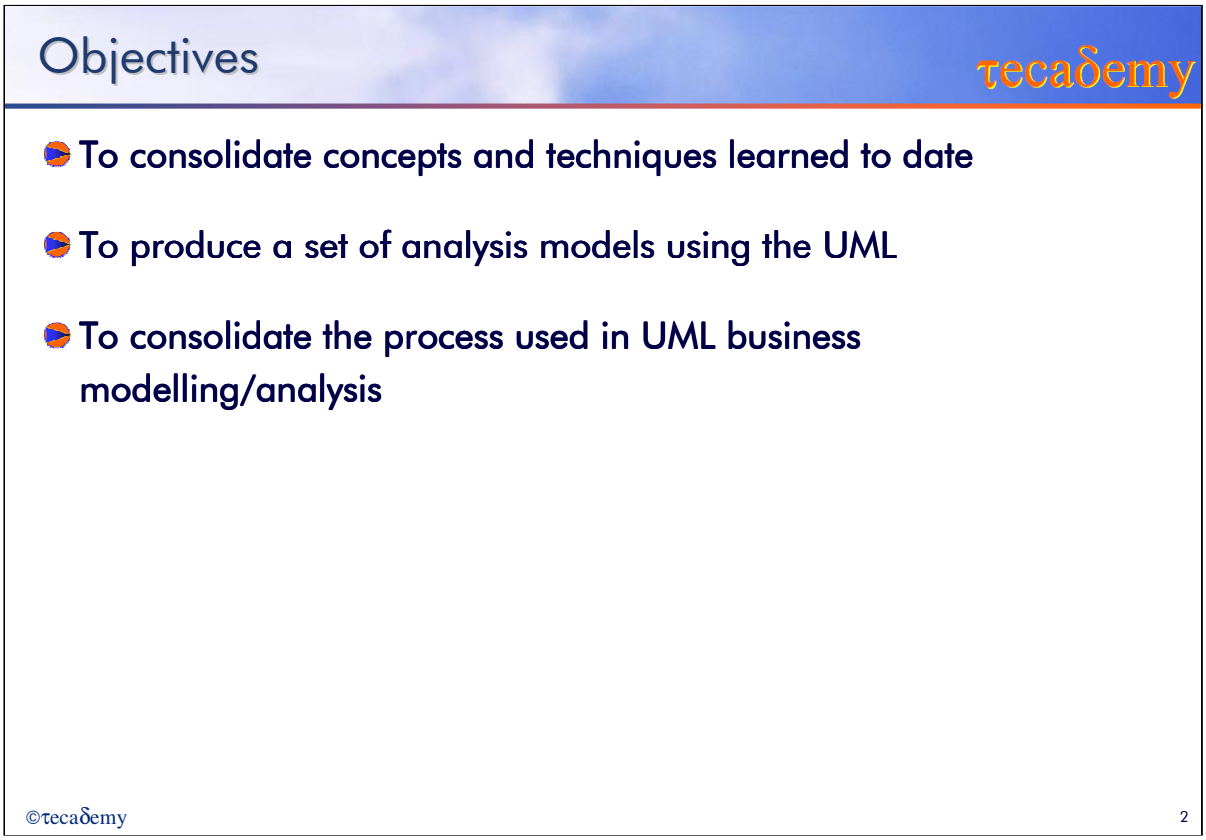
Case Study

τecaδemy


# Sid's Supermarket



1




In this exercise, we will analyse a system that manages the inventory and sales processes of a supermarket.

The exercise is organised into a sequence of activities that follows an idealised path through the life cycle – i.e. we have no time for iterations! Complete as much of each exercise as you can in the time given, which will never be enough. There is no correct solution, but there is a suggested solution against which you can compare your own, and other syndicates will be comparing theirs with yours. Your job as an analyst is to challenge and compare possible solutions, and either pick the one that seems most useful or synthesis a new solution from the best bits of each alternative (this is the most sensible approach). At each stage, you have the prerogative of choosing which solution to progress.

There will be no need for you to proceed on the basis of any assumptions – as indeed you shouldn't. This exercise does not provide you with all the information you need to complete the exercise – one of its aims is to encourage the black art of knowledge elicitation. There will be two customer representatives from whom you can get information: **Sid**, who is the supermarket manager, and **Dwayne**, who operates one of the checkouts. **Your Instructor** is a RUP/UML consultant who has no domain expertise but can help you with any modelling issues. Please prefix each question with the name of the person to whom it is addressed. Answers will be honest but not necessarily correct or consistent. *C'est la vie*.

Start by reading the context statement, ask any questions you feel necessary, and move to the first exercise. And…

… *have fun!*

## Exercise 1 – Scoping

τecaδemy

- Read the statement on the following page.
- Draw a Business Context Diagram (BCD) showing the supermarket business and the entities (actors) with which it has *direct* interaction. Show each functional component of the business and the interfaces between them, clearly identifying and describing the actors and major business processes.
- Construct a glossary. Make sure each model element carries a glossary definition.
- Establish the scope of the problem with respect to the rest of the exercise. Hint: it will be a subset of the BCD.

30 mins





Don't forget – if you're modelling without involving your customer, you're NOT modelling – you're guessing.


***Exercise 1 – Read this…***

**Sid's Supermarket**
**Context Statement**

The inventory processing capabilities allow the store manager and the inventory manager to access the product database. The store manager sets the prices for products and the inventory manager adjusts the stock levels of items as goods are delivered.

The sales processing governs the sales transactions for a customer at each of the checkout registers of the supermarket. A cashier is identified to the system for as long as they are operating the register. A checkout session is the processing of all the transactions for a single customer. Each item (or quantity of the same item at the same time) purchased by the customer is considered a single transaction. When a bar code, UPC (Universal Product Code), or internal code number is input, a coded transaction is initiated. When a cashier types in a unit price and a category, a price-entry transaction is initiated.

A cashier uses a scanner to read a product's bar code and convert it to a product code number. If this operation is successful, the scanner flashes a green light. If the scanner cannot read the bar code, the scanner flashes a red light. The cashier may enter information manually through the keyboard. A UPC or internal code number may be entered manually if the scanner cannot read the bar code, or for products such as loose fruit that do not have a bar code.

If the customer is purchasing more than one of the same product, the cashier may enter the item quantity followed by the UPC or internal product code. The system must keep track of how many of the products are purchased. For example, if the price is 3 for £1.00, the charge should be 35p for the first and second items and 30p for the third. Variations are possible: a product could be priced at 40p each, Buy One Get One Free (BOGOF). The charge should be 40p for the first item and 0p for the second. It makes no difference whether the items are scanned together or interspersed with other products.

For products such as loose fruits that need to be weighed, a scale is attached to and read by the cash register. The price is determined by multiplying the measured weight by the price per unit weight.

If the UPC or internal code number is unavailable or does not exist for a product, the cashier may enter the unit price for the product, the number of items (optional), and the product category (non-taxable grocery, taxable grocery, meat, dairy, etc.).

At the end of the checkout session, the total bill is computed (including any VAT), cash payment is taken, change is computed and the cash drawer opened. A receipt is given to the customer with their change.

## Exercise 2 – Activity Diagram

τecaδemy

- Draw an activity diagram showing the end-to-end process of handling one customer's purchases.
- Use swimlanes to allocate different parts of the process to the appropriate entities.

15 mins

## Exercise 3 – Actors and Use Cases

τecaδemy

- Assuming the scope will be restricted to the *checkout* system (that is, the system as used by Dwayne) draw a use case diagram showing the checkout system and identify the entities (actors) with which it has *direct* interaction. Add the actor definitions to your glossary. Identify each actor as *primary* or *secondary* with respect to the use case.
- Identify opportunities for using «*includes*», «*extends*» and *generalisation* dependencies.
- Write a full description of one use case, including pre- and postconditions and at least one alternate flow.

40 mins




Remember – it's not daft to ask questions. But it is certainly daft *not* to.

## Exercise 4 – 'Candidate' Classes

τecaδemy

- Use the information from previous exercises to construct a list of candidate classes to be used when writing scenarios.
- To each candidate class, try to assign a stereotype.
- Include each candidate class, with description, in your glossary.

20 mins

## Exercise 5 – Scenarios and Sequence Diagrams

τecaδemy

- Using nouns drawn from your list of candidate classes, construct sequence diagrams for the following scenarios:
  - A checkout session consisting of one successfully scanned item
  - A checkout session consisting of one item with an unreadable bar code
  - A checkout session consisting of one successfully scanned item and one item sold by weight
  - A checkout session consisting of a price entry transaction and a number of identical items having a readable bar code
- Remember – One '*Subject-Verb-Object*' clause per line. Look for sub-scenarios that can be re-used.
- If time – do the same for the other scenarios.

