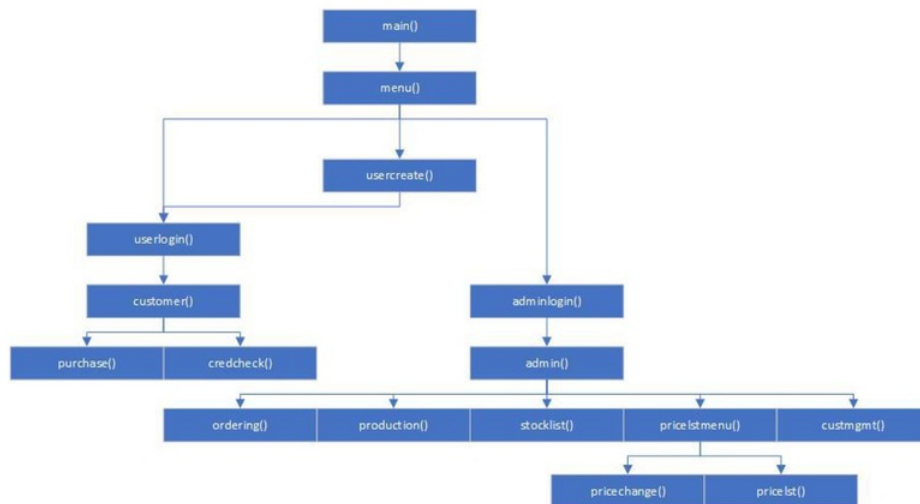












Code Examples

These are some examples of the different techniques utilised in object-oriented programming. The following images are excerpts from the program code.

Overloading

```
void Subcomponents::SubcompDec(char receive, int amount) //Subcomponent decrementation function
```

Inheritance

```
#include "Subcomponents.h"  
#include "Stock.h"
```

Object Usage

```
Stock* stock = new Stock;  
stock->fillAllStock();
```

OOP Assignment

GRADEMARK REPORT

FINAL GRADE

GENERAL COMMENTS

Instructor

32/50

PAGE 1



Comment 1

Novel name, however, there may be a copyright issue

PAGE 2

PAGE 3

PAGE 4

PAGE 5

PAGE 6

PAGE 7

PAGE 8



Comment 2

Good example of a use-case diagram



Comment 3

This could be either include or extend here be careful of the way you word use-cases



Comment 4

Again another good diagram if a little unclear

PAGE 9



Comment 5

This is a good diagram



Comment 6



Comment 7

Probably essential checks not optional in which case would be an include not an extend



Comment 8

representative of the software created, may not be the most logical way of tackling this problem



Comment 9

This arrow doesnt exist in UML class diagrams. An empty standard arrow represents inheritance rather than a filled arrow.



Comment 10

This is an accurate representation of the system you have produced and is very detailed



Comment 11

Again, a well thought out diagram



Comment 12

Include unnecessary here



Comment 13

This is a good example, however, you could have explained a little more and included the default signature for comparison



Comment 14

This in itself does not evidence inheritance, examples of the class definition would have been a far better example. As with overloading, a little more explanation could have helped here



Comment 15

This is a good example of instantiation here. Could have provided a small explanation of the purpose and benefits

