Computing Coursework

Matt Ling

October 24, 2014

Contents

Chapter 1

Analysis

1.1 Introduction

1.1.1 Client Identification

My client is Tom Henderson; he is thirty seven years old and is the principle veterinary surgeon and managing director of Beacon Veterinary. Tom graduated from Melbourne University, Victoria, Australia in 2000 and found employment with the Beacon Vet Centre in Aspatria in 2003. Since then Tom has brought partnership in the practice.

As well as being the principle surgeon, Tom also takes resposability for other roles, such as managing the businesses finance, organising staff payrolls and Managing the stock of the products they sell and equipment they use. On busy days, usually during the weekdays, Tom spends most of his working hours in the operating thetre. However, on less busy days or outside of opening hours, Tom spends time managing and organising the stock of the products the Veternary sell. To aid him, Tom uses an Edexcel Spreadsheet to record and track the stock of each product.

Tom would like me to develop a new system for him because due to recent sucess in the business, Beacon Vets has introduced an increased range of products available to sell. Tom has requested for this new system because he stated that his current system is now becoming too complicated and difficult to use now that the new products have been added. My client wants the new system to be easier to use and wants managing the stock to be a faster prcoess, so he doesn't have to spend as much time at the veternary centre and can spend more time with his children during the weekends.

1.1.2 Define the current system

Despite being a veterinary centre, Beacon vets sold roughly a dozen products, such as Vetericyn, a non toxic cut / wound treatment for animals. The products sold at the Vets are usually products to maintain the animals health after an operation, however Beacon Vets are now selling products such as dietry foods and other products that customers can just come in and buy without coming in to see the Vet. Because only a dozen items were being sold, and each customer never purchased over about 3 products, keeping track of the stock of each item used to be a relativly easy process.

Last year Beacon Vets expanded on the range of the products they sold, providing a wide variety of foods and treatments for all sorts of animals such as dogs, cats and rabbits etc. Although Beacon Vets greatly expanded on the products they sold, they did not change their system of keeping track of stock. The current system is a Microsoft Excel spreadsheet containing the information of the product, the selling price, the current amount in the front of the store, the current amount being stored in storage. This current system was useful and easy to use when handling a dozen products but has become difficult and confusing now many more products are being sold. here is the current Edexel Spreadsheet in use:

item	price	front stock	back stock
Drontal Plus Flavour Bone Shaped Dog Worming tablet (3 Pack)	11.95	7	11
Drontal Plus Flavour Bone Shaped Dog Worming tablet (single)	£4.99	4	8
Boehringer Seraquin Cat & Dog	£1.55	3	6
Molar Proden Plaque Off Animal for Cats & Dogs 60g	£9.10	6	12
Molar Proden Plaque Off Animal for Cats & Dogs 180g	£24.90	4	6
Molar Proden Plaque Off Animal for Cats & Dogs 420g	£48.95	6	4
Protexin Pro Kolin Plus (15ml)	£8.90	6	5
Protexin Pro Kolin Plus (30ml)	£12.25	9	8
Bayer Drontal Puppy Worming Suspension	£9.49	2	8
Bayer Drontal Puppy Worming Suspension	£16.90	5	4
Intervet Panacur Wormer Paste (5g)	£4.45	9	10
Virbac Epiotic Spherulites Ear Cleanser (60ml)	£5.90	7	16
Virbac Epiotic Spherulites Ear Cleanser (125ml)	£8.90	2	5
Bob Martin Ear & Eye Wipes (30 wipes)	£1.99	7	8
Endogard Plus Flavour Tablets for Dogs	£1.57	10	11
Virbac C.E.T Toothpaste And Toothbrush Kits (cat)	£8.30	7	7
Virbac C.E.T Toothpaste And Toothbrush Kits (dog)	£8.70	4	1
Gro-Well Feeds Joint Aid For Dogs (250g)	£10.45	2	5
Gro-Well Feeds Joint Aid For Dogs (500g)	£17.95	2	4
Gro-Well Feeds Joint Aid For Dogs (1kg)	£61.90	1	3
Johnsons Easy One Dose Wormer Dog (dog 6kg to 20kg)	£34.10	1	4
Johnsons Easy One Dose Wormer Dog (dog 6kg to 40kg)	£34.80	1	5
Johnsons Easy One Dose Wormer Dog (up to 80kg)	£35.90	1	4
Johnsons Sweet Breath Tablets	£2.80	11	17
Intervet Panacur 22% Granules (1.8g)	£0.90	7	6
Intervet Panacur 22% Granules (2.5g)	£1.90	4	6
Homeopet Skin & Itch Relief	£7.85	8	9
Blooming Pets Msm Cream For Dogs & Cats	£4.90	4	11
Denes Natural Rhus Tox Homeopathy Remedy	£8.10	4	9
Johnsons Pre-Biotic Pet Odour Tablets	£4.45	4	14
Virbac Vet Aquadent	£5.90	5	5
Aniwell Filtabac Sun Block Skin Cream (120g)	£7.40	2	11
Aniwell Filtabac Sun Block Skin Cream (500g)	£16.90	3	3
Ceva Hepatosyl Plus Capsules (x90)	£62.90	2	1
Ceva Hepatosyl Plus Capsules (x60)	£24.90	3	4
Bayer Keto Diastix Urinalysis Reagent Strips	£8.50	5	8
Peridale Granules for Dogs Stool Bulking Agent	£18.80	3	7
Vetplus Coatex Efa Liquid Pump (150ml)	£31.90	6	4
Vetplus Coatex Efa Liquid Pump (65ml)	£19.50	5	8
Pfizer Pet Cal Tablets Calcium Supplement (x180)	£30.00	2	2

Figure 1.1: An example of the current system being used.