45 mins

 8

**Example badly written scenario: A correctly scanned item**

Cashier hits RESET key. *(on what? On the Cash Register. Remember – Subject-Verb-Object)*

Cash Register resets for new customer, zeros display, and notifies Checkout Session that a new session is beginning. *(avoid 'And'. One line per event)*

Cashier scans one item. *(on what?)*

Cash Register reads bar code, converts it to UPC number, and flashes green light. *(three events here…)*

Cash Register passes UPC code to Checkout Session. *(much better!)*

Checkout Session computes price, determines category, and adds to total. *(what's wrong with this and subsequent lines….? Correct and re-write)*

Checkout Session passes price to Cash Register and it is displayed.

Cashier hits PAYMENT key.

Cash Register notifies Checkout Session that total bill is needed.

Checkout Session computes total bill and notifies the Cash Register which displays it.

Cashier enters amount of payment and hits ENTER key.

Cash Register sends amount of payment to Checkout Session.

Checkout Session computes amount of change owed and sends to Cash Register.

Cash Register displays amount of change and unlocks cash drawer.

**Tips:**

Include the following objects in your scenarios:

Cashier (it is usual to include the actor in early scenarios)

Cash Register

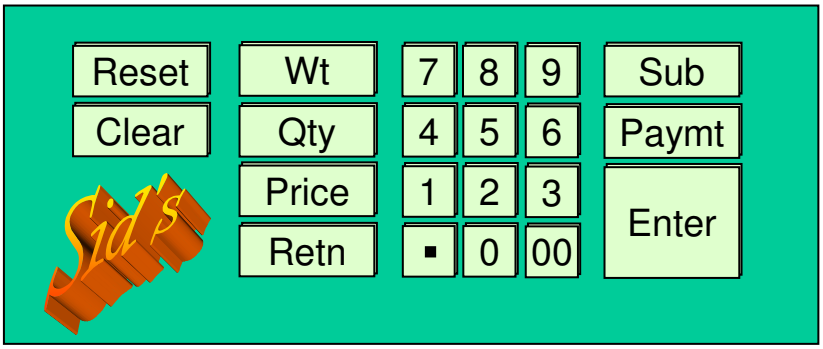
Checkout Session

Product

Assume that the scanner, light, scale, display etc are part of the cash register and would be included in more detailed scenarios not addressed in this part of the exercise. Similarly, the checkout session object contains transactions and totalling functions.

Assume that the cash register works by first hitting a function key, entering the appropriate digits, and then the ENTER key. In addition to the numeric keypad, the following function keys are available on the cash register:

| | |
|---|---|
| **RESET** | prepare for a new customer |
| **WEIGHT** | tells Cash Register to read Scale |
| **PRICE** | next input is manually entered price |
| **QTY** | next input is number of identical items |
| **SUBTOTAL** | calculates total cost of goods processed so far |
| **CLEAR** | clears the transaction in process |
| **PAYMENT** | no more items for this customer |
| **ENTER** | submits keyboard entry |




**Additional stuff for the interested:**

Refine your model to cover these additional scenarios:

A checkout session consisting of one manually entered invalid UPC number

A checkout session consisting of multiple manually entered UPC numbers, both valid and invalid

A checkout session consisting of a quantity of coded items (manually entered)

A checkout session consisting of a quantity of uncoded (price entry) items

A checkout session consisting of a manually entered quantity and a scanned in product

A checkout session consisting of a weighed uncoded item

Can you think of any other scenarios?

## Exercise 6 – More Sequence Diagrams

τecaδemy

- Using the sequence diagrams from the previous exercise, identify sub-scenarios from which other scenarios could be constructed.
- Extend these to show the interactions between the actor, cash register components (scanner, keyboard, display etc) and checkout session components in more detail.
- Use notation to distinguish between *message* events and *response* events.

30 mins



10

In the last exercise, we viewed the cash register as being the sum of its parts. This was so we could identify the interface between the cash register and the world without getting trapped in detail. In this exercise, we are going to 'explode' the cash register into its components, and show how they interact to provide the behaviour of the cash rgister as a whole. We will then turn our attention to the checkout session and do the same thing, breaking it down into transactions and so on.

The end result will be a sequence diagram showing most of the classes in our model.

## Exercise 7 – The Class Diagram

τecaδemy

- Use your sequence diagram(s) to construct a class diagram showing classes, attributes, operations, associations/aggregations with multiplicity, and generalisations (if any).
- Show only that which can be supported by the information you have elicited.
- Be prepared to present and argue for your model.
- If time: partition your class diagram into packages and draw a package diagram showing dependencies

20 mins

## Exercise 8 – The State Diagram

τecaδemy

- Use the four sequence diagrams from exercise 5 to construct a state diagram for the Cash Register showing states, transitions and activities/actions.
- Show only that which can be supported by the information you have elicited.
- Be prepared to present and argue for your model.

## Exercise 9 - Packages

τecaδemy

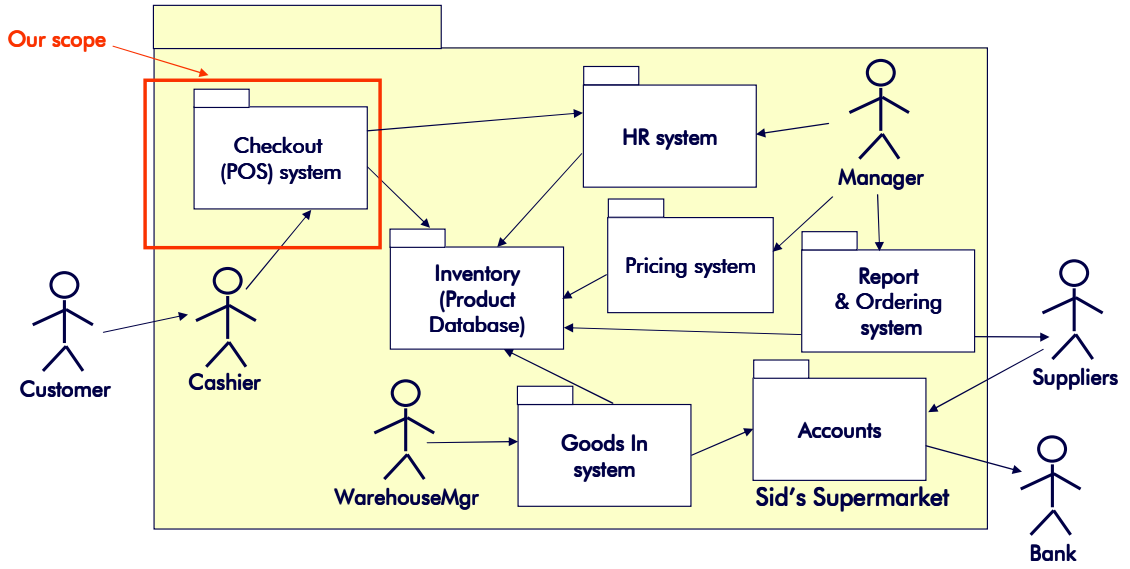
- Partition the class diagram from exercise 7 into packages.
- Draw a package diagram showing dependencies.

## Some Suggested Solutions

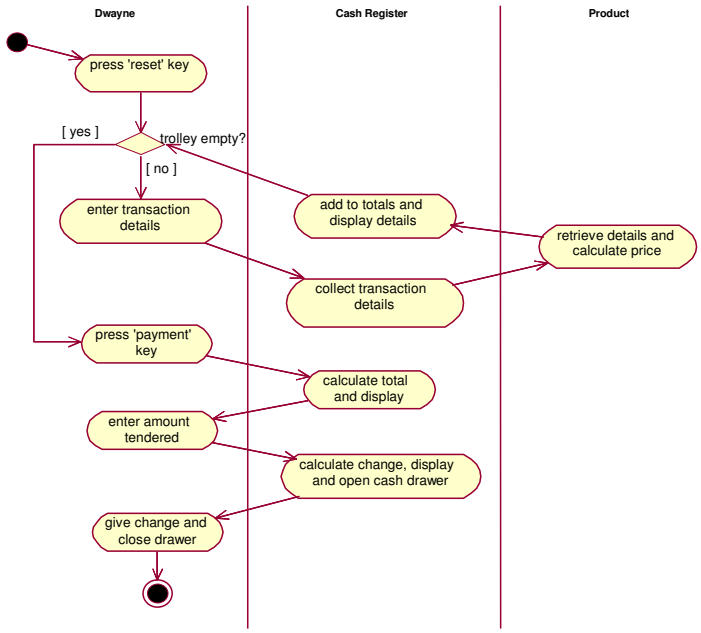
τecaδemy

- These 'solutions' are NOT definitive!
- The best solutions probably lie somewhere between these and those produced by you.
- Use these to promote discussion.

## Solution 1 – Scoping

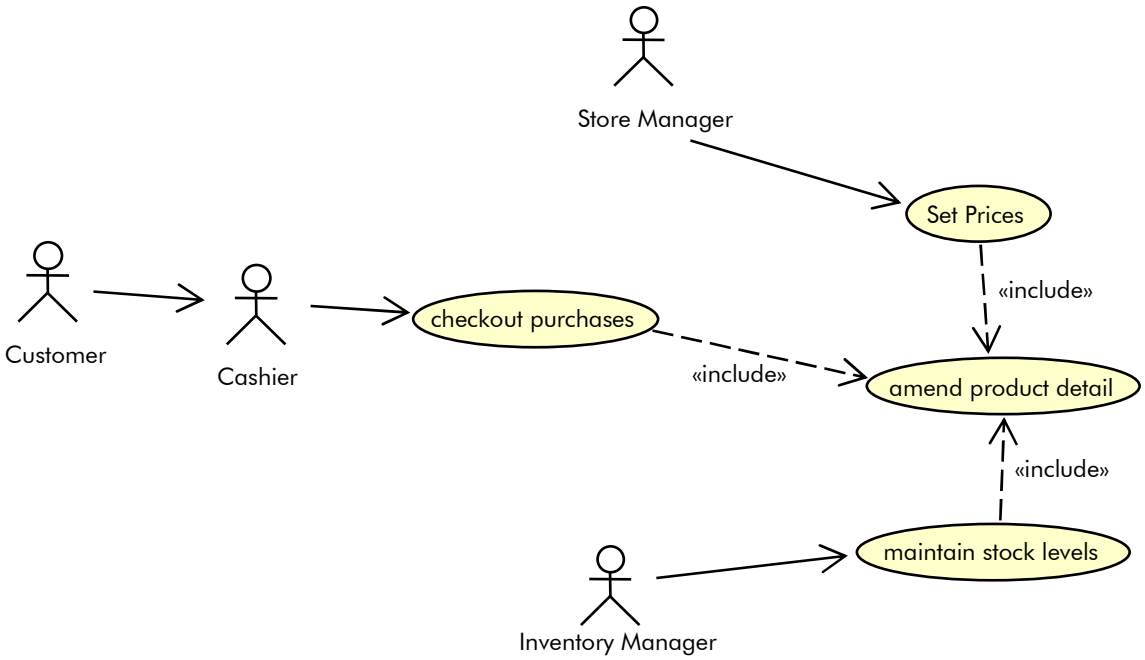


# Solution 2 – Activity Diagram

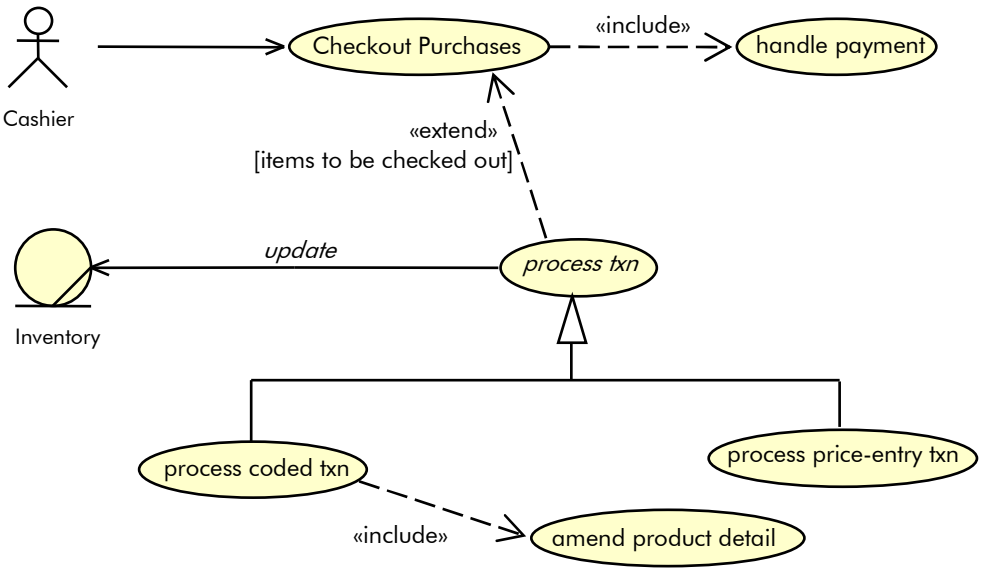


**Solution 2 – Actors and Use Cases**

Use Case Diagram 1



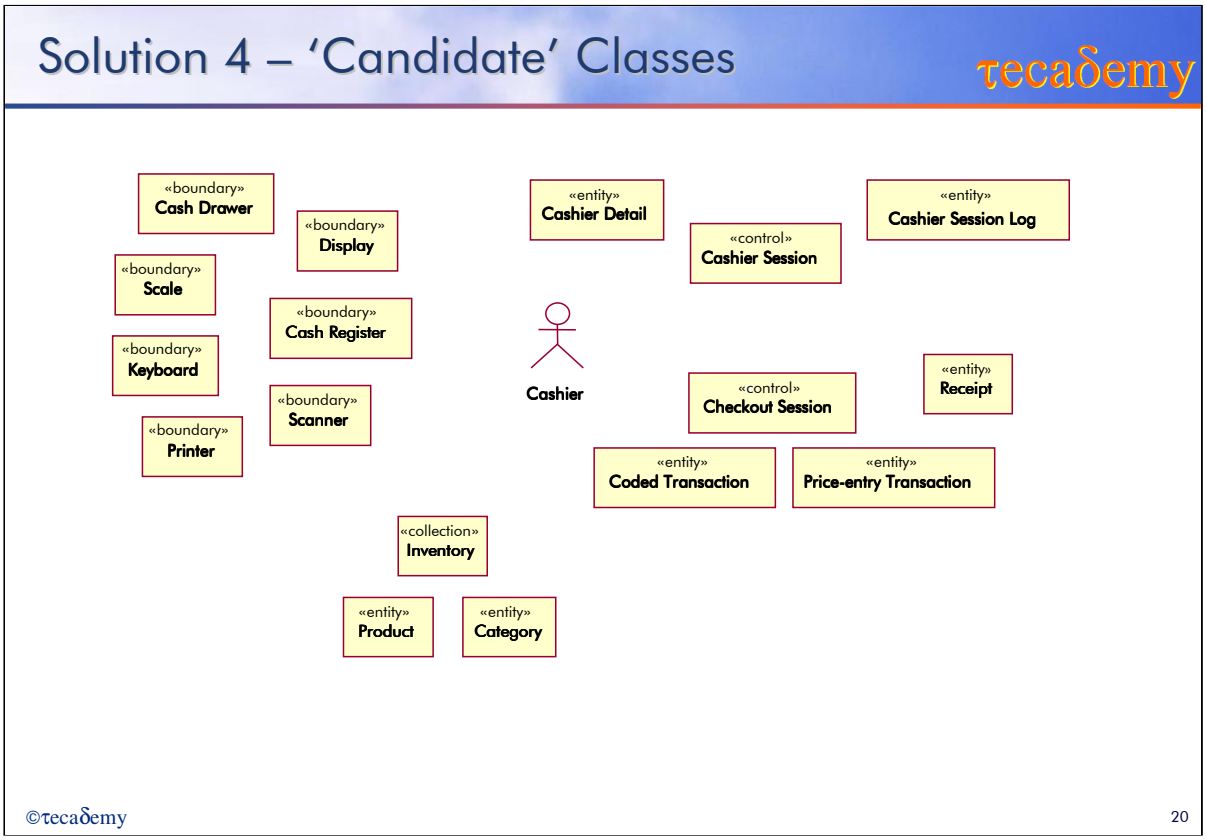

Use Case Diagram 2




| | |
|---|---|
| **Use Case:** | Checkout Purchases |
| **Summary:** | A user performs checkout of purchases, for a customer, by using the Supermarket Information System (SIS) |
| **Actors:** | Cashier |
| **Assumptions:** | System handles no more than 20 checkouts at the same time. There is no intersection of checkout sessions on the same register. |
| **Preconditions:** | An individual has entered the SIS as a Cashier. The SIS has checked that this individual has Cashier privileges and therefore may checkout purchases. |
| **Description:** | The checkout register is in a reset mode; that is, ready for a checkout session for a new customer. The cashier begins the checkout of purchases for a customer by initiating a transaction, either Price-Entry or Coded, for the first product.<br>1. Coded Transaction:<br>a: Scanned Bar Code: The Cashier swipes the product's bar code over the scanner, which converts the bar code to an UPC (Universal Product Code) number. If this operation is successful, the scanner flashes a green light. If not successful, the scanner flashes a red light; the Cashier may retry the bar code read operation until it is successful. [*Exception: The scanner cannot convert the bar code to an UPC number.*] If the customer is purchasing more that one item of the same product, the Cashier may enter the number of the items and then swipe the product's bar code over the scanner. The Cashier then processes another transaction, if not the last.<br>b: Manual Code Entry: The Cashier may manually enter either the UPC number or the Internal Code Number through the keyboard, if the bar code cannot be read or for products that do not have a bar code. [*Exception: The SIS responds that the UPC number or Internal Code Number is not valid.*] If the customer is purchasing more than one item of the same product, the Cashier may enter the number of items and then enter either the UPC number or Internal Code Number for the next item, whether a type of Coded or Priced-Entry Transaction.<br>2. Price-Entry Transaction:<br>The Cashier manually enters the number of items (optional), product's unit price, and product category if the UPC number or Internal Code Number is invalid or unavailable. For products that must be weighed, the Cashier weighs the product and then enters the price per unit weight and product category on the keyboard. The Cashier then processes another transaction, if any.<br>At any time in a checkout session, the Cashier may press the *SUB* key to receive the total price of purchases to that point. After the completion of the transaction for the last product, the Cashier presses the *TOTAL* key to compute the total bill (including VAT as applicable) for the customer's purchases. The Cashier enters the cash payment (other forms are outside the scope of this problem) accepted from the customer, to compute the change due to the customer. The Cashier then gives the customer change and a receipt. |
| **Exceptions:** | Exception: *The scanner cannot convert the bar code to an UPC number.* The cashier may enter either the UPC number or the Internal Code Number manually through the keyboard.<br>Exception: *The customer requests a cancel.* The Cashier cancels the checkout session and the Use Case is terminated.<br>Exception: *The SIS responds that the UPC number or Internal Code Number are not valid.* The Cashier must enter the unit price, number of items (optional), and the product category manually through the keyboard. |
| **Postconditions:** | The system is ready for another iteration. |