Looking at *Figure 1.1*, The current system is a basic spreadsheet created using Microsft Excel containing 4 columns and a row for each product. The column ***item***, contains the information to uniquely identify each product. Some of the data entered into the item comlumn can be be long because if the Veternary sells different quantities or weights of the ptoduct, then this is specified in this column. the column ***price***, is used make sure that the item being edited is correct. This coumn is usually only used when a specif product has either two quantites or weights and the ***price*** column is used to second check that the right product is being edited. The ***front stock*** column specifies the quantity of each product currently being displayed in the front of the vets. This where the customers can find the product they want, and buy it. There is also stock of the each product being held in storage, which are moved to the front of the vets when the products available to the customers get low. The stock held in storage is under the column ***back stock***.

Every time products are sold, the document is manually changed. For example if the veterinary sold 1 bag of dietary dog food, Tom or another employee would search the system for the product that was sold, and reduce the front stock by one. When the total stock gets low (*front stock + back stock*) Then that product gets highlighted in red to indicate that it needs to be restocked.

When the new stock is brought in they put some of the new stock in the back and some in the front and update the stock in the front and back accordingly.

1.1.3 Describe the problems

The main problem identifed by my client is that the current system is becoming confusing and difficult to use. This is because as more products are being added the databse, its getting harder to find a specific product. my client now uses keyboard shortcuts to search the system for a specific product. This can speed up the process of finding a product, however there is still errors being made when a product has more than one quantity. This can cause the stock to be inaccurate and can sometimes cause a specific product to not be displayed at all in the vets without my client or the employees knowing.

Another problem with the current system is that sometimes the information within the system can be changed, however the data sometimes may not save when my cient closes the system. This is because the data has to be manually saved before the user closes the application. If my client forgets to save the information within the database, the data will not be updated. There can also be problems if my client is not sure whether the information was updated as there is no indication to whether it was updated or not.

Having to keep manually changing the document every time a product is sold, moved to the front of house or if a delivery comes in. Editing the document becomes tedious and sometimes errors are made changing the values. Selling the product and changing the document are two different processes. This is a problem as both these processes can be combined into one to improve the system.

The current operating system on the computer is Windows 7. The program Microsoft Excel which may not be compatible with other operating systems such as Mac or Linux. This is a problem with the current system because if a new Computer is brought with a different operating system, the current system may not be compatible and usable on the new computer. If the current system is used for a long period of time (for example 5 years), The version of Excel the system was created in may not be compatible with an up to date version of Excel. This could also be problematic because it would mean the current system would stop functioning. A similar problem would be that the current system was created using an up to date version of Excel and the current version of Excel on the computer is not up to date, which could result in the System not being compatible with the version of Excel installed on the computer.

1.1.4 Section appendix

Interview Questionnaire

1. What is the Current System?

- Done on Excel

- Shows product name and quantity in the front and back of vet

- when there are only 10 - 15 of the product left a new order is made

- when stock is low in the front, stock is moved from the back to the front

2. What are the problems with the current system?

- Data is sometimes not saved, which can cause problems

- Takes a long time to initially set up the system with all the products

- Every time a product is sold its data on the system has to be changed.

- There is a lot of data is needed to identify each product (Weight, 300g / 450g)

- Selling a product and managing the stock are two different processes.

3. What data is recorded in the system?

- Product Name

- Quantity if sold in different quantities

- Weight if sold in different weights (i.e. 500g / 750g)

- Quantity in the front of the vets(On display to the customer)

- Quantity in the back of the vets (Being stored in the storage room)

4. How many clients buy products on a day to day basis?

- Roughly about 10 - 20 people depend on how busy we are on that day.

5. How many products do each client buy?

- some customers may be advised to buy only one product, however some may be 5/6 depending on what problems their pet / pets have

6. What features should the new system have?

- The checkout system and the stock control system should be integrated together

- When a product is sold, its quantity is changed in the stock system automatically

- A search function to be able to search for a specific product.

- Categorize items so that alternative items can be given if there is a problem with the current product.

- A notification should be displayed if a product needs to be restocked

- The program should show which product and amount should be moved from the back of house to the front display.

- A picture of the product should be displayed so each product can be easily identified by a member of staff

- The product name, quantity, and weight (if needed) should be clearly displayed

- Because the checkout system and stock control should be integrated the price of each product should be displayed.

- The price of the product should be easily changed if it needs to be

7. are any parts from the current system going to transfer to the new system?

- The name of the product, quantity in the front of house, quantity in back of house will still be stored.

- When stock gets low in front of house, products will need to be moved from the storage room the front.

- When total stock gets low more products will have to be ordered in.

8. How often will data need to be input?

- Only Once when the system is initially set up

- When a new product is going to be sold its data will be needed to be put into the system

- When the price of a product needs to be changed

- If a customer wants to buy more than one of a specific item the quantity will need to be entered

9. what computer resources will the system have available to run on?

1. Windows 7 Home Edition
2. Intel® Core™ i3-3240T Processor (2.9 GHz)
3. 4 GB DDR3 RAM
4. 1 TB HDD, 7200 rpm

10. Is installing additional software?

-Not an issue unless there is a problem installing the new software or if I don't know how to install the new software.

1.2 Investigation

1.2.1 The current system

Data sources and destinations

Currently there is only one data source in the system. This is when data is input into the database. The data being put into the database is the product and the relative information about the product. The relative information being stored within the database is: the product name, the Product ID, The weight / quantity / volume of the product, the price, the current amount in the store, and the current amount in storage. The Product name is used for the identification by the customer, however one product may have different quantity or weights. The ProductID uniquely identifies every product, meaning the same product with two different quantities (i.e500g and 750g) will be identified differently. The weight / quantity / volume is specified because a specific product may come in various quantities. There is currently only one data destination which is the data is stored inside a database. There is only one data destination because once the data is stored inside the database, it is only viewed and edited. The data in the database is never taken out, so there is only one data destination.