| | |
|---|---|
| **Use Case:** | Get Product Data |
| **Summary** | A user requires data on supermarket products in the course of fulfilling the role of either Store Manager (update prices), Inventory Manager (update inventory), or Cashier (checkout purchases). The Supermarket Information System (SIS) provides this product data to the user. |
| **Actors:** | Store Manager, Inventory Manager, and Cashier |
| **Preconditions:** | The user is in the course of fulfilling the role of Store Manager, Inventory Manager, or Cashier and requires supermarket product data. |
| **Description:** | For example, during the course of updating the inventory, the Inventory Manager requests product data by UPC/internal code number or by selecting from a list. The SIS retrieves this information from the product database and returns it to the update inventory process. [*Exception: The requested product data does not exist or cannot be retrieved.*] Product data is retrieved similarly for requests made by the Store Manager and Cashier. |
| **Exceptions:** | *Exception: The requested product data does not exist or cannot be retrieved.* The SIS returns an appropriate error message and advises the user to check with the system administrator or supervisor |
| **Postconditions:** | The user returns to the next sequential operation in the starting Use Case (either Update Prices, Update Inventory, or Checkout Purchases), with product data as required. Otherwise, none (*CANCEL* selected or attempt failed). |

**Supplementary question:**
Could the use cases given above be improved? If yes, then how?




**Cashier** «actor» A person who will process and take payment for a customer's purchases.

**Cashier Detail** «entity» Information relating to an authorised user of a cash register.

**Inventory** «collection» Place where information relating to products is kept.

**Product** «entity» Information relating to a coded line of merchandise (i.e. identified by a UPC or internal code number) stocked by a store e.g. Diet Pepsi 330ml, Niblets Sweetcorn Kernels 350g etc.....

**Category** «entity» Information relating to an arbitrary meta-classification of products e.g. taxable grocery, non-taxable grocery, meat, fish etc...

**Coded Transaction** «entity» Information relating to one purchase of a coded item.

**Price-entry Transaction** «entity» Information relating to one purchase of an uncoded item.

**Cashier Session Log** «entity» Information relating to the activities pursued by a Cashier while logged into the system.

**Receipt** «entity» Record of the purchases and money exchanges made by one customer. Alternative name - Checkout Session Log.

**Checkout Session** «control» Manages the processing of one customer's purchases

**Cashier Session** «control» Manages the activities of a Cashier using the system.

**Cash Register** «boundary» Device for recording sales information.

**Scanner** «boundary» Device for translating barcodes into character-based information. Flashes green light if successful, red if not.

**Keyboard** «boundary» Data entry device dependent on digital dexterity to depress buttons representing sales-specific symbolic constructs.

**Scale** «boundary» Device for measuring the gravitational attraction between the item being purchased and the Earth.

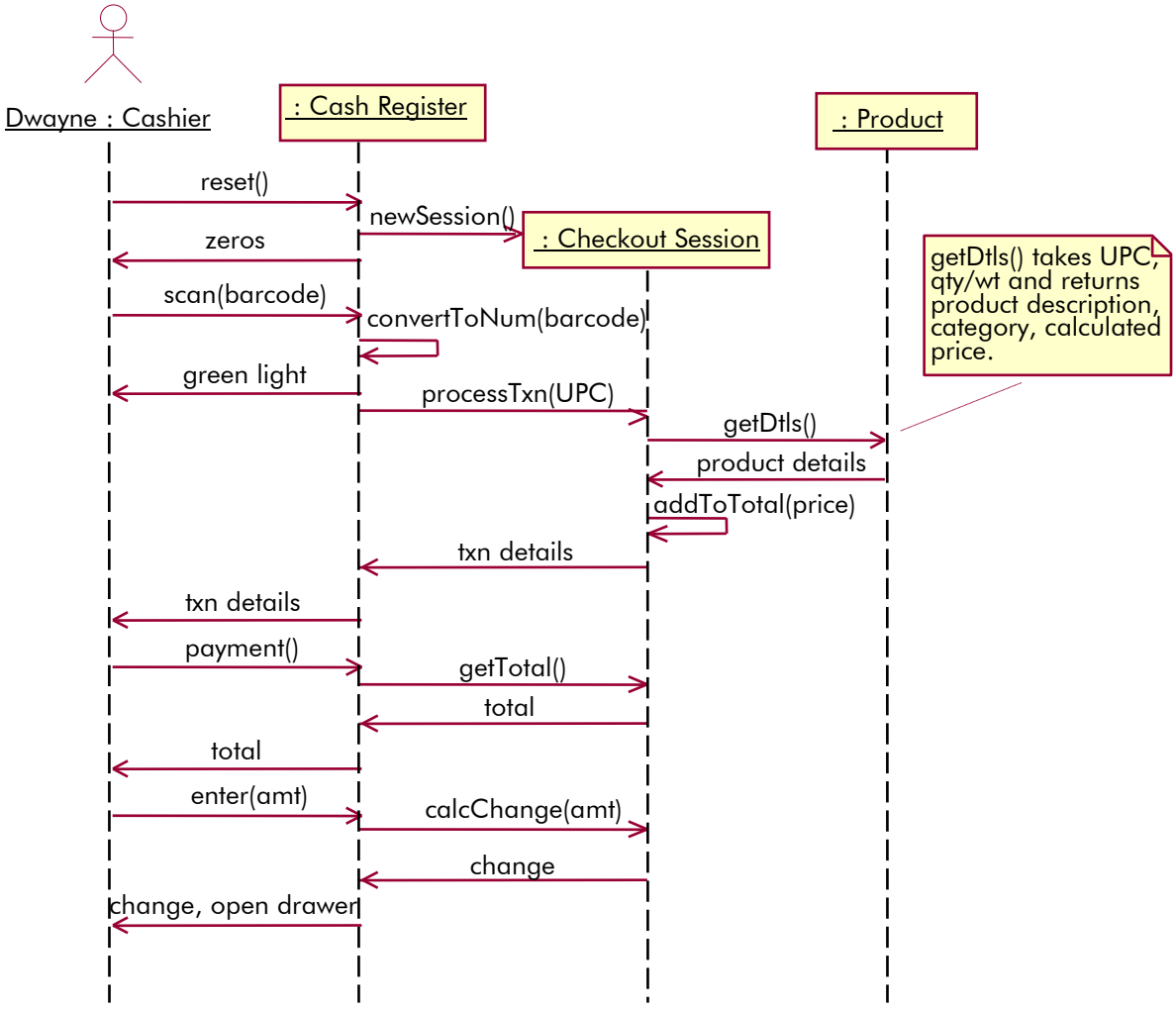
**Display** «boundary» Device for presenting information in human-readable form.