This is the table for data sources and destinations:

Data Source	Data	Data Type	Data Example	Data Destination
Employee inputting the data	• *ProductName* • *ProductID* • *Size* • *Price*	• *String* • *Integer* • *Integerl* • *Real*	• *'Pedigree Joint care'* • *25* • *x20 / 500g 250ml* • *9.90l*	Database
Employee Adding / Editing Stock	• *Front Stock* • *Back Stock*	• *Integer* • *Integer*	• *Front Stock* • *Back Stock*	Database
Customer	Money	Real	20.00	Receptionist
Receptionist	Customer's Money	Real	20.00	Till
Receipt	• *ProductName* • *ProductID* • *Quantity* • *Price* • *TotalPrice* • *Amount Paid* • *Change*	• *String* • *Integer* • *Real* • *Real* • *Real* • *Real* • *Real*	• *Drontal Puppey worming Suspension* • *53* • *x1* • *9.99* • *13.50* • *20* • *6.5*	Customer
Receptionist quering the database for product price	• *Product Name* • *Qunatity*	• *Integer* • *Integer*	• *142* • *x20*	Database

Algorithms

There are many algorithms used in the current system. the algorithms have been split into the algorithms used when selling a product and algorithms used when handling the stock of the products. I have included the algorithms used when selling products because even though this is not the system I am creating, my client wants me to implement this process into the new stock control system so I have included some algorithms from selling products.

The first algorithm is used when a customer buys a product. The algorithm takes the total price so far (if the item is the first item of the list then total price = 0) and then adds the price of that product onto the total price. the next product is added and so on until all the products that the customer

wants to buy have been added and a total price has been calculated.

Algorithm 1 Adding Product to Total Price

```
1: Done ← False
2: TotalPrice ← 0
3: WHILE NotDone
4:     OUTPUT "Please enter the price of the product(0.00)"
5:     Price ← USERINPUT
6:     OUTPUT "Please enter the quantity of the product (0 − 100)"
7:     Quantity ← USERINPUT
8:     IF Price == 0 THEN
9:         Done ← True
10:        TotalPricel ← TotalPrice + Price ∗ Quantity
11:    ENDIF
12: ENDWHILE
```

The second algorithm is used to calculate the change when the customer gives money for the products they are buying. If the customer gives the right amount of money then no change is needed, however if the customer gives more than the total price of the products, then the following algorithm calculate how much change they need.

Algorithm 2 Calculating Change

```
1: Done ← False
2: OUTPUT "PleaseEnterMoneyGiven"
3: MoneyGiven ← USERINPUT
4: OUTPUT "PleaseentertthePrice
5: Price ← USERINPUT
6: IF MoneyGiven ¡ Price THEN
7:     OUTPUT "More Money Please"
8: ELSE IF MoneyGiven = Price THEN
9:     OUTPUT "Thank You!"
10: ELSE IF MoneyGiven ¿ Price THEN
11:    Change ← MoneyGiven − Price
12:    OUTPUT Change
13: ENDIF
```

These algorithms are for the stock control system. This IS the system I am changing. The first algorithm is used to add a new product to the system

Algorithm 3 Adding A New Product

```
ProductName ← NameOfProduct
ProductID ← Product_List[ProductID]
OUTPUT "Does the product have a specific weight"
Weight ← USERINPUT
IF Weight = "Yes" THEN
    OUTPUT Enter The Weight Of The Product
    WeightOfProduct ← USERINPUT
    ProductWeight ← WeightOfProduct
ENDIF
ProductQuantity ← QuantityOfProductInStock
ProductPrice ← PriceOfProduct
OUTPUT "Is The Price of the product been reduced (OnOffer)"
Reduction ← USERINPUT
IF Reduction > 0 THEN
    ProductPrice ← ProductPrice * (Reduction/100)
ENDIF
```

Algorithm 4 Categorizing A Product

```
OUTPUT "|DOG|CAT|BIRD|EQUINE|REPTILE|OTHER"
OUTPUT "Choose a Category in which the product comes under"
Category1 ← USERINPUT
IF Category1 = "DOG" THEN
    OUTPUT "|FOOD|HEALTHCARE|STESSANDANXIETY"
    OUTPUT "Choose a Category in which the product comes under"
    Category2 ← USERINPUT
ELSE IF Category2 = FOOD THEN
    OUTPUT "|WETFOOD|DRYFOOD|"
    OUTPUT Choose a Category in which the product comes under
    Category3 ← USERINPUT
    ProductCategory ← CategoryList[Category1][Category2][Category3]
ELSE IF Category2 = HEALTH CARE THEN
    OUTPUT "|ORAL|JOINTS|WORMING|EYE|SKIN|EAR|OTHER"
    OUTPUT Choose a Category in which the product comes under
    Category3 ← USERINPUT
    ProductCategory ← CategoryList[Category1][Category2][Category3]
ELSE IF Category = STESS AND ANXIETY THEN
    OUTPUT "|DIFFUSER|TABLETS|DROPS|"
    OUTPUT Choose a Category in which the product comes under
    Category3 ← USERINPUT
    ProductCategory ← CategoryList[Category1][Category2][Category3]
ENDIF
```

Algorithm 5 Moving Items From Storage to Front

1: $ProductID \leftarrow UniqueProductidentification$
2: **OUTPUT** *Please Enter Total Stock*
3: $TotalStock \leftarrow$ **USERINPUT**
4: **OUTPUT** "$PleaseentertheWantedAmountintheShop$
5: $MinimalAmount \leftarrow$ **USERINPUT**
6: **OUTPUT** *Please enter the current amount in the shop*
7: **IF** $ShopAmont < MinimalAmount$ **THEN**
8: **OUTPUT** " (MinimalAmont- ShopAmount) of ProductID need to be moved from the storage to the store"
9: **ENDIF**

Data flow diagrams

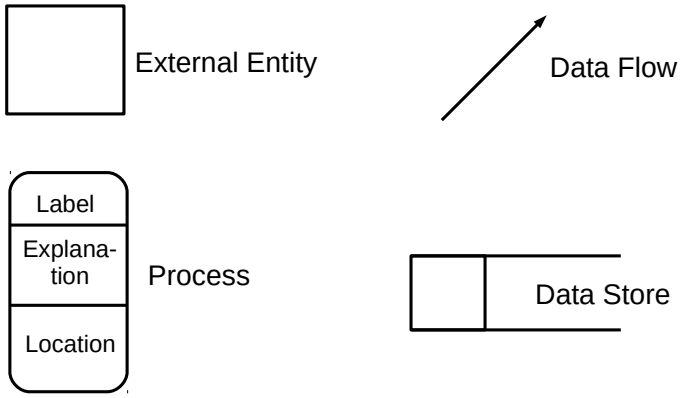

Figure 1.2: This is the Key For the Data Flow Diagrams.