**Printer** «boundary» Device for printing information relating to Cash Register activity.

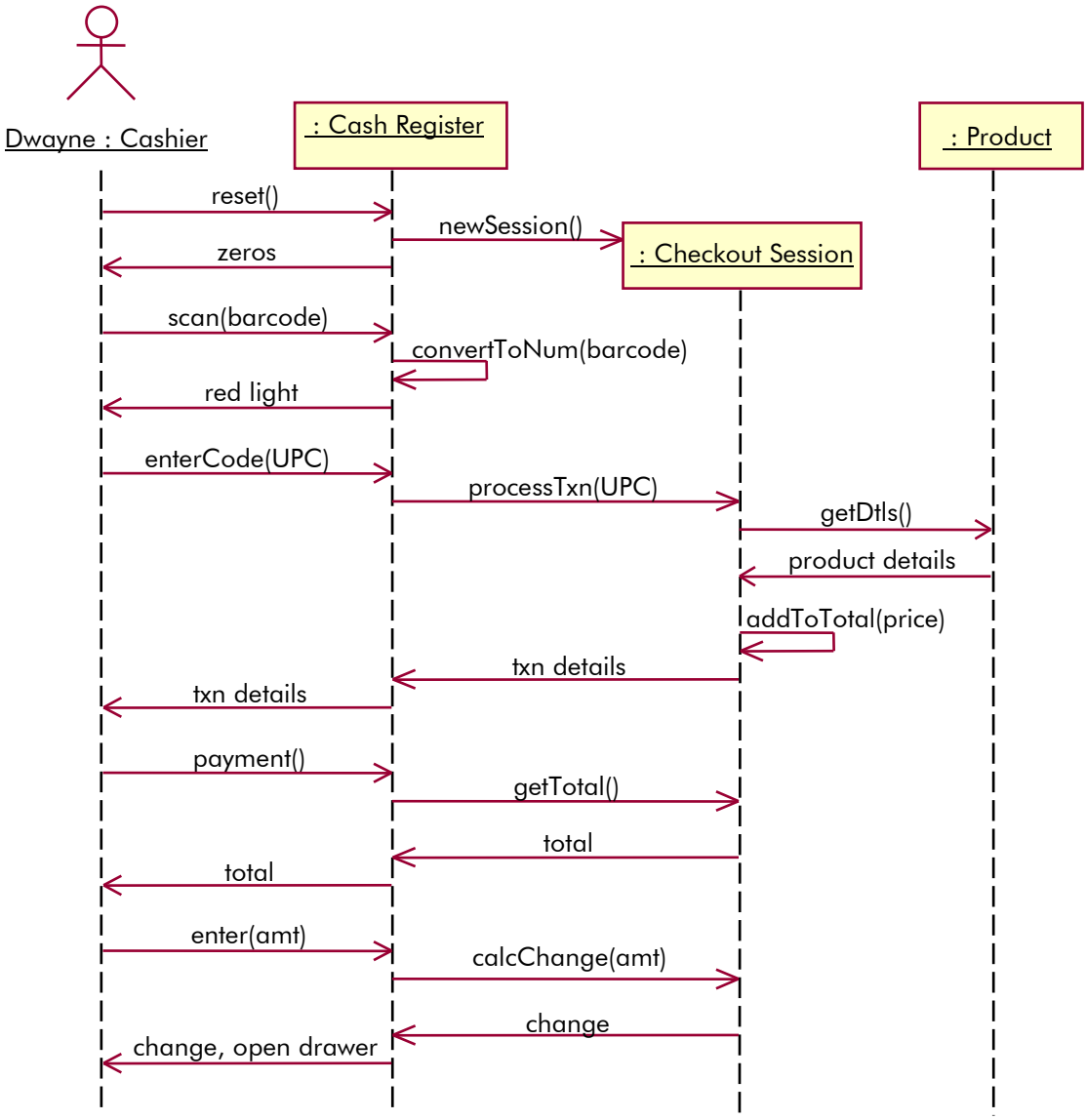
**Cash Drawer** «boundary» A box for containing money that can be opened and closed.

**Solution 5 – Scenarios and Sequence Diagrams**

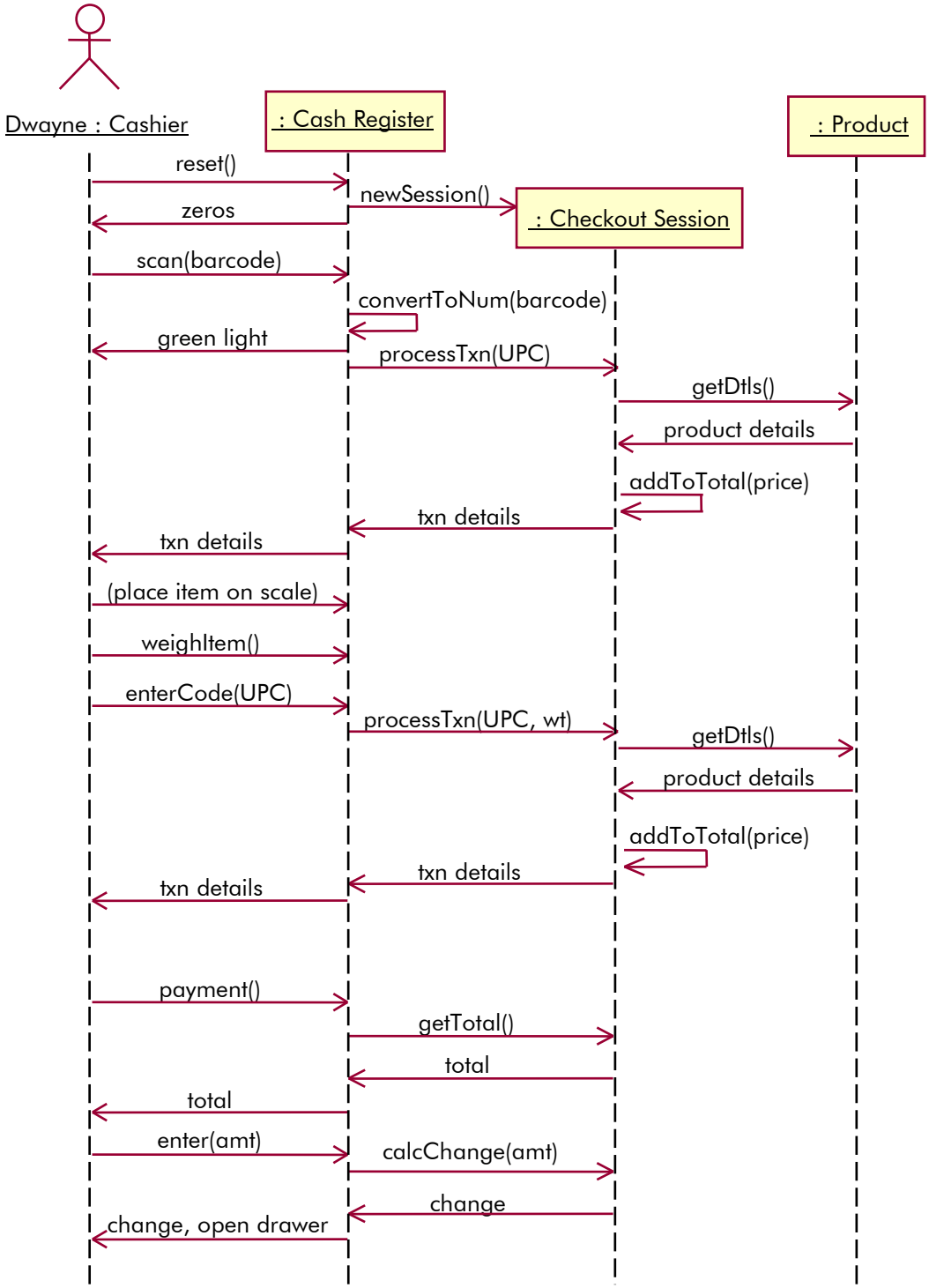
**Sequence Diagram 1:** A checkout session consisting of one successfully scanned item



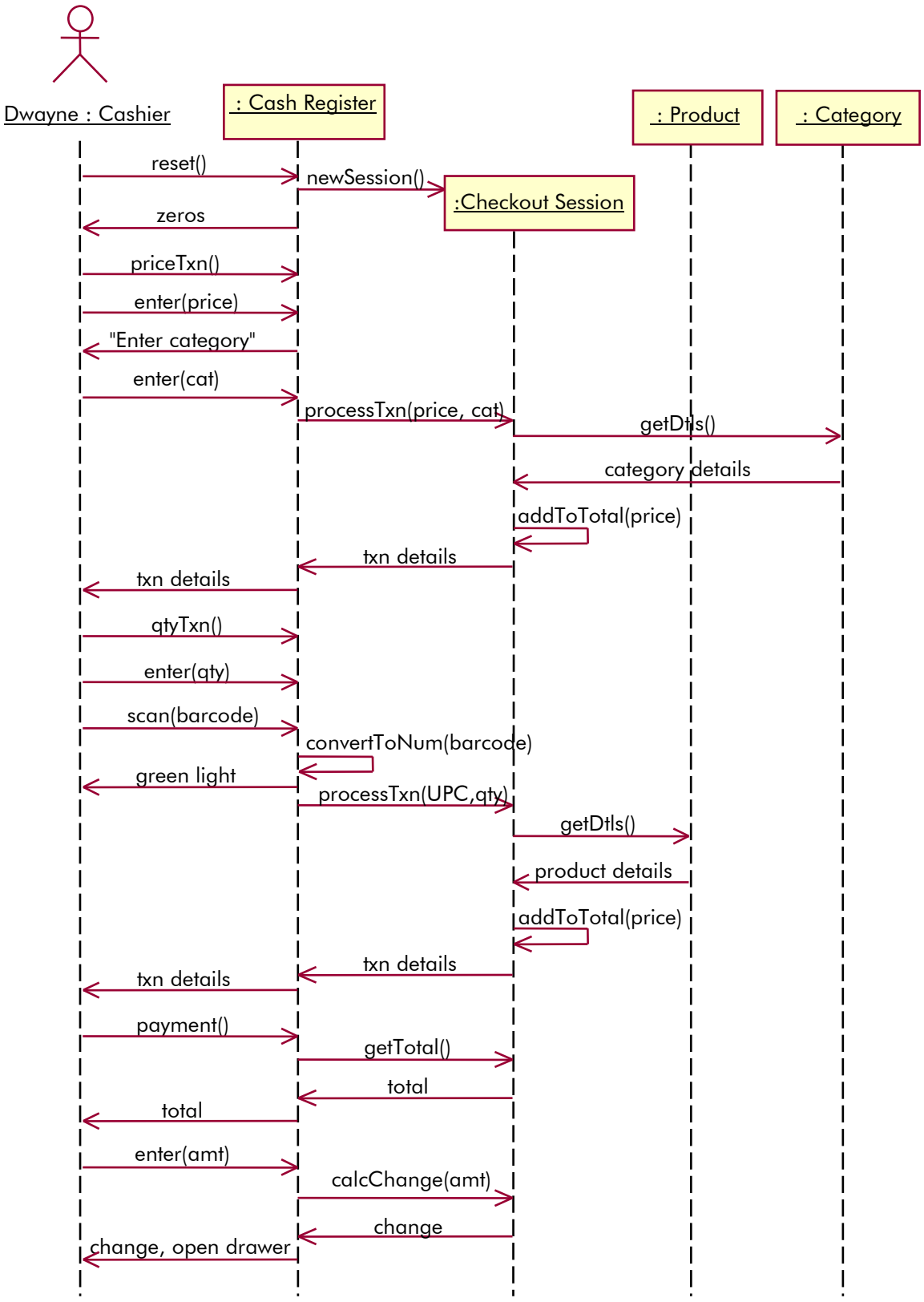
**Sequence Diagram 2:** A checkout session consisting of one item with an unreadable bar code



**Sequence Diagram 3:** A checkout session consisting of one successfully scanned item and one item sold by weight

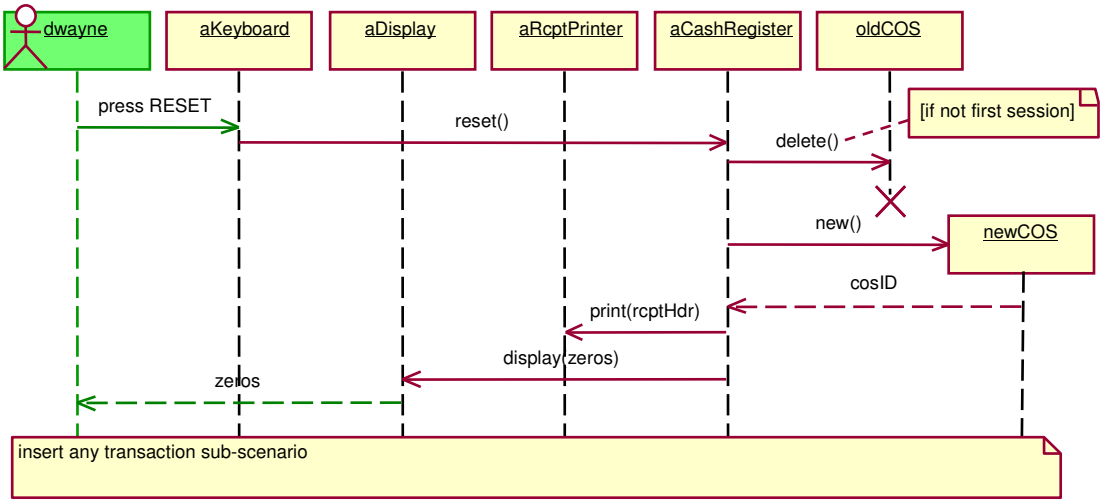


**Sequence Diagram 4:** A checkout session consisting of a price entry transaction and a number of identical items having a readable bar code

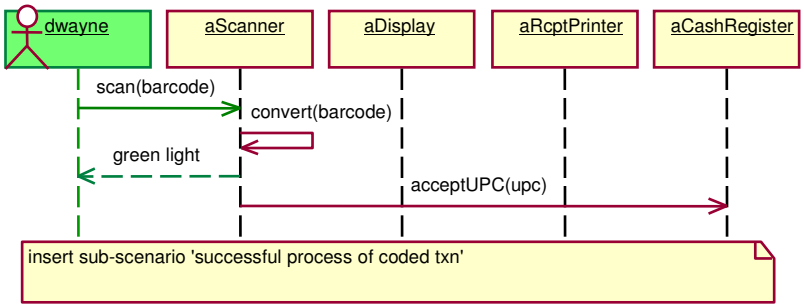


## Solution 6 – More Sequence Diagrams

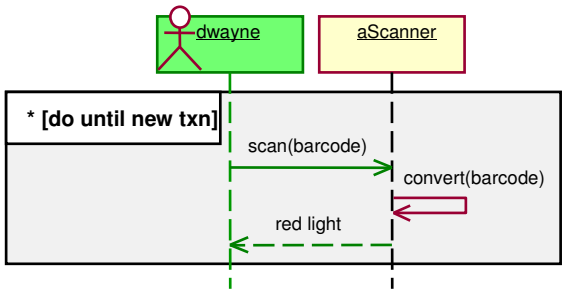
### Sub-scenario 'reset for new customer'