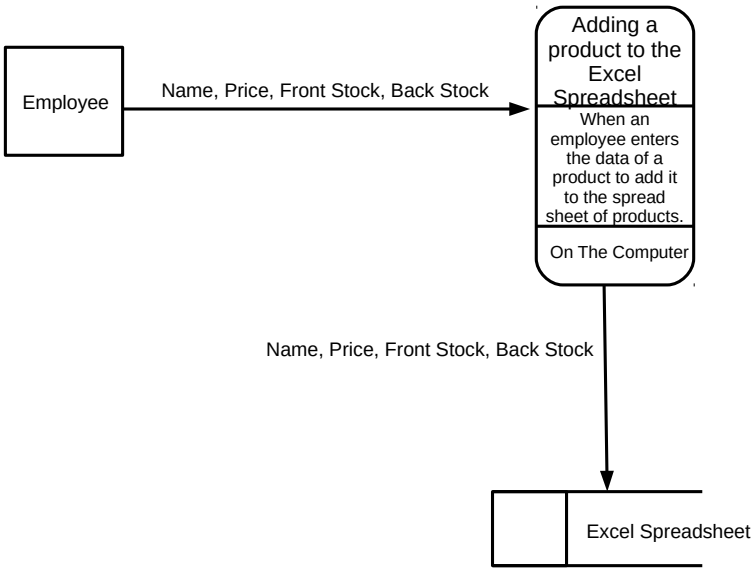

Figure 1.3: This is the Data Flow Diagram for an employee adding a product to the database.

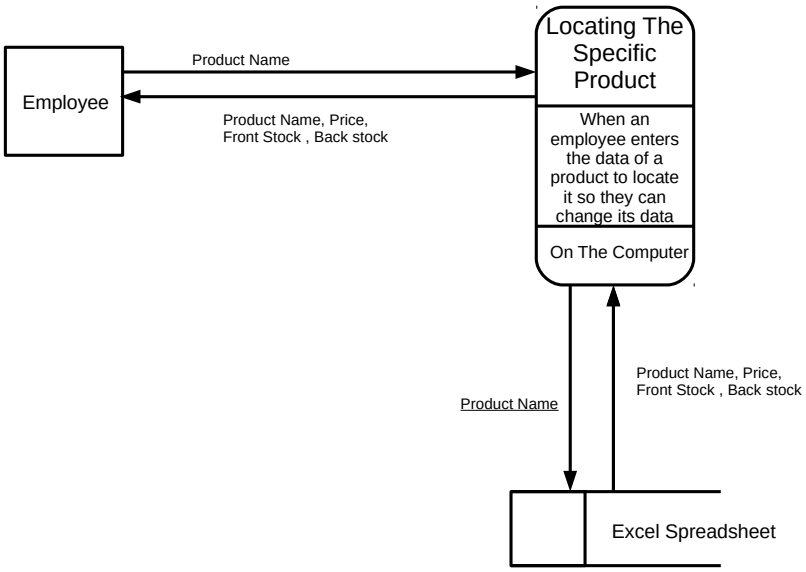

Figure 1.4: This is the Data Flow Diagram for a customer editing the data of an existing product..

Input Forms, Output Forms, Report Formats

The current system only has one input form - Adding a product to the database. when a product needs to be added to the database, the product name, quantity, price and category needs to be know in order for it to be added. The product ID is automatically assigned to the item when it is added to the database. Because items are only added to the database and the information is only stored, viewed and edited when needed, there are currently no output forms for the data. In the new system, the client wants the stock control system and the selling process to be integrated. When the new system is implemented then there would be one output form which would be a receipt given to the customer when they purchase a product. The receipt would contain the following information: The Name of the product, the quantity/weight/volume of the product the quantity and the price of the product. This is so the customer knows exactly what products they bought and know how much each product costs. The Product ID is not supplied on the receipt as that information is not relevant to the customer.

1.2.2 The proposed system

Data sources and destinations

There is only one data source in the current system. Because my client wants to integrate the selling process into the current system, There will before new data sources, giving five total data sources .The first data source is when a client purchases a product or many products. The Product name, weight, quantity are entered into the new system, or a search is made into the database for the product that the customer wants to buy, when all the attributes of the product match the ones of the product stored in the database, the price of that product is taken from the database and is added to the total bill. The information entered is repeated onto a receipt that is given to the customer to clarify which items they purchased. I am not storing the data for every purchase from a customer in the new system, as there is only limited storage space and storing all the data from every purchase will require an extremely large amount of storage space. The second data source is when a new product is added to the system. This data source was used in the old system. The information about the product will be stored in the database as it is used when a customer wants to purchase a product. This information is also used when the customer wants to find a product she does not know that name of. The third data source is when the information about a product needs to be updated or edited. This new data will override the current data stored under each product and will be held inside the database. The fourth data store, is when the stock needs to changed, when an import of new stock comes in. This data will also be held inside the database.

Another Data source will be when a customer supplies their MemberID if they are a member. if they are a member then they get a 10 percent discount, along with other advantages unrelated to this system. Therefore, the information about the Members of the veterinary will be stored within the database.

Data flow diagram

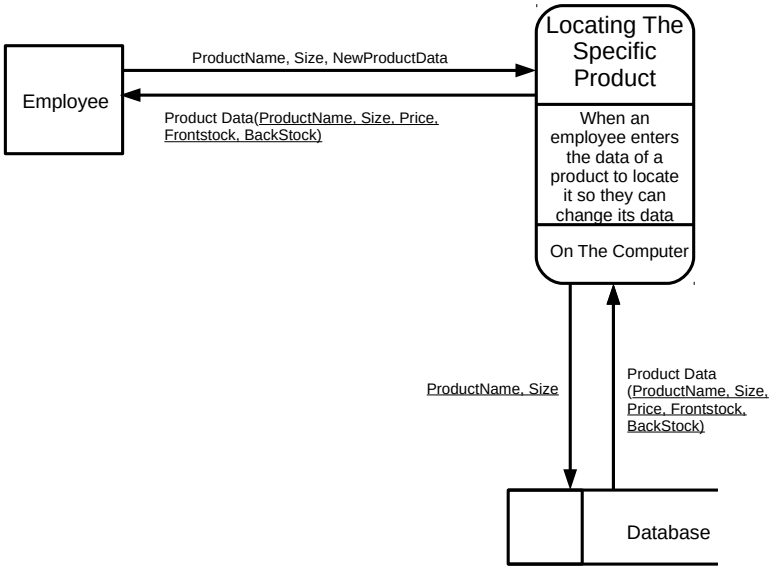

Figure 1.5: This is the Data Flow Diagram for an employee editing the data of an existing product.

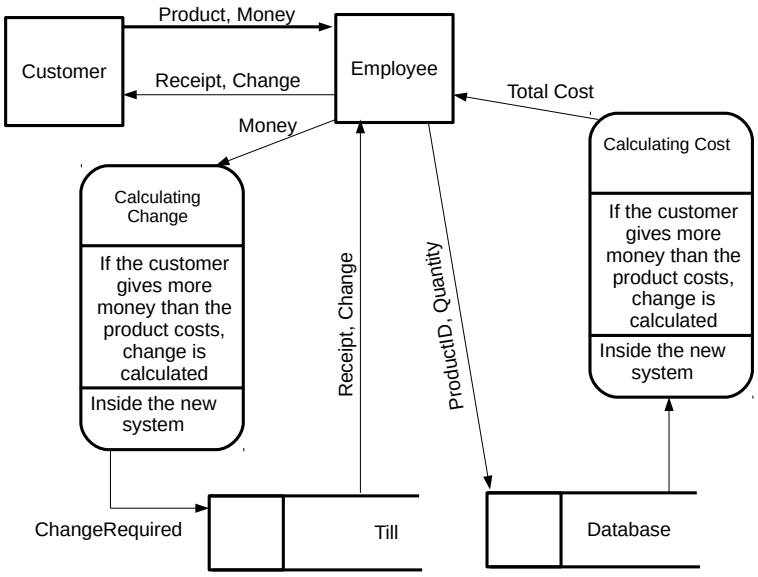

Figure 1.6: This is the Data Flow Diagram for a customer Buying A product

Data dictionary

Data	Data Type	Length	Validation	Example Data
ProductID	Integer	100	Range	42
ProductName	String	100	length	OPTEX Ear Drops
Size	Integer	1000	Range	750 grams
Price	Real	250.00	Range	9.99
Catagory	String	100	Length	Dog food
FrontStock	integer	50	Range	5
BackStock	Integer	50	Range	11
TotalStock	Real	100	Range	20
MoneyGiven	Real	200.00	Range	20.00
TotalPrice	Real	300.00	Range	20.00
ChangeRequired	Boolean		If Money ¿ total Price	True
Change	Real	20.00	If Change required = True	2.35
MemberID	Integer	300	Range	24
Firstname	String	50	Length	Thomas
LastName	String	50	Length	Brennan
AddressFirstLine	Float	50	Length	66 Market Street
AddressSecondLine	Float	50	Length	Fordham, Ely
TelephoneNumber	Integer	11	range	07577209094
Email	String	50	Length	Example@email.com

The ProductID in the new system will be created by the number that product is in the database. For example the first item to be added to the database will be assigned to ProductID = 1, the second item to be added ProductID =2 ect .... Therefore, the length of the ProductID is dependent on the number of products being stored within the database. Currently, my client stores the data of 57 products. He wishes to expand to about 75 and continue to expand on this as the company grows. As a result, I have decided to set the length of ProductID to 200. This is because I want to compensate for when the company grows in size and begins to sell a lot more products. I could have decided to fix the length to 100, however I feel that this limit could be reached quite quickly.

The length of the ProductName has been set to 100. This seems like far too many characters needed for just a name but sometimes, some of the products have very long names." Aristavet Panzym Concentrated Pancreatic Enzymes Powder" is an example of a product that has a very long name that is stored in the current database.

The Price has been set to a limit of 200. This seems quite high for a customer buying a few products but currently the most expensive product sold at the veterinary is"Protexin Professional Probiotics Sachets" which is £134.95.

I have decided to size 'size' to 1000. This seems very excessive, when it comes to quantities of products, however if the product is classified by its mass or its volume(i.e. 750g / 250ml ect . . .) then usually the mass / volume will reach 999 grams or millilitres. Usually when it reaches 1000, it is changed into kilograms / litres.

1 Litre = 1000 millilitres, 1 kilogram = 1000 grams.

Volumetrics

To calculate the **maximum** file size the system could be, i need to record the **maximum** amount of characters that arepossible to be stored within the system. (1 Byte is the amount of storage required to hold 1 character.)

Refer to Data Dictionary for the length of each piece of data

Maximum ProductID = 3 Characters (i.e 125) = **3 Bytes**

Maximum ProductName = 100 Characters = **100 Bytes**

Maximum Size = 4 Characters (i.e 750) = **4 Bytes**

Maximum Price = 6 characters (assuming the '.' counts as a character') = **6 bytes**

Maximum Catagory = 100 characters = **100 bytes**

Maximum FrontStock = 2 character (i.e 35) = **2 bytes**

Maximum BackStock = 2 character (i.e 21) = **2 bytes**

Maximum TotalStock = 3 characters (i.e 100) = **3 bytes**

Maximum MoneyGiven = 6 characters (assuming the '.' counts as a character') = **6 bytes**

Maximum TotalPrice = 6 characters (assuming the '.' counts as a character') = **6 bytes**

Maximum Change = 4 characters (assuming the '.' counts as a character') = **4 bytes**

Maximum MemberID = 3 characters = **3 bytes**

Maximum FirstName = 50 characters = **50 bytes**

Maximum LastName = 50 characters = **50 bytes**

Maximum AdressFirstLine = 50 characters = **50 bytes**

Maximum AdressSecondLine = 50 characters = **50 bytes**

Maximum TelphoneNumber = 11 characters (assuming its a real telephone number and also assuming no intenational numbers will be stored in the

database (this would require 12 characters, as the '0' at the start will have to change to +44)) = **11 bytes**

Maximum Email = 50 characters = **50 Bytes**

Therefore, the **maximum** space required to store one Product = 3 + 100 + 4 + 6 + 100 + 2 + 2 + 3 = **220 bytes**

The maximum amount of products my client is likely to store within the new system, so the **maximum** sorage space required to store 100 products = 100 * 220 bytes = **22,000 bytes = 22Kilobytes / 22Kb**

The Amount of storage space require to hold the information concerning 1 customer = 3 + 50 + 50 + 50 + 50 + 11 + 50 = **264 bytes**

The maximum amount of members my client expects to have is 100. therefore The **maximum** space required to store all the memebr data = 264 * 100 = **26,400 bytes = 26.4Kilbytes / 26.4Kb**

The Transactions made are not perminantly stored within the database, they are only temporarily used then errased. The maximum sotrage space required for a transaction = 6 + 6 + 4 = **16 bytes**

Therefore, the **maximum** storage space required (assuming the information about a transaction is currently being sotred) = 26,400 + 22,000 + 16 = **48,416 Bytes = 48.416 kilbytes / 48.416Kb**

Although it is only a very rough estimate, I used the file size of one of my previous long applications as an estimate to the file size required for the application. The file size of the application is **7,000 bytes = 7 Kilbytes / 7kb**

Finally, The **maximum** storage space, the new system could possibly need, is 48,416 + 7000 = **55416 bytes = 55.416 Kilbytes / 55.416Kb**

This amount of storage is extremely low compared to the amount of file storage available on my clients hard drive. This should mean, there should be no problem with the new system to do with not having enough memory to store the data in the database.

1.3 Objectives

1.3.1 General Objectives

- Data can be added to the database easily.
- Data can easily be edited within the database.
- Clear/ easy to understand the layout of the information

- Clear/ easy to locate a product within the database
- The system to be Clear/ easy to use.

1.3.2 Specific Objectives

- For an image to be displayed when an item is searched for (identifying a product if the Product information is unknown)
- For the Stock control system to be integrated with the process of selling items. This is so that the stock is automatically updated when a product is sold.
- For the new process of selling products to be quicker, to minimize the time for people queuing.
- For the current stock to automatically update when the products are bought
- For a reminder message to pop up when stock needs to be moved from storage to the front of the vets.
- For each and every product to be categorised for easy identification is the Product Name and ID is unknown.
- To calculate how much stock will be required for next month.
- For the MemberID to be entered and the identity of the client is confirmed to make sure they are a member.
- For Keyboard Shortcuts to be availble for the system to be acessed faster.
- To Format a well structured receipt that is easy for the customer to read and to understand
- allowing the Order of the products in the database to be changed.(i.e. Max - Min Price, A-Z ect . . .)

1.3.3 Core Objectives

- For the Stock control system to be integrated with the process of selling items. This is so that the stock is automatically updated when a product is sold.
- For the Stock to automatically update itself.
- To calculate how much stock will be required for next month.

1.3.4 Other Objectives

- allowing the Order of the products in the database to be changed.(i.e. Max - Min Price, A-Z ect . . .)
- For Keyboard Shortcuts to be availble for the system to be acessed faster.
- To Format a well structured receipt that is easy for the customer to read and to understand

1.4 ER Diagrams and Descriptions

1.4.1 ER Diagram

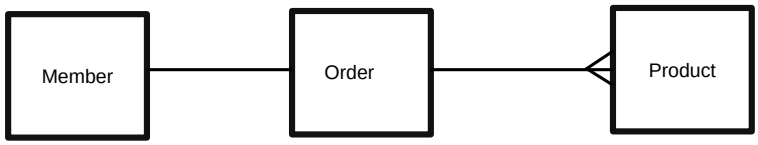

Figure 1.7: Entity Relationship Diagram.

1.4.2 Entity Descriptions

Product(ProductID, ProductName, Size, Price, Category, FrontStock, BackStock, TotalStock)

Member(MemberID, Firstname, LastName, AddressFirstLine, AddressSecondLine, TelephoneNumber, Email.)

Order(OrderID,*ProductID*, Quantity)

The relationship between order and Member is that a Member can make One Order. This order Can contain many prodcuts therefore, the relatyionship between Order and Product is one Order can have Many products. The ProductID is the primary key for Product, but is also the foreeign key for order as the ProductId is required to make an order.

1.5 Object Analysis

1.5.1 Object Listing

Product(All relevant information about that product, including a unique identifier for each product.)

Member(All relevant information about each Member, including a unique identifier for each Member so they can identified quickly...) Order(all relevent information about each order and the quantity of each product in that order)

1.5.2 Relationship diagrams

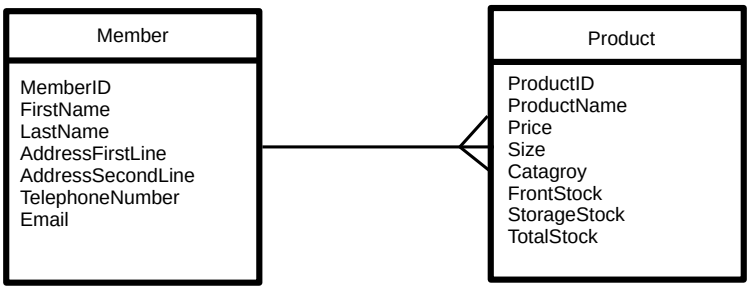

Figure 1.8: A Relationship Diagram between Member and Product.

1.5.3 Class definitions

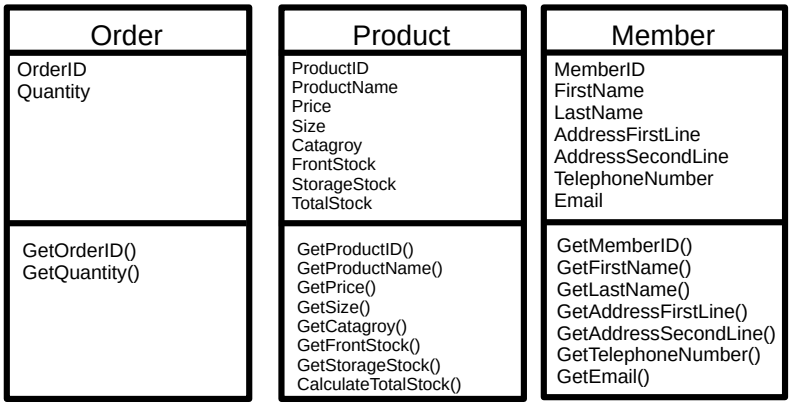

Figure 1.9: The Class Definition For Product and Member.

1.6 Other Abstractions and Graphs

There are no current graphs to draw for the system

1.7 Constraints

1.7.1 Hardware

Currently my client uses a Lenovo Desktop Pc for the current system. The computer is also used to check emails and to check appointment times with customers. The new system will be able to run on this machine.

Computer Specifications:

- 21" LCD Monitor
- Intel® Core™ i3-3240T processor

- 4 GB DDR3
- 1 TB HDD, 7200 rpm
- H61 Motherboard

The demand of the purposed system will make little to no impact on the Computers speed. The requirements for the purposed system is far less than the current hardware can provide meaning there will be no problem running the new system on this computer.

Although the proposed system will be able to run on this computer there are a few hardware constraints. One of which is that the fact this computer is a desktop computer. This means that my client will not be able to move this system and will have to keep the computer in one place. This is because the desktop computer does not have an internally stored battery.

Another Hardware constraint is that the new system will have to be designed and implemented to fit the screen resolution of my clients computer. par

1.7.2 Software

My client currently uses Microsoft Excel as his system. Tom is happy to use any other software. Because tom takes full ownership of the computer he has no limitations on what he would like to install. for example if the computer belonged to someone else, the user of the computer would want as little changes as possible. Tom's computer currently runs Windows 7 Home Edition, which the new system will be able to run on.

1.7.3 Time

My client has set no deadline or time limit for the completion of the system. The only time restriction I have on this project is the deadline set by my teacher, which is the 1st May 2015. Tom says he is happy to start using the new system when it is ready. The sooner it is finished the better.

1.7.4 User Knowledge

At this current time tom does not have any qualifications in ICT and has not taken any IT courses. Despite this, Tom has a quite good knowledge when it comes computers, Gaining most of his knowledge whilst doing his Veterinary Medicine degree. He has helped devolve web pages, and is currently helping create a new website for Beacon Vets. However Tom informed me that he would not be the only one to be using the system, It will also be used by the receptionists when customers buy products. This means that The interface has to be simple and easy to use.

1.7.5 Access restrictions

There proposed system should be accessible to anyone that works at beacon vets. Every employee will have full access/privileges to all of the data on the system. The computer will be password protected along with the system to prevent unwanted users on the computer to access the database, and potentially erase or edit all of the data within the database.

Because the database stores personal information about people, the system has to follow to Data Protection Act. Tom and all the employees at Beacon Vets will have to be informed of this law in hope they do not break it.

1.8 Limitations

1.8.1 Areas which will not be included in computerisation

Areas that shall not be included in computerisation is the exchange of money when products are brought. Currently, The vets allows two method of payment. The two methods of payment are by cash or by Credit / Debit card. The proposed system will only handle information from customers that pay by cash.(i.e. Calculating change ect . . .) The proposed system will not handle data from customers who pay by card because that information is handled externally by an external card reader, which will be too complicated to integrate into my new system. The process of moving products form storage to the front of the vets obviously will not be computerised. This will be done externally, and a tick box will be supplied by the system for the employee to tick when the products have been moved. The limitation with this is that the system will assume the employee has moved the correct amount of products. When adding a new product to the system the data will be input manually. Although this is not part of the system, teaching Tom how to use the new system will not be computerised. Tom would then have to use the knowledge I have taught him and teach all of the employees at Beacon Vets.

1.8.2 Areas considered for future computerisation

An area that could be computerised is teaching tom / the employees, on how to use the new system. Written instructions could be processed into a document for a employee to refer to if they need to do something, but do not currently know how. This will mean that Tom will not have to take time out explaining how to use the system to each and every employee. He can send an email with the instructions attached to it for each of the employees to read and the instructions document could be easily accessible in the system if they forget / don't know how to do something. The process of moving products

using a computer is not possible. The process of selling product can be computerised by creating a page on the website where customers can buy the products online. This could be possible as the location of Beacon Vets (Silloth) is a small town and is mainly only used by the local community. This is useful because it means that product deliveries can be made within a maximum of 10 miles from the store. The limitations with computerising this method is that creating a web page like this is that I do not currently have the skill sets to create a webpage like this. The webpage will have to be secure, and will have to have things such as PayPal incorporated into it. To prevent the customers from having to enter their card details in every time they make a purchase, their card details will have to be stored on the system. This will then mean that the system will have to abide by the data protection act, however I feel this information can be stored securely over PayPal, which will reduce the problems faced when creating the system.

1.9 Solutions

1.9.1 Alternative solutions

Solution	Advantages	Disadvantages
Custom Spread Sheet	My client already uses a spreadsheet software called "Microsoft Excel", so no training is required. No new software will have to be downloaded.	my client uses spreadsheet software for his current system. He explains that the system is easy and simple to mange with a few products but as time went by and more products were added it because difficult, unordered and confusing.
Web Based Application	the information stored in the database can be stored in the cloud. This would reduce the file size of the system. The system can be accessed on more than one computer and no information has to be installed.	Data is not as secure if it is stored in the cloud. It is possible for someone to intrude into your database and modify the data. This would be less likely if the data was stored on a hard drive as the information could only be accessed on that one computer in which the hard drive is in. The system would have to constantly be online, meaning that there would have to be more security measures to prevent data theft. Hosting the web application will be more expensive than having a program. A lot more knowledge in building the web based system compared to using a program.
Re-organising/ re designing the current manual system	Would take a small amount of time to create. No New software will need to be installed. No new skills will have to be taught to the employees.	The problem my client currently has may not be overcome. Very simple system that is hard to change.
Command-line Application	Easier to design and program compared to a gui application. often runs faster than any gui application. I believe less computer resources would be used compared to using a GUI application	Command line applications are not user friendly when sued by people with little computer knowledge. Many of the employees will have had little / no training using command line applications. A lot of error checking will have to be implemented so that the user does not ac-

1.9.2 Justification of chosen solution

I have chosen to use a "Python Desktop Application including a GUI" for my system, this is because:

1. I Already know the Python programming language, which will reduce time in me having to learn how to create the system.
2. I know a very limited amount of HTML and don't have any knowledge on any other web based languages, so I would have to take a lot of time out to learn about how to use HTML to create the Web based application and how to integrate it with things such as PayPal and The Cloud ect ...
3. I can use a tool called 'Git' to back up my work so if I do have a problem, I can just roll back to my last saved piece of work. Git also allows me to see the changes I have made since I last updated my system so I can see where the errors have been created if they occur.
4. The new system will be simple and easy to use which will mean my client and his employees will not have to spend much time learning how to use the new system.
5. Using a Python application with a GUI is a lot easier to use compared to a command line application where my client and his employees will have to learn commands to use in the system.

Chapter 2

Design

2.1 Overall System Design

2.1.1 Short description of the main parts of the system

2.1.2 System flowcharts showing an overview of the complete system

2.2 User Interface Designs

2.3 Program Structure

2.3.1 Top-down design structure charts

2.3.2 Algorithms in pseudo-code for each data transformation process

2.3.3 Object Diagrams

2.3.4 Class Definitions

2.4 Prototyping

2.5 Definition of Data Requirements

2.5.1 Identification of all data input items

2.5.2 Identification of all data output items

2.5.3 Explanation of how data output items are generated

2.5.4 Data Dictionary

2.5.5 Identification of appropriate storage media

2.9.1 Outline Plan

Test Series	Purpose of Test Series	Testing Strategy	Strategy Rationale
Example	Example	Example	Example

2.9.2 Detailed Plan

Test Series	Purpose of Test	Test Description	Test Data	Test Data Type (Normal/ Erroneous/ Boundary)	Expected Result	Actual Result	Evidence
Example	Example	Example	Example	Example	Example	Example	Example

Chapter 3

Testing

3.1 Test Plan

3.1.1 Original Outline Plan

Test Series	Purpose of Test Series	Testing Strategy	Strategy Rationale
Example	Example	Example	Example

3.1.2 Changes to Outline Plan

Test Series	Purpose of Test Series	Testing Strategy	Strategy Rationale
Example	Example	Example	Example

3.1.3 Original Detailed Plan

Test Series	Purpose of Test	Test Description	Test Data	Test Data Type (Normal/ Erroneous/ Boundary)	Expected Result	Actual Result	Evidence
Example	Example	Example	Example	Example	Example	Example	Example

3.1.4 Changes to Detailed Plan

Test Series	Purpose of Test	Test Description	Test Data	Test Data Type (Normal/ Erroneous/ Boundary)	Expected Result	Actual Result	Evidence
Example	Example	Example	Example	Example	Example	Example	Example

3.2 Test Data

3.2.1 Original Test Data

3.2.2 Changes to Test Data

3.3 Annotated Samples

3.3.1 Actual Results

3.3.2 Evidence

3.4 Evaluation

3.4.1 Approach to Testing

3.4.2 Problems Encountered

3.4.3 Strengths of Testing

3.4.4 Weaknesses of Testing

3.4.5 Reliability of Application

3.4.6 Robustness of Application

Chapter 4

System Maintenance

4.1 Environment

4.1.1 Software

4.1.2 Usage Explanation

4.1.3 Features Used

4.2 System Overview

4.2.1 System Component

4.3 Code Structure

4.3.1 Particular Code Section

4.4 Variable Listing

4.5 System Evidence

4.5.1 User Interface

4.5.2 ER Diagram

4.5.3 Database Table Views

4.5.4 Database SQL

4.5.5 SQL Queries

4.6 Testing

4.6.1 Summary of Results

4.10.1 Module 1

Chapter 5

User Manual

5.1 Introduction

5.2 Installation

5.2.1 Prerequisite Installation

Installing Python

Installing PyQt

Etc.

5.2.2 System Installation

5.2.3 Running the System

5.3 Tutorial

5.3.1 Introduction

5.3.2 Assumptions

5.3.3 Tutorial Questions

Question 1

Question 2

5.3.4 Saving

5.3.5 Limitations

5.4 Error Recovery

Chapter 6

Evaluation

6.1 Customer Requirements

6.1.1 Objective Evaluation

6.2 Effectiveness

6.2.1 Objective Evaluation

6.3 Learnability

6.4 Usability

6.5 Maintainability

6.6 Suggestions for Improvement

6.7 End User Evidence

6.7.1 Questionnaires

6.7.2 Graphs

6.7.3 Written Statements