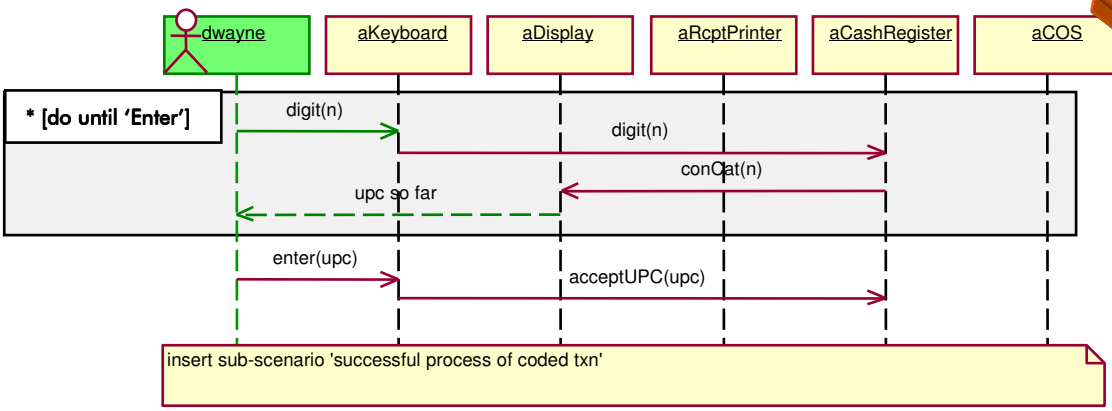

### Sub-scenario 'successful scan'



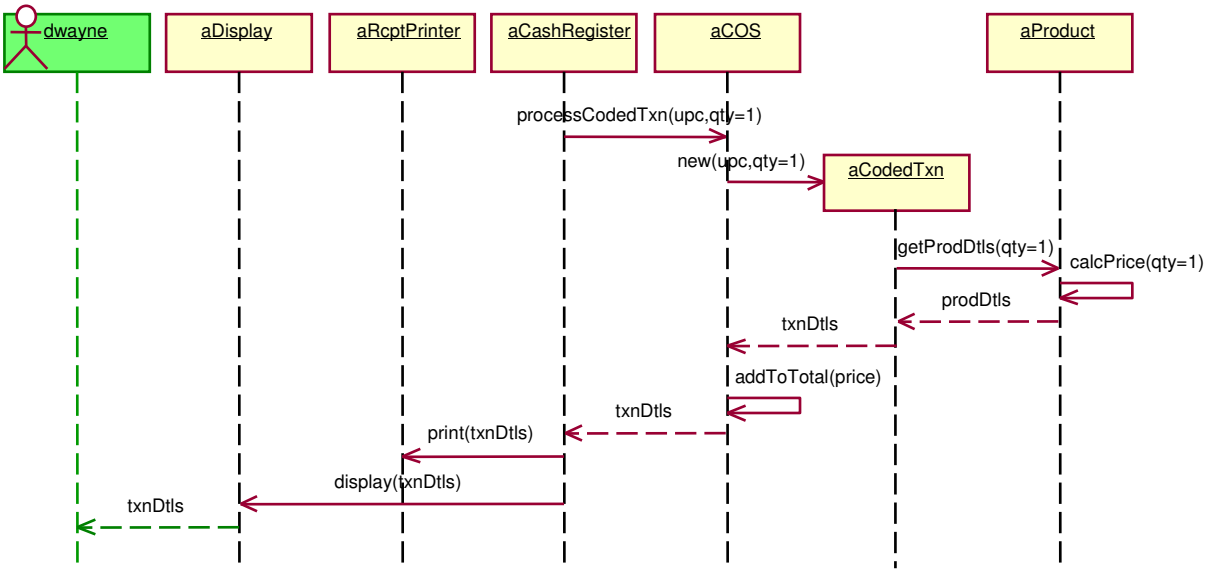

### Sub-scenario 'unsuccessful scan (barcode unreadable)'



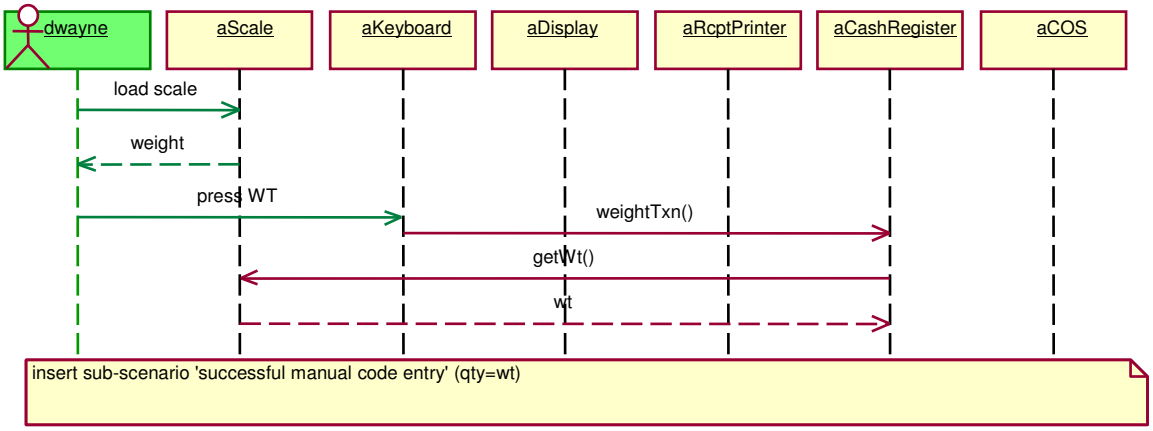
### Sub-scenario 'successful manual code entry'



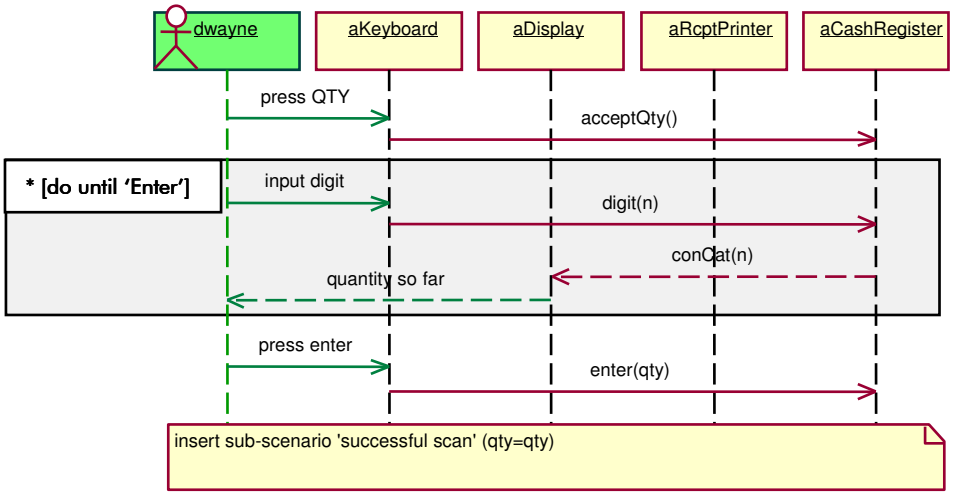

### Sub-scenario 'successful process of coded txn'



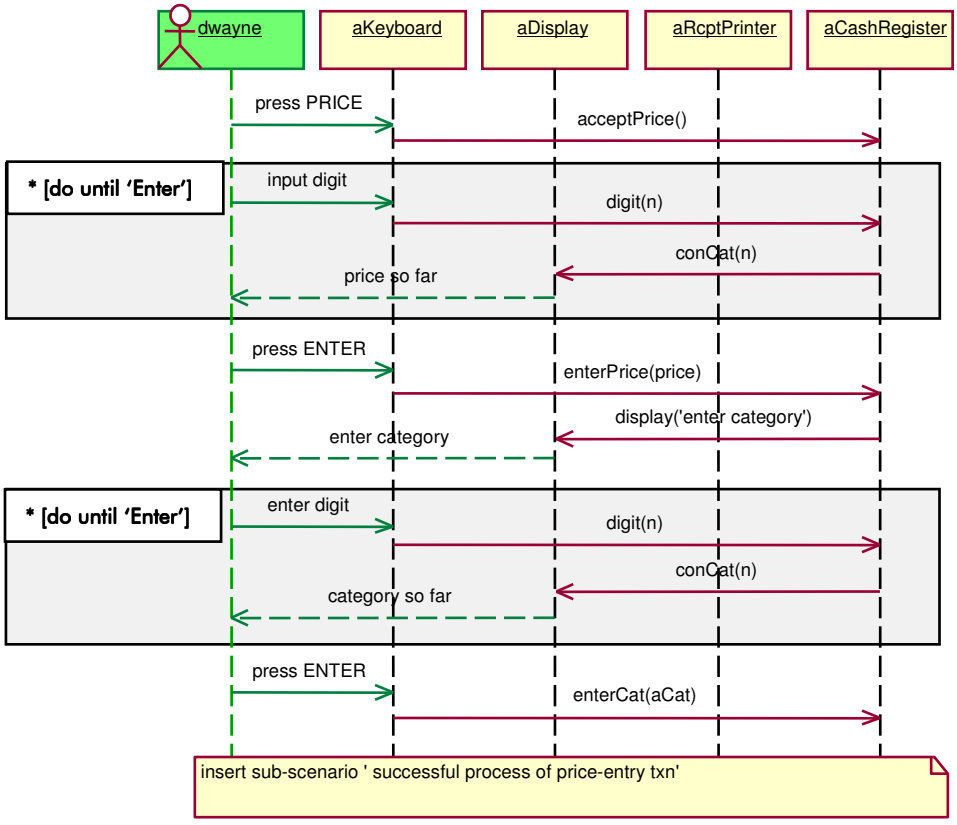

### Sub-scenario 'successful process of weighed txn'



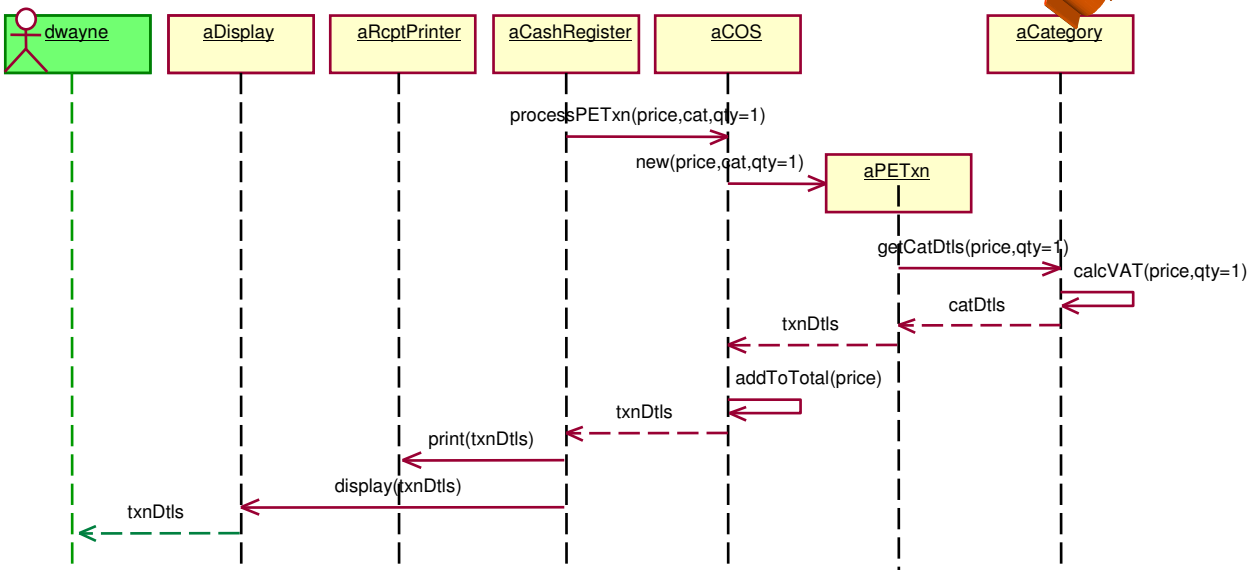
### Sub-scenario 'successful process of quantity of identical items, one scan'



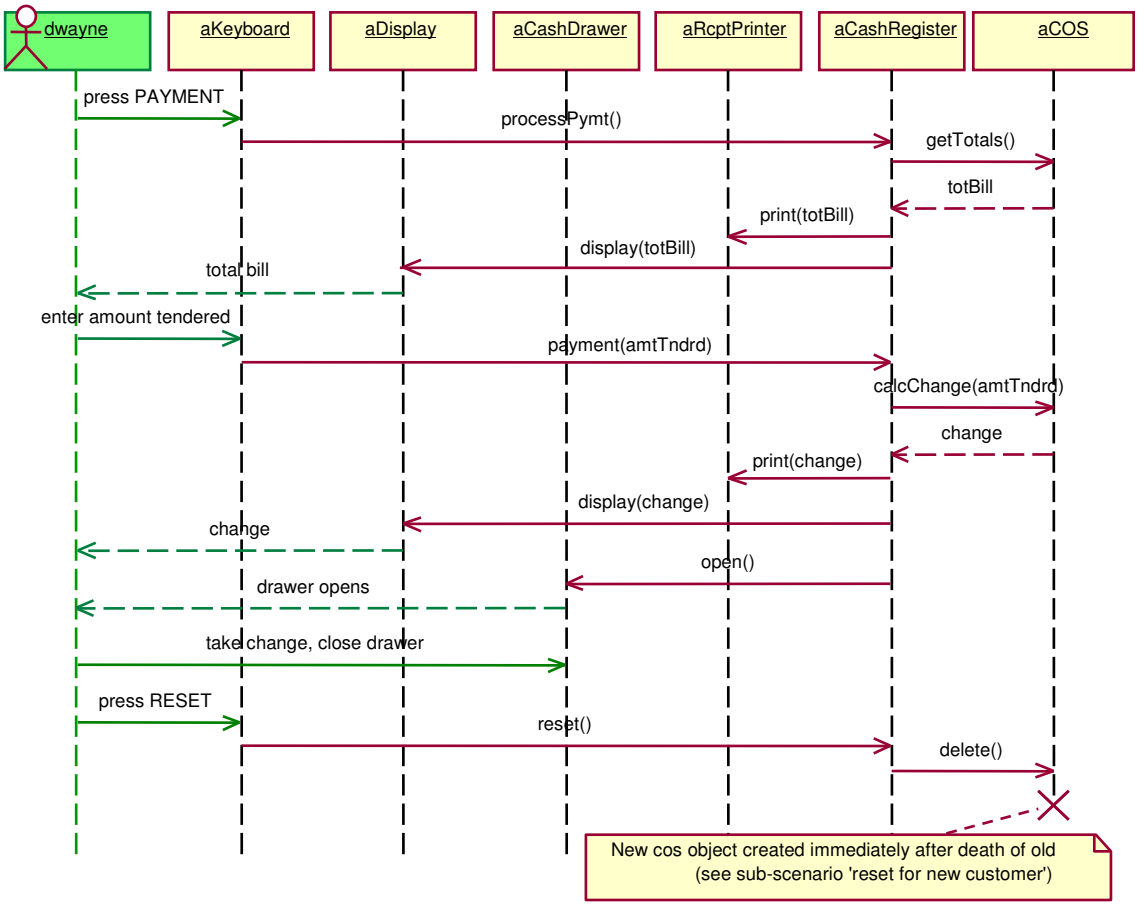

### Sub-scenario 'successful process of price-entry and category'



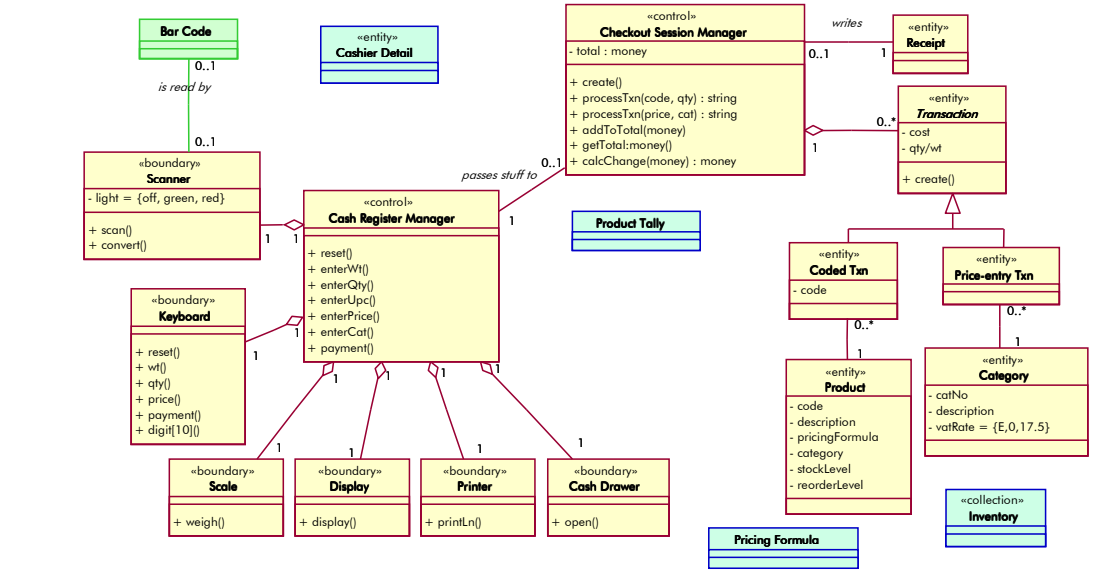
### Sub-scenario 'successful process of price-entry txn'




### Sub-scenario 'successful handling of cash payment'



## Solution 7 – The Class Diagram





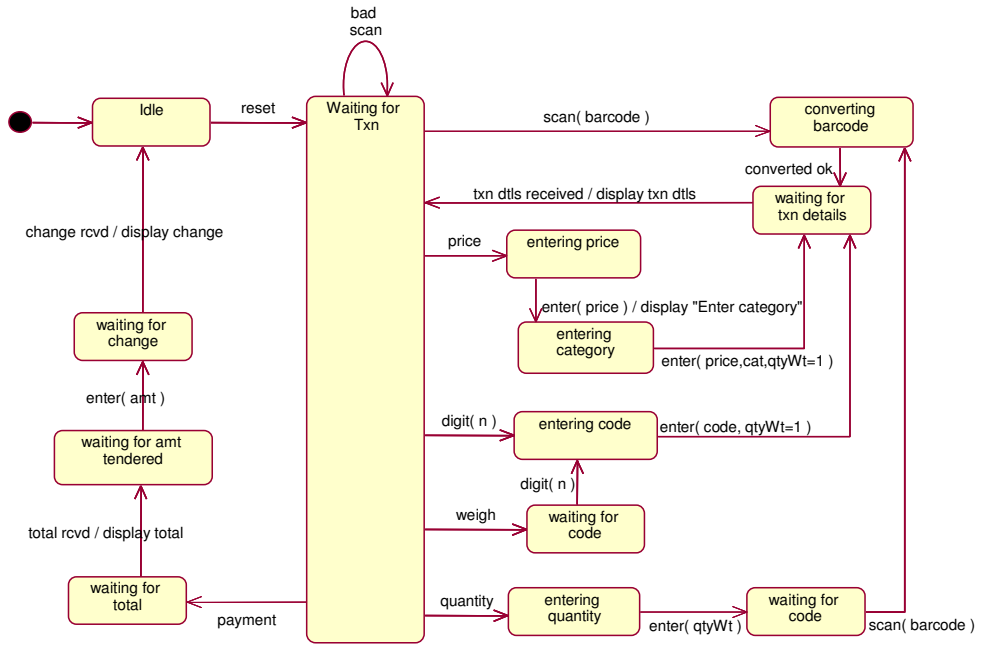


The Bar Code class plays no part in the system once it has been read and converted by the Scanner. It is shown mainly to provide a point of entry into the model for presentation purposes.

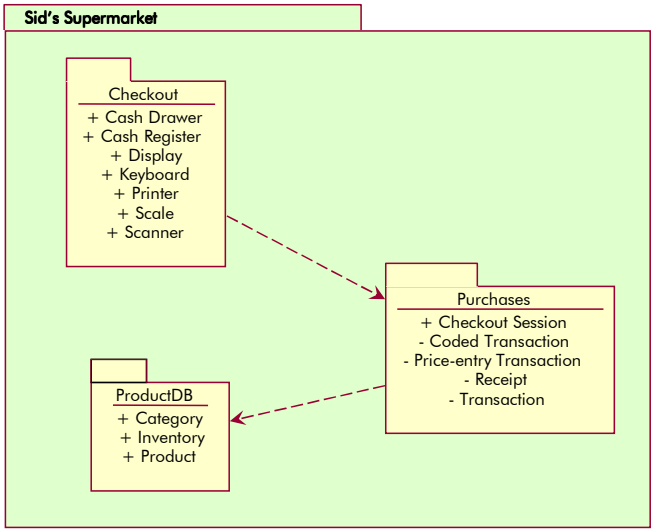
Pricing Formula is a class which has the responsibility for calculating the price of each item as it is processed, in particular those which are on special offer. The unit price is a special case of the pricing formula. Some work will be needed on this class – it is included so that it is not forgotten.

Product Tally is a class whose job is to keep track of how many items for a product code have been processed for each session. This is to allow for the correct calculation of price for multi-buy special offers when the items are interspersed with other purchases. Some detail needs to be added here, too!

## Solution 8 – The State Diagram



# Solution 9 – Packages



τecaδemy

- Notes: