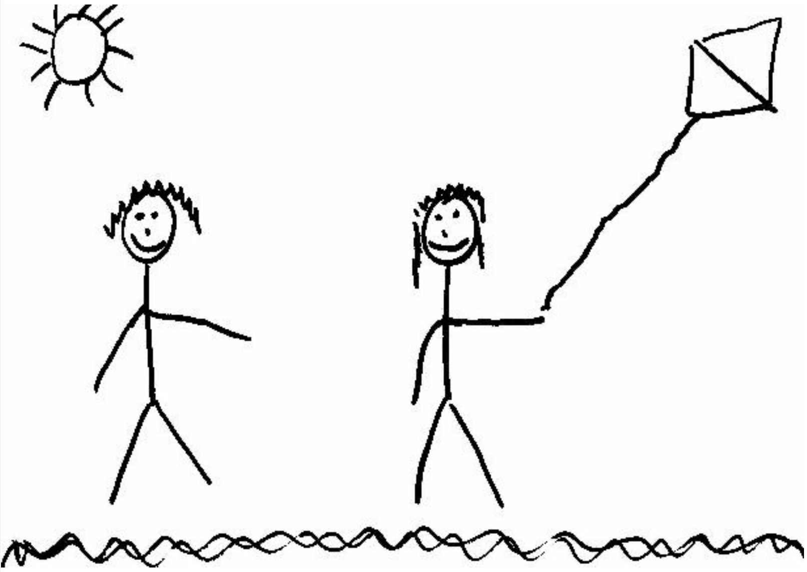
Use Case Diagrams



Use-case actors relaxing

UML

- Unified Modelling Language
- A standard graphical language to model computer applications.
- Is programming-language independent, can be used with C++, Visual Basic, Java, Python: anything.

Use Case Diagrams

- UCD
- The diagram we need to know for VCE SD.
- Is a picture of *functional requirements* in an information system.
- Lists the tasks that a system can *do*.

Use Case Diagram:

- Used during **analysis phase** of PSM
- Maps out the **functional requirements** of a system
- The emphasis is on *what* a system does rather than *how*.
- (*How* is worked out during design phase of PSM)

System?

- UCD describes **what a (software) system does** from the standpoint of an **external** observer who **uses the system** or is **affected by** the system

UCD

- Serves to **scope** the system’s high-level functionality
  - Defines what functions will or will not be performed by the system
  - Shows developers if they have missed anything important.
- Help development teams visualise the functional requirements of a system
- Used to highlight new requirements as the system is analysed and the design takes shape.
- E.g. “if we now want the system to do **X**, we’ll also need it to do A, B and C.”

UCD

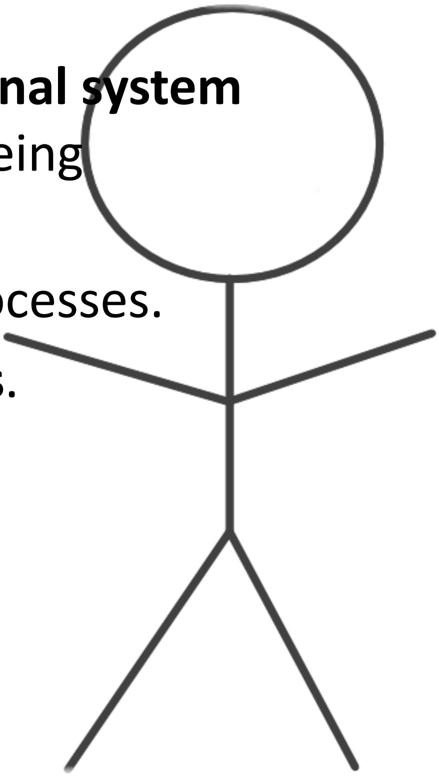
- UCD is a good tool for communicating with **clients** because UCDs are so simple and easy to read.

UCD COMPONENTS

- Actors
- Use Cases
- Associations (communications)
- <<Includes>>, <<extends>>
- System boundaries

ACTORS

- A **person**, **organisation**, or **external system** that plays a role in the system being developed.
- May initiate or participate in processes.
- Actors are drawn as **stick figures**.

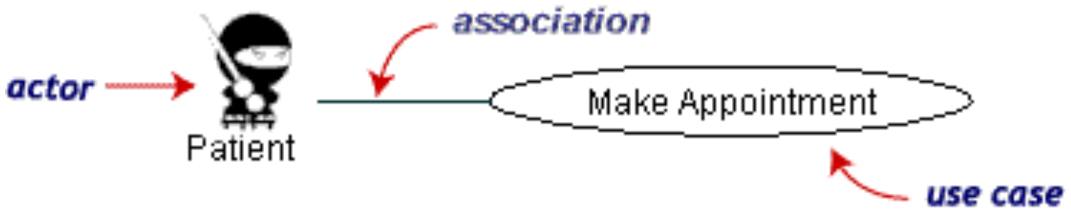


ACTORS

- Always drawn around the **edges** of the UCD
- The same actor only appears **once** in a UCD (unlike a DFD where external entities can appear more than once)
- Must be connected to at least one process
- ‘Process’ is called a **Use Case**.

Use Case

- Pronounced “yoose case”, not “yooze case”
- Drawn as an **oval** or **circle**
- Name is put in the circle
- Name must start with an **active verb** (“make”, “open” etc)



Watch out!

- In a use case’s name don't combine sentences with commas or 'and'.
- This implies that actions have been combined.
- If you see a process with an AND, snap it into two use cases.

Associations

- Sometimes called “communication”
- Shown as a line.
- Sometimes shown as **arrow** (more details later)
- Connects an actor to a use case.
- Each actor may be associated with several use cases.
- Each use case may be associated with several actors.

Arrowed associations

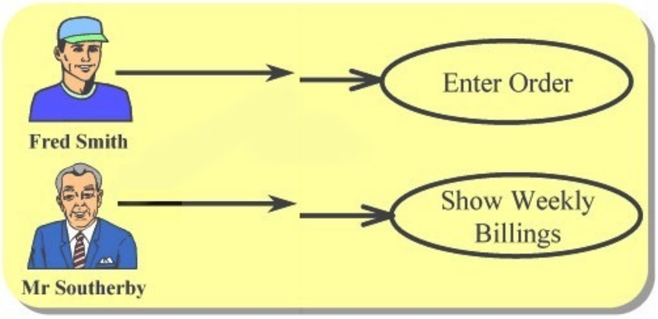
Important thing to make clear now

- Arrows do **not** indicate data flow!
- They are **not** Data Flow Diagrams (DFD)
- Assume communication is always **two way** between actor and use case.
- Do not need separate arrows for each datum (as you do in a DFD)

Arrows in UCD

- An arrow just indicates which actor **initiated** the use case (the **primary actor**) and which actor was affected by it starting (the **passive actor**)
- With no arrow, it's unclear which actor **started** a use case...

Actors are **roles**, not individuals or job titles



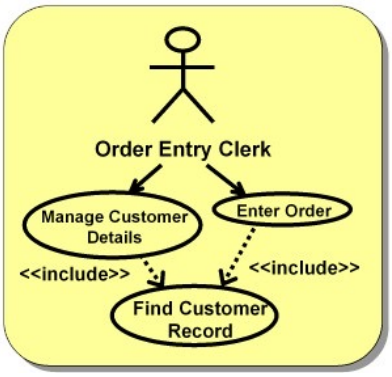
- **Mr Southerby** as *CEO* needs access to management information.
- **Fred Smith** as *Order Entry Clerk* needs to enter orders.

<<INCLUDE>>

- Like a subprogram
- Denotes the inclusion of behaviour described by **another use case**.
- Saves repetition and wasted space.
- Shown as a dotted line with "<<include>>" label

<<INCLUDE>>

- Both the *Manage Customer Details* and *Enter Order* use cases include the *Find Customer Record* use case.
- They both have the procedures from *Find Customer Record* in common.



The <<include>> arrow

- Arrow points **from** the base use case **to** the included use case.
- Think of “this base use case > INCLUDES > this use case”

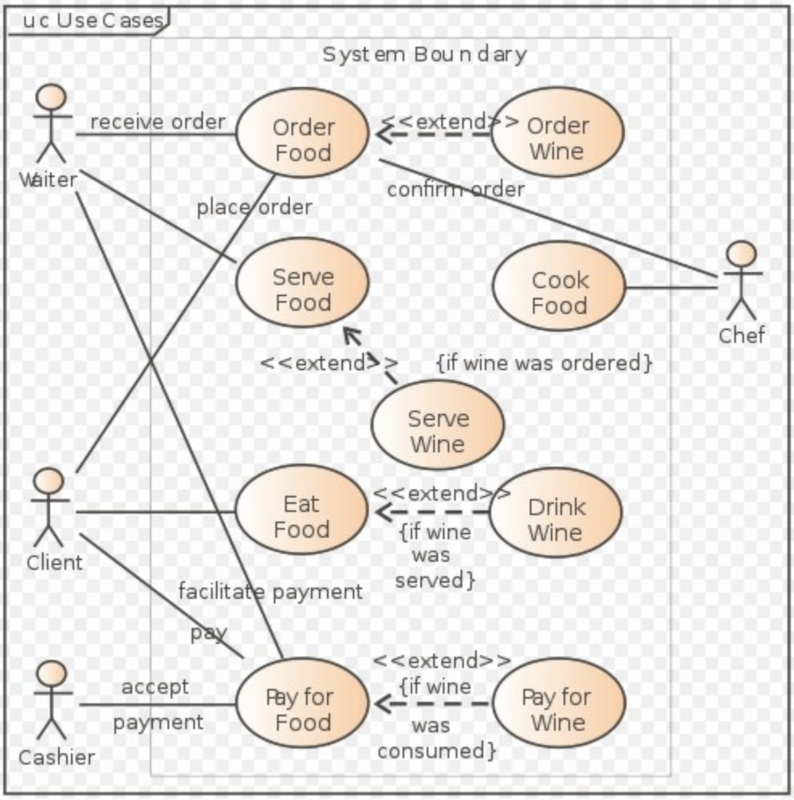
<<extend>>

• Like a **conditional** <<include>>.
• Only takes place under certain conditions.
• An <<extend>> is a use case that **extends** another use case (the ‘base’ use case) when circumstances require it.
•When the extending use case’s activities are finished, the base use case’s activities resume from the point they branched off into the <<extend>> use case.

<<extend>>

- The extending use case is really an alternate course of the base use case
- It can be used to describe different actions to be taken dependent on current condition.
- In other words, it can indicate conditional branches in activities.
- Shown as a dotted line with “<<extend>>” label

<<extend>>

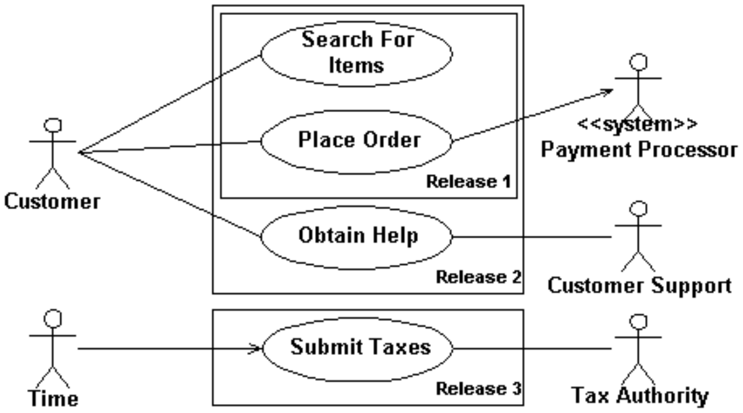


Note how some activities only take place if wine has been ordered.

Note: Solid lines or arrows between use cases are **not allowed**. The only way to link use cases is with *includes* or *extends*.

System Boundaries

- You can draw a rectangle – the system boundary - around use cases to indicate the scope of the system. They are usually not needed, but use them anyway.
- Useful to mark off functional components that will be rolled out in different releases of a system...



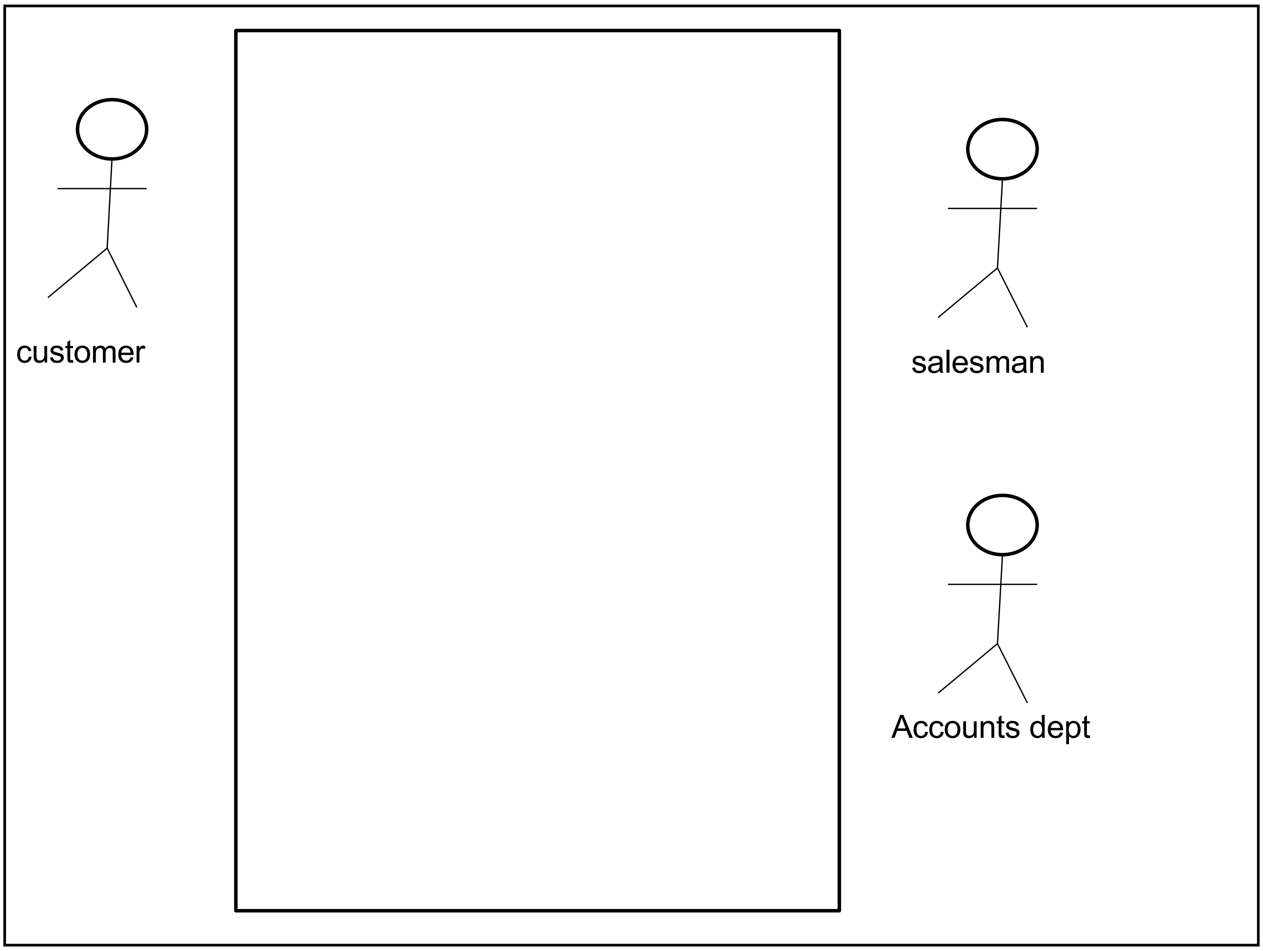
Practice

Create a UCD to describe the following requirements of a new system.

- Customers can contact salesmen to make a purchase.
- Salesmen enter the sale into the sales database.
- Customers can track their sale's progress via the database.
- The accounts department sends an invoice to the customer.

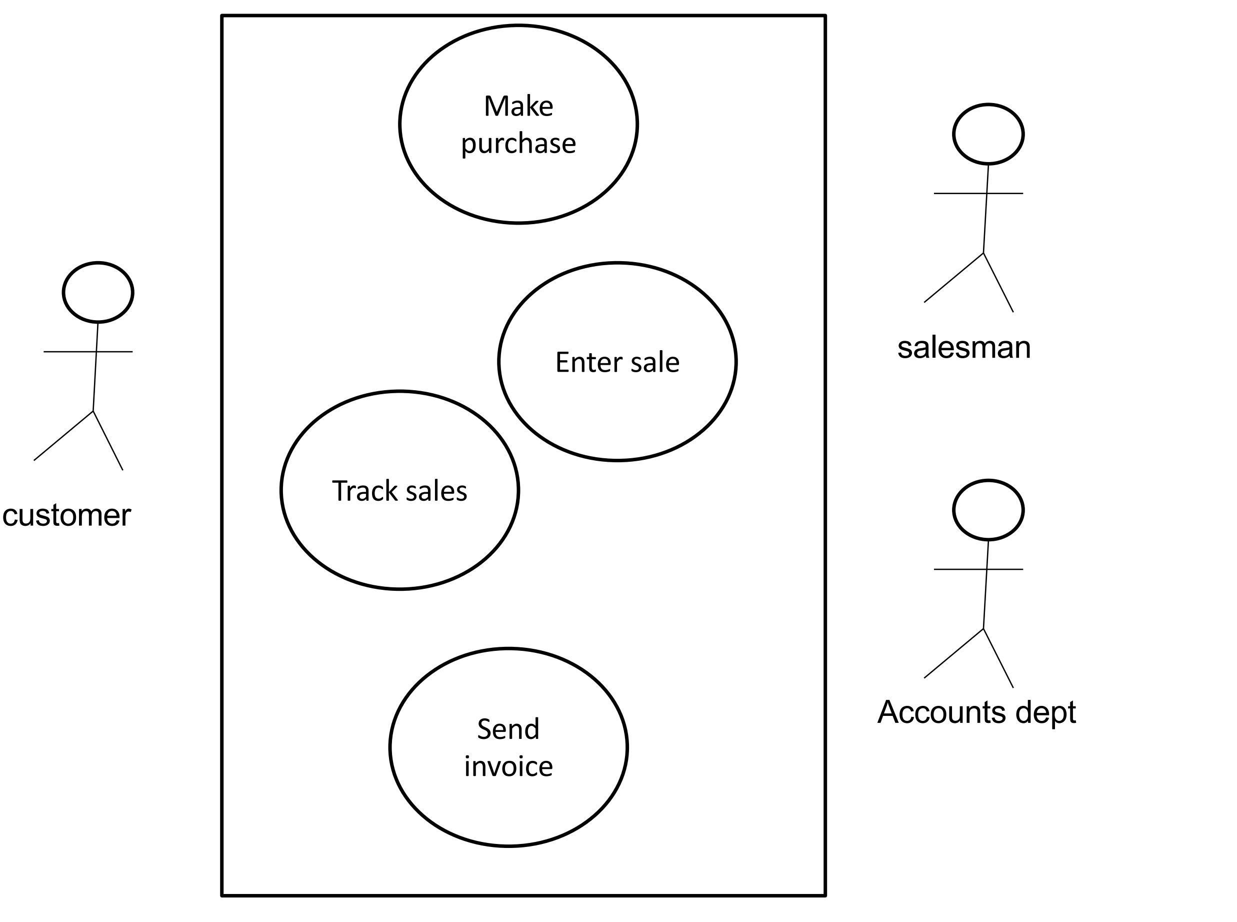
1. Identify the actors

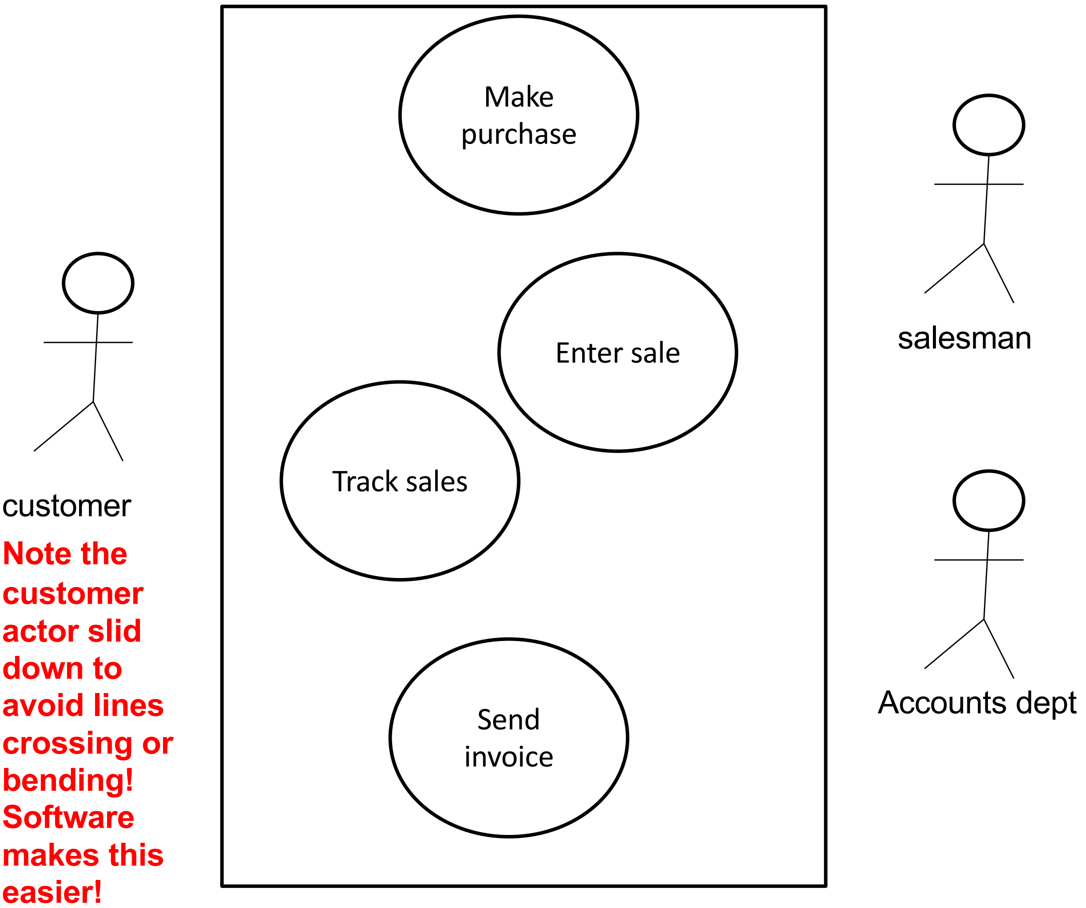
- **Customers** can contact **salesmen** to make a purchase.
- Salesmen enter the sale into the sales database.
- Customers can track their sale's progress via the database.
- The **accounts department** sends an invoice to the customer.
- All of these participants interact with the system from the outside.
- No, the **database** is **not** an actor because it is **part of the system.** It is **not shown** in the UCD (but it *would appear as a data store in a DFD!)*



2. Identify the use cases

- **1.** Customers can contact salesmen to **make a purchase.**
- 2. Salesmen **enter the sale** into the sales database.
- 3. Customers can **track their sale's** progress via the database.
- 4. The accounts department **sends an invoice** to the customer.

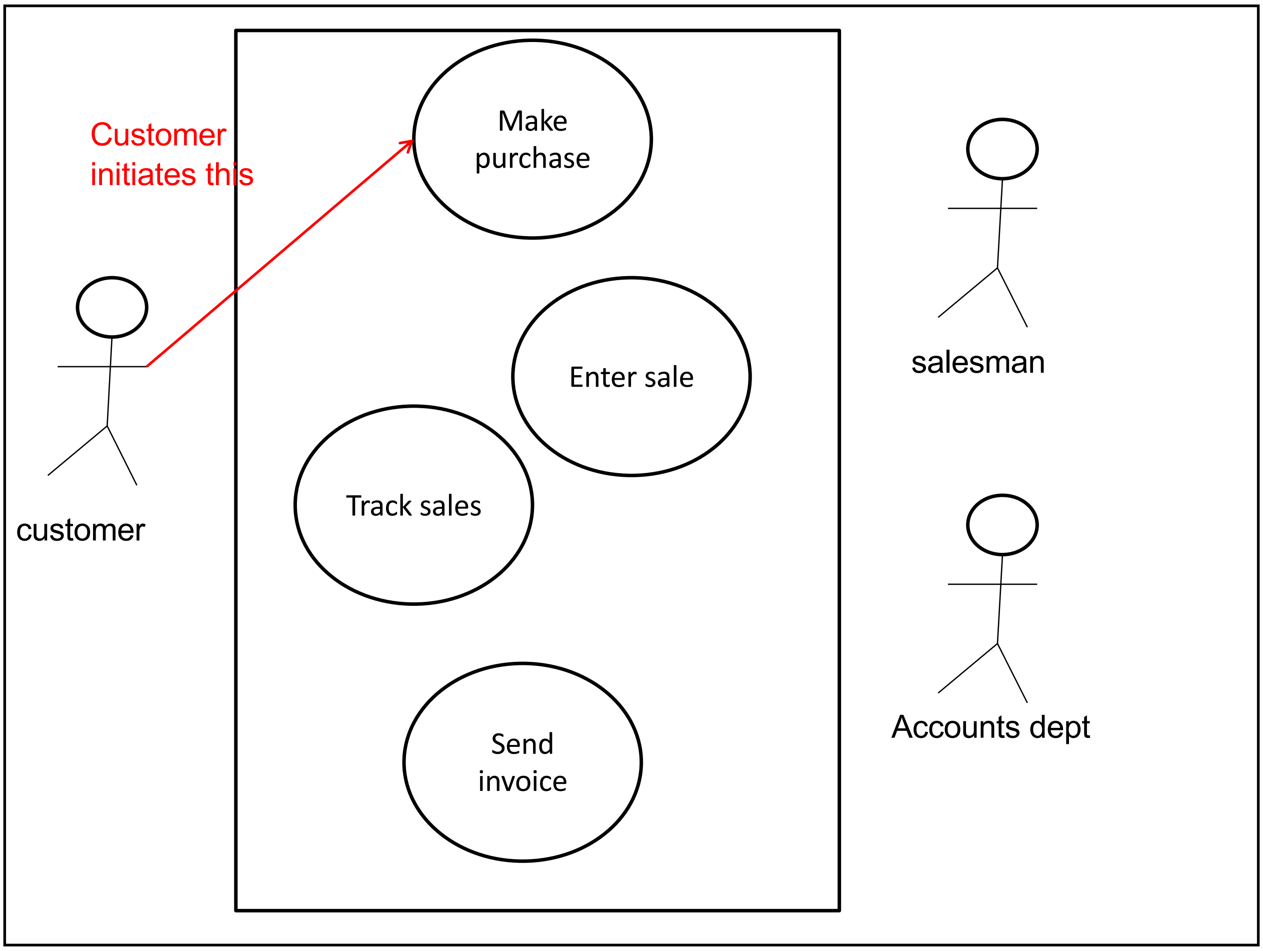




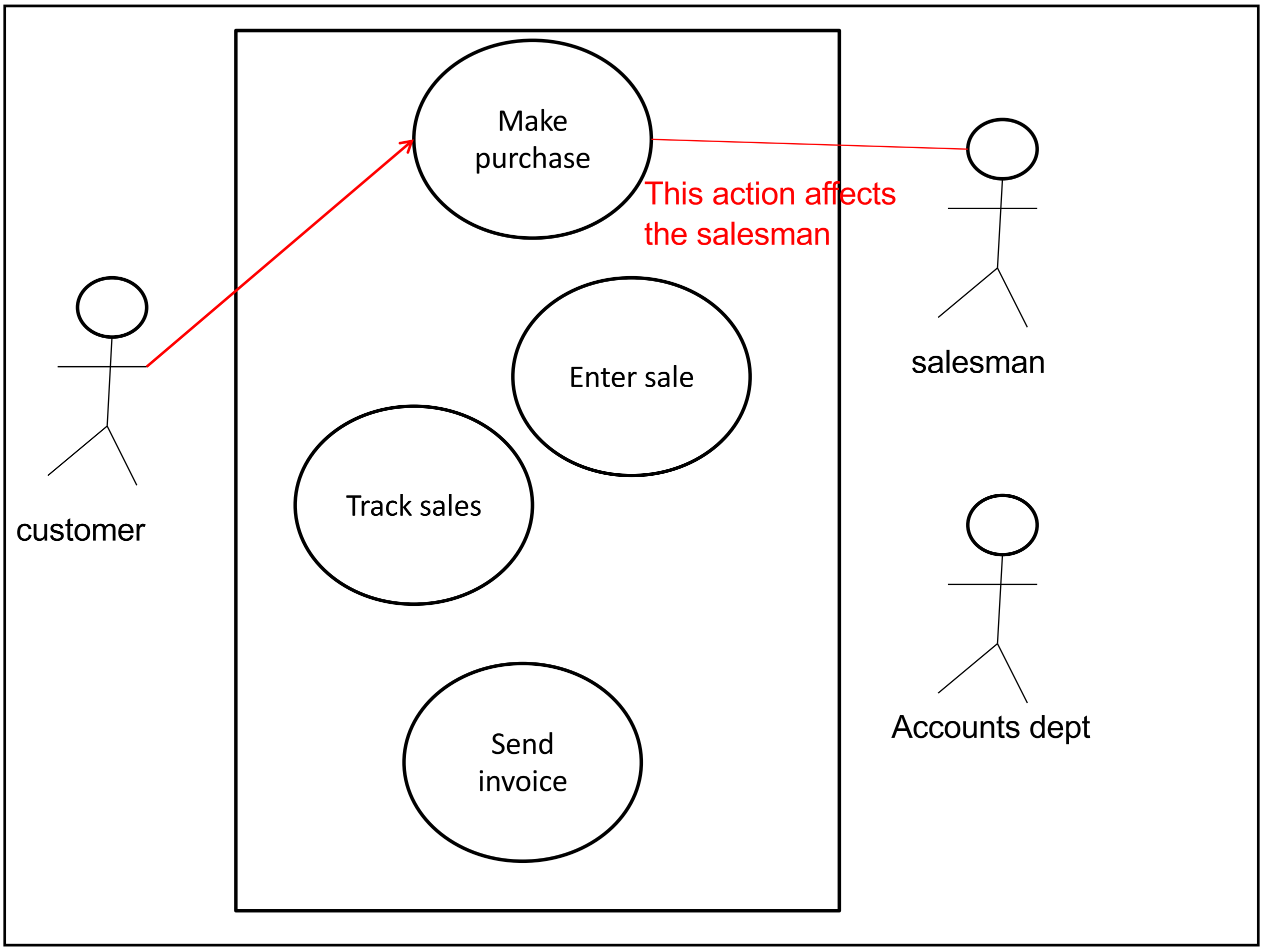
Note the customer actor slid down to avoid lines crossing or bending! Software makes this easier!

Add the association lines

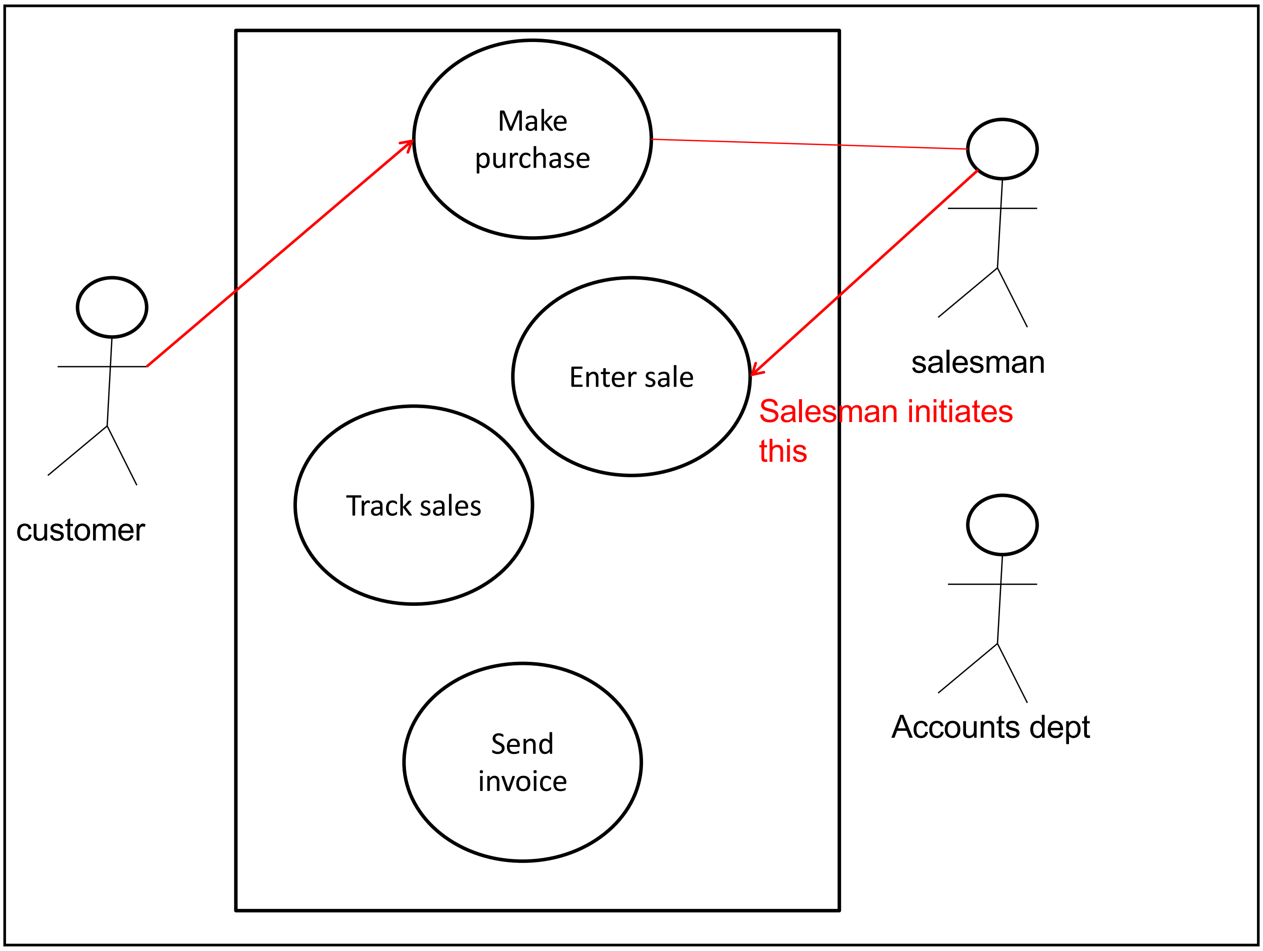
- Add arrows if an actor initiates the use case

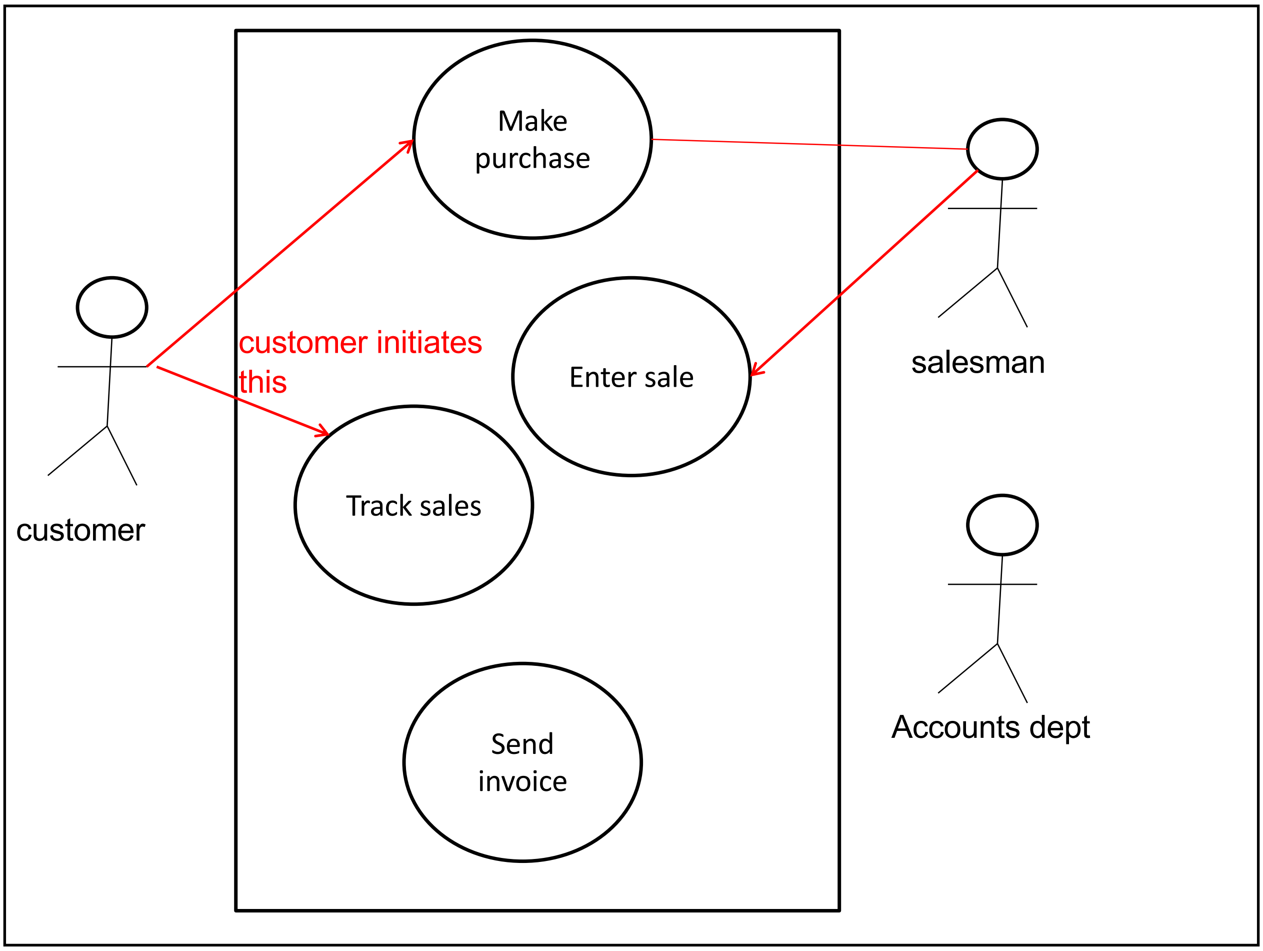


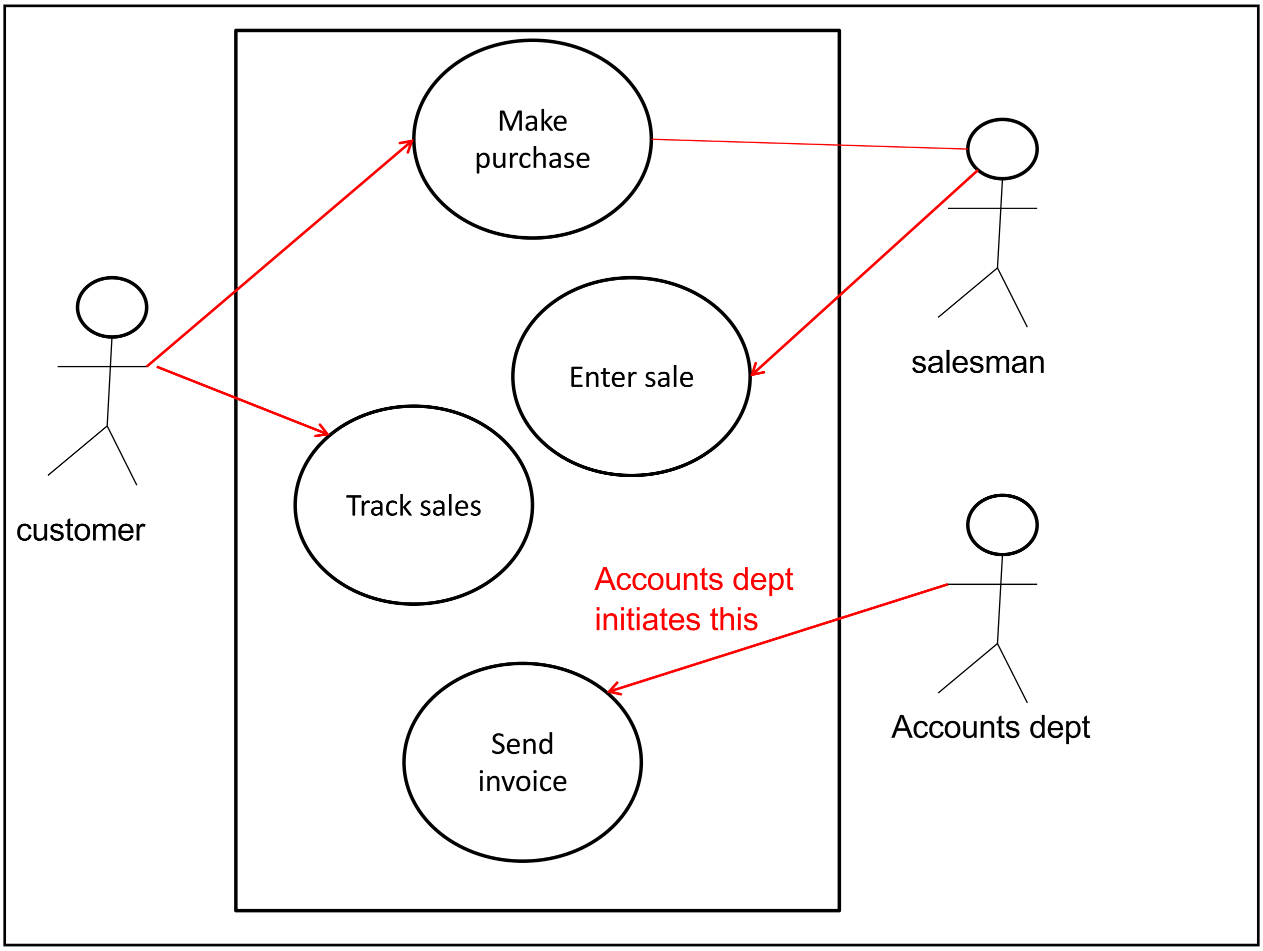
Customer initiates this



This action affects the salesman

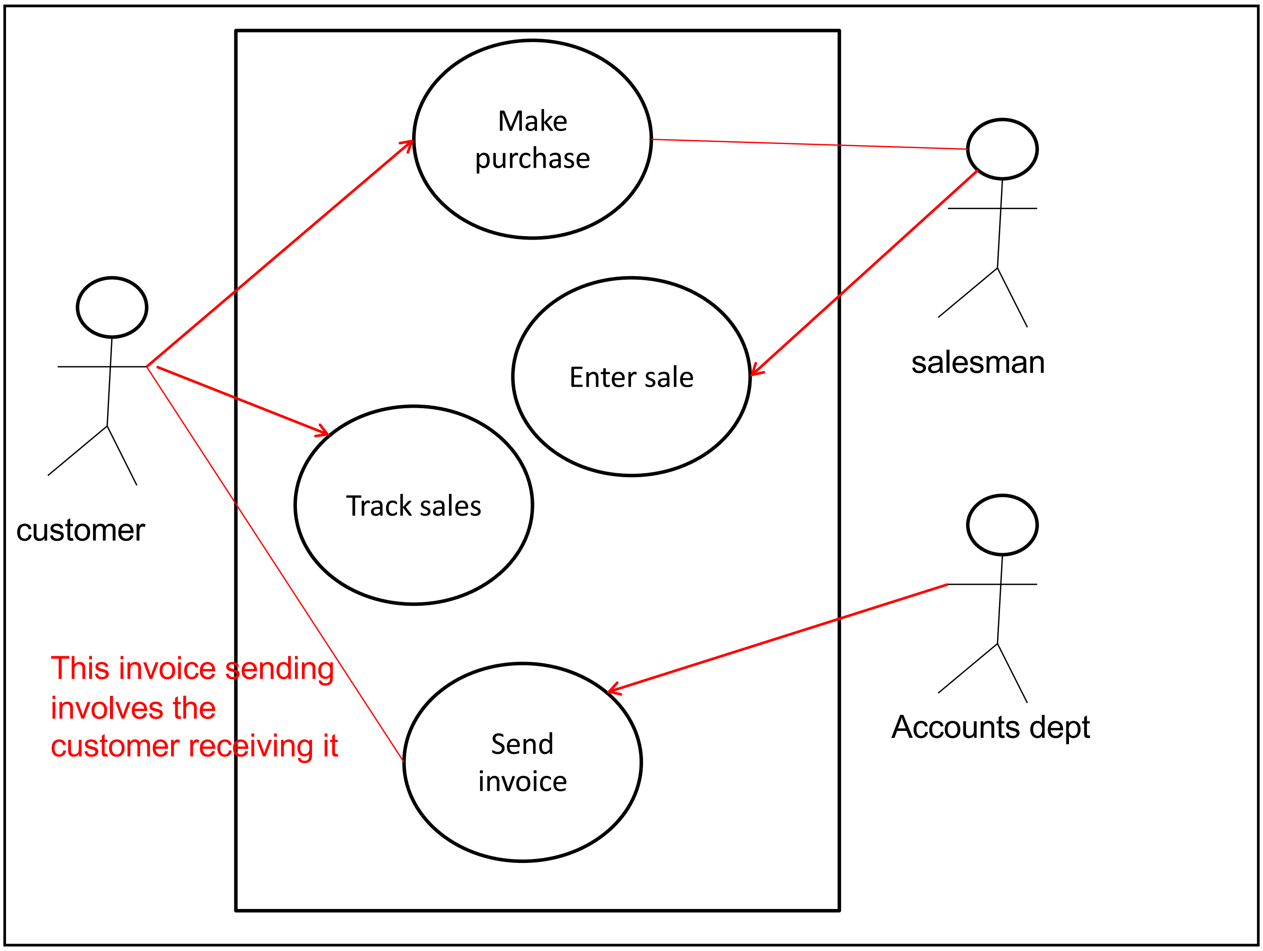

Make purchase
Enter sale
Track sales
Send invoice
customer
salesman
Accounts dept
Salesman initiates this


Make purchase
Enter sale
Track sales
Send invoice
customer
salesman
Accounts dept
customer initiates this



Accounts dept initiates this

35



This invoice sending involves the customer receiving it

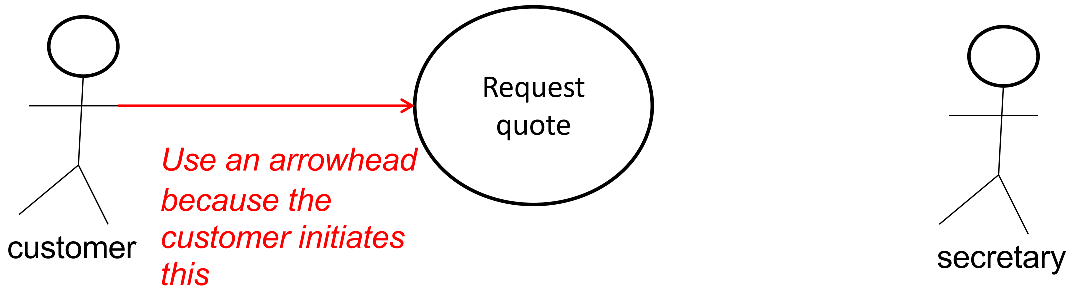
Practice

Create a UCD to describe the following requirements of a new system in Windoze P/L.

- Customers contact the company secretary to request a quotation
- The secretary books the quotation **and** advises the customer of the date and time
- The window installer uses the system to calculate the quotation at the customer's house.
- The bursar processes the customer's payment. If the customer paid by credit card, the card needs to be processed first.

Solution

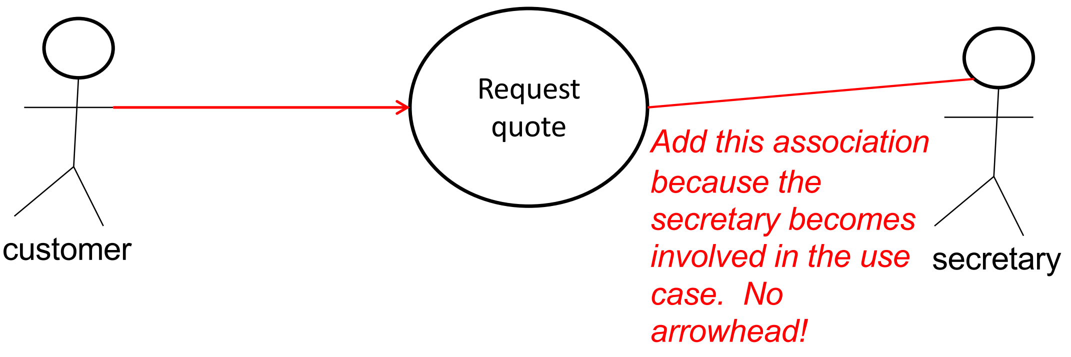
- Customers contact the company secretary to request a quotation



Use an arrowhead because the customer initiates this

Solution

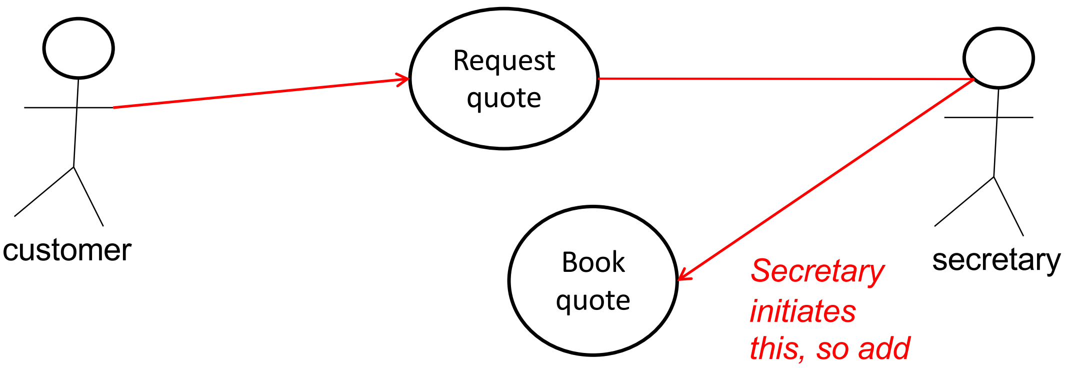
- Customers contact the company secretary to request a quotation



Add this association because the secretary becomes involved in the use case. No arrowhead!

Solution

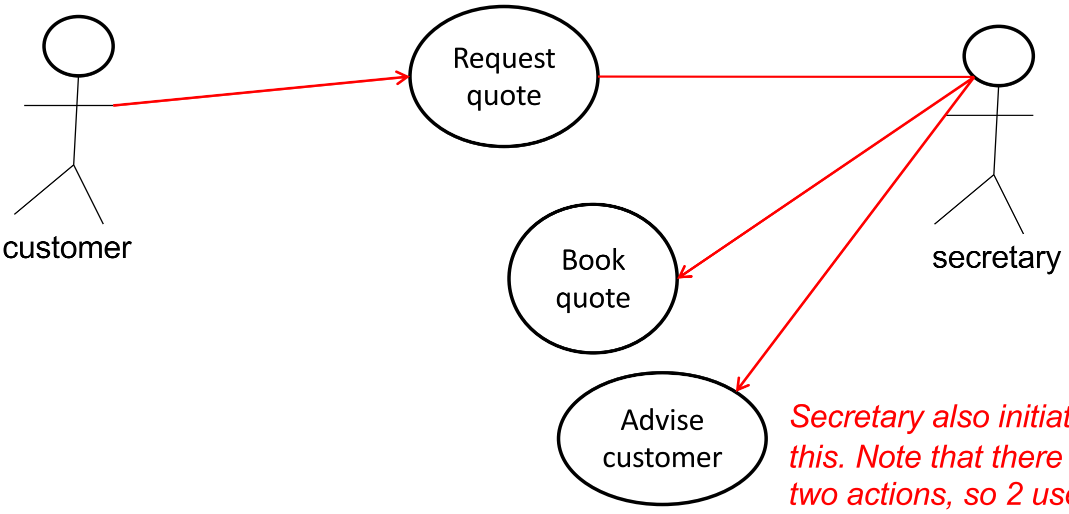
- The secretary books the quotation **and** advises the customer of the date and time



Secretary initiates this, so add arrowhead

Solution

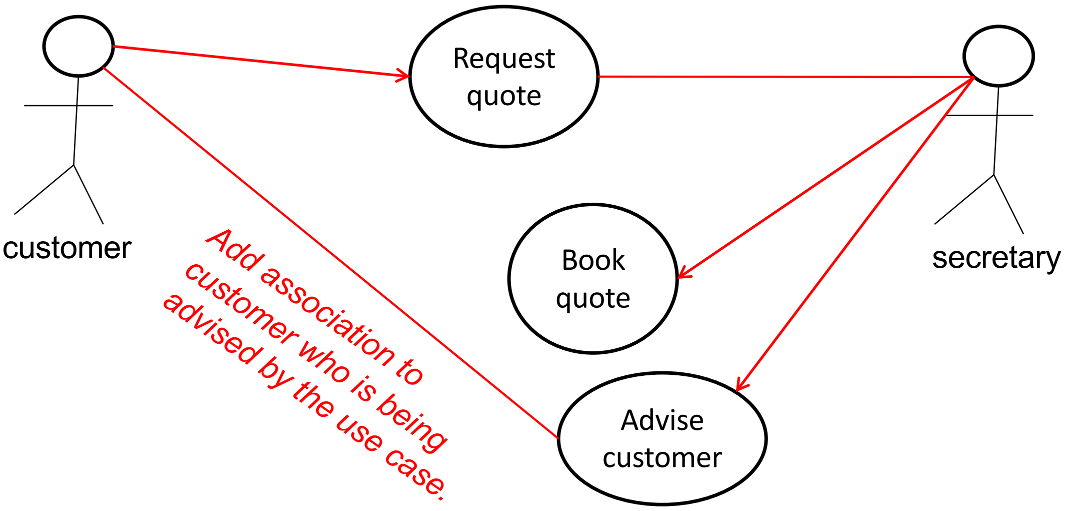
- The secretary books the quotation **and** advises the customer of the date and time



Secretary also initiates this. Note that there are two actions, so 2 use cases.

Solution

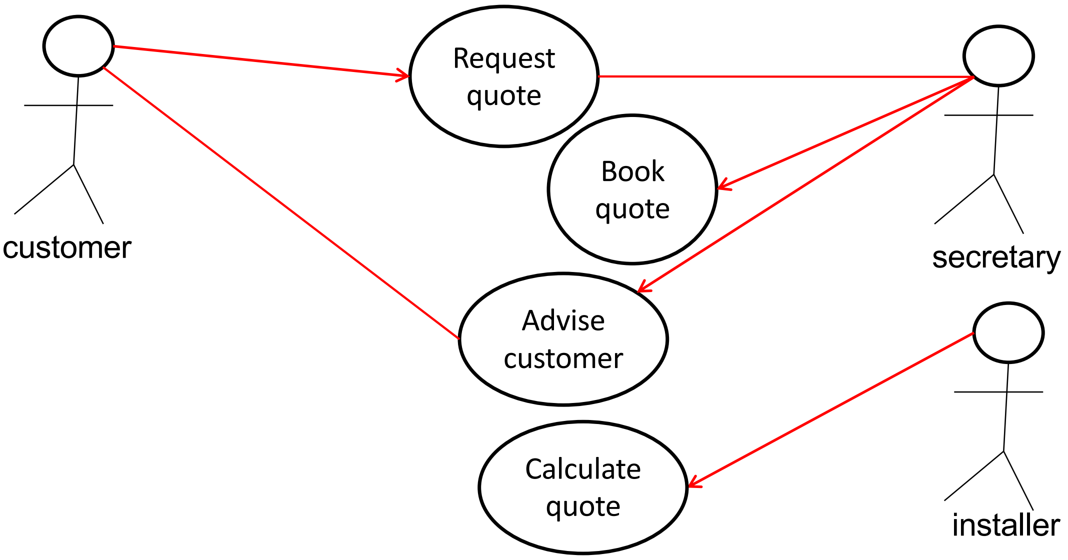
- The secretary books the quotation **and** advises the customer of the date and time



Add association to customer who is being advised by the use case.

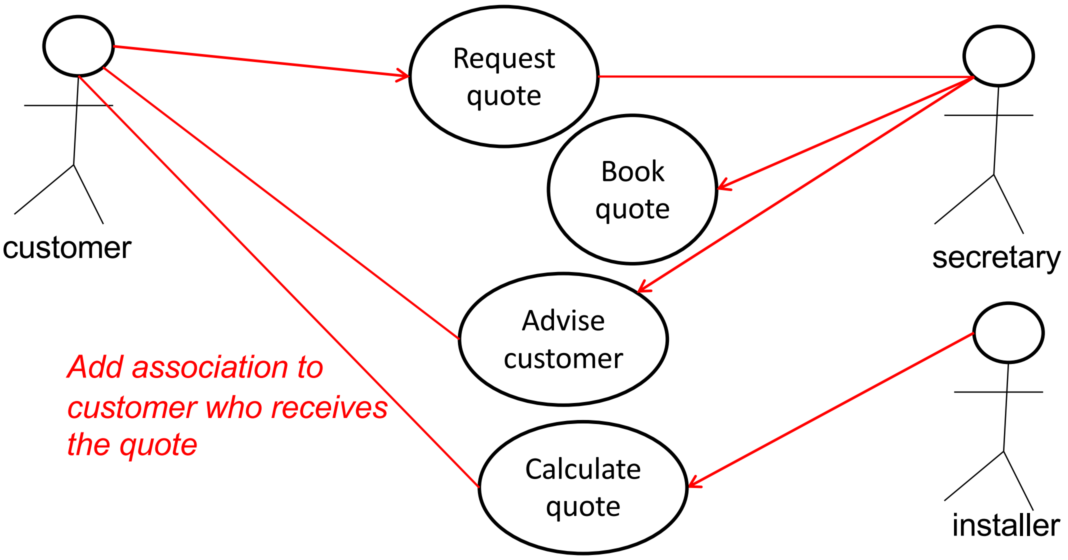
Solution

- The window installer uses the system to calculate the quotation at the customer’s house.



Solution

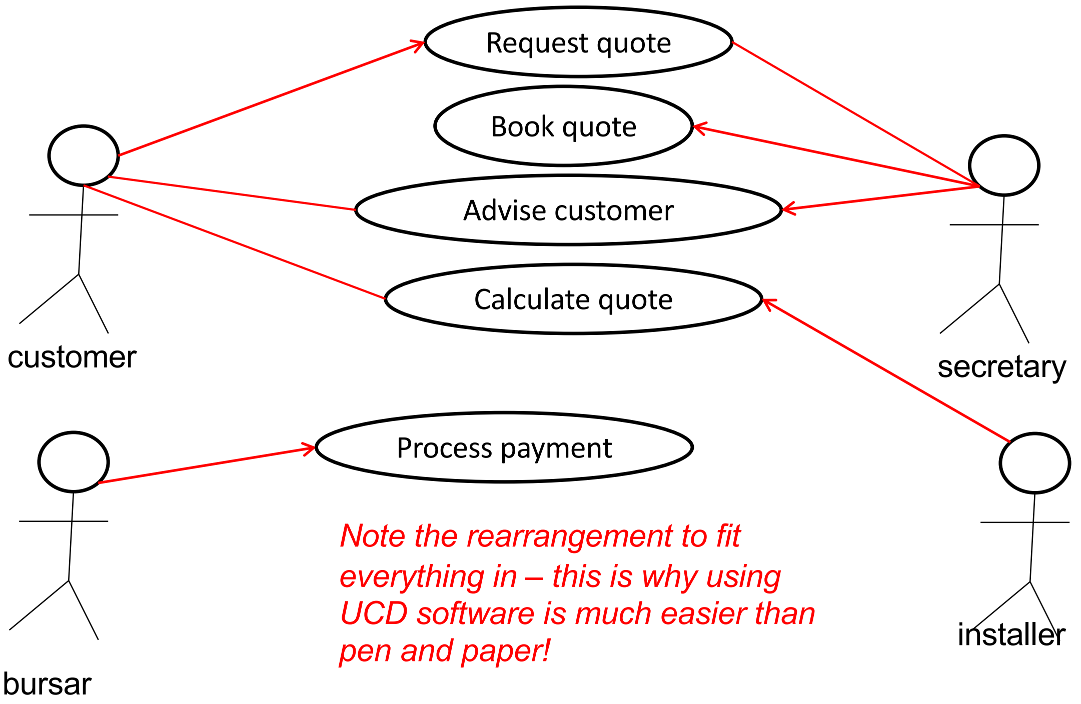
- The window installer uses the system to calculate the quotation at the customer’s house.



Add association to customer who receives the quote

Solution

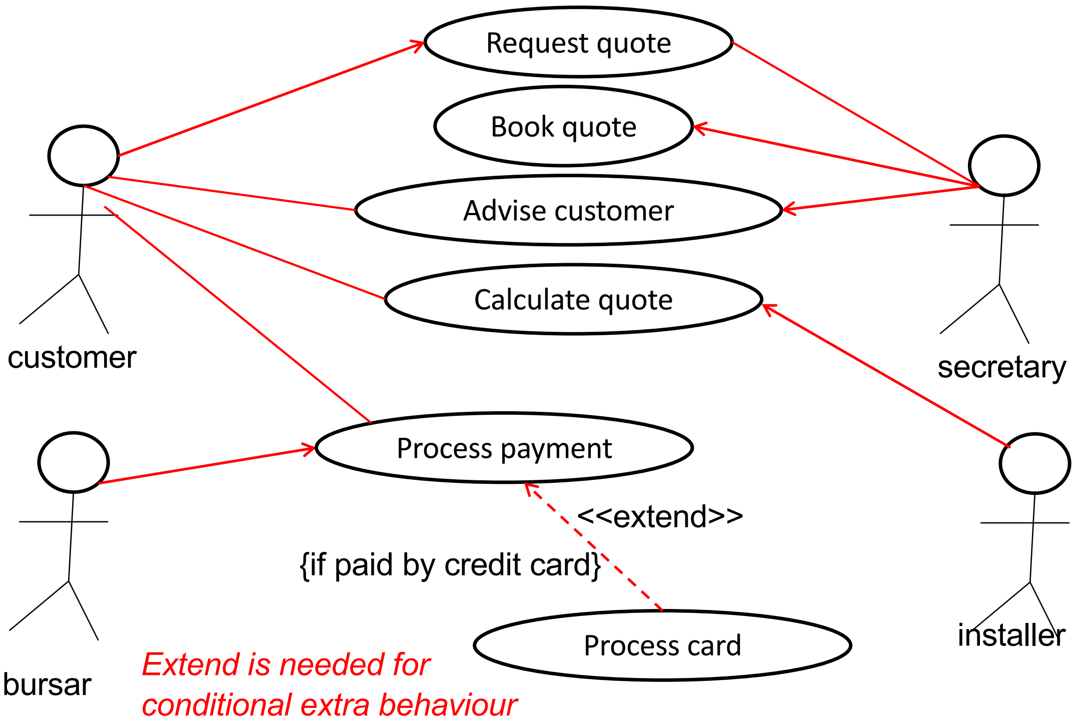
- The bursar processes the customer's payment



Note the rearrangement to fit everything in – this is why using UCD software is much easier than pen and paper!

Solution

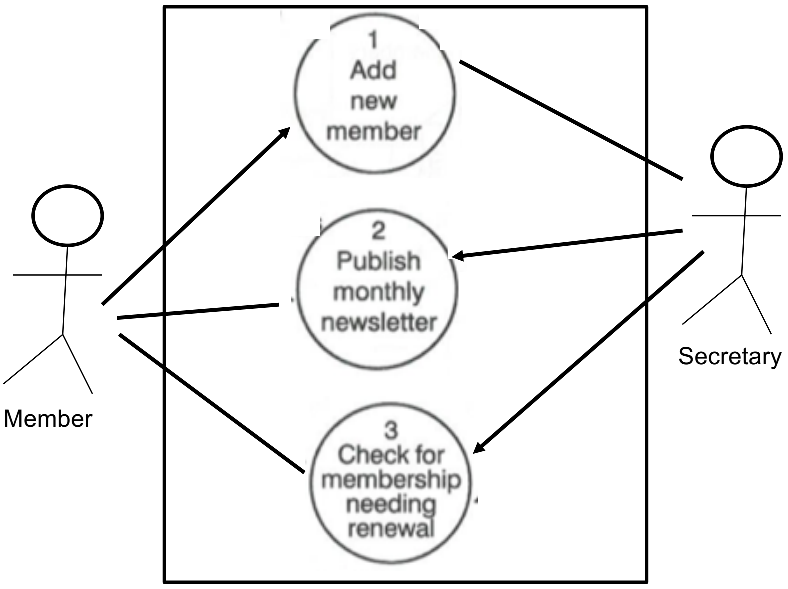
- If the customer paid by credit card, the card needs to be processed first.



Extend is needed for conditional extra behaviour

Case Study: Brindabella Bird Watching Club

Brindabella Bird Watching Club has hundreds of enthusiastic members. People can join the club by sending in a membership form along with $10. Members receive newsletters outlining the club's activities for the coming month. Members also receive reminder letters when their membership is due for renewal.

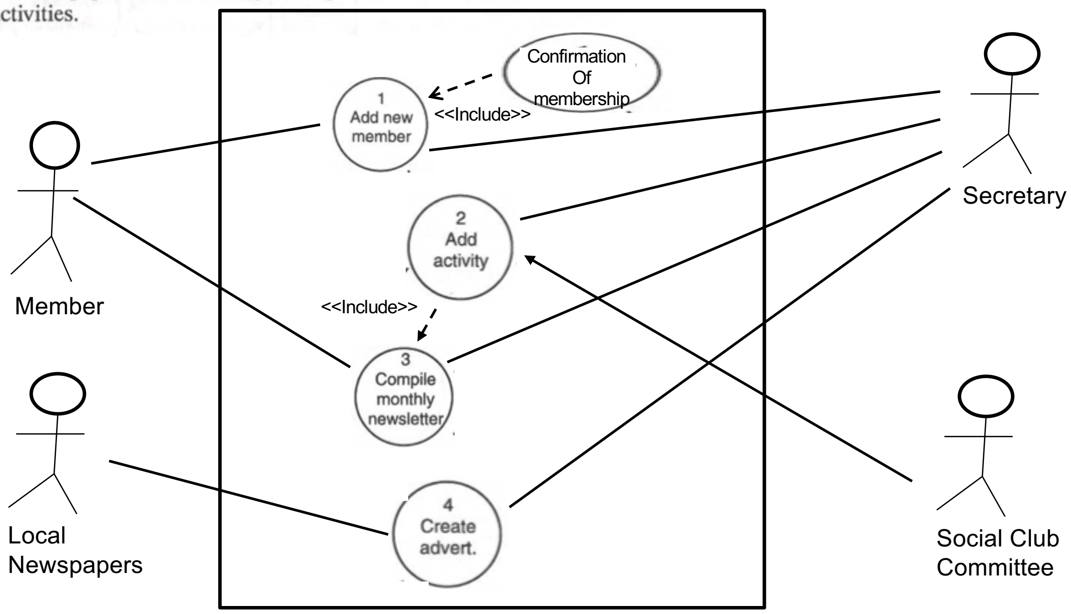

1
Add new member
2
Publish monthly newsletter
3
Check for membership needing renewal
Member
Secretary

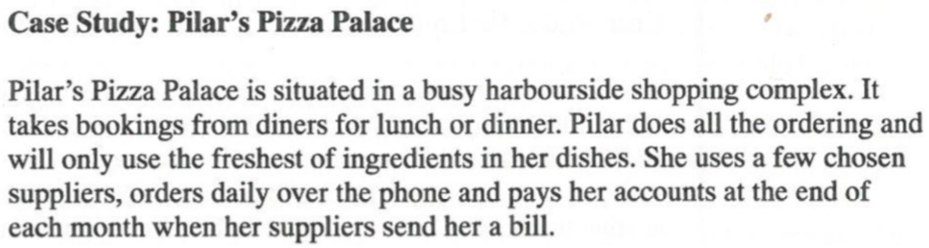
47

Case study: Black Stump Social Club

The Black Stump Social Club has an enthusiastic club secretary who has set up a system for keeping track of all the club's members as well as all the club's activities. New members' details are added to the members file and confirmation of membership letters are sent out.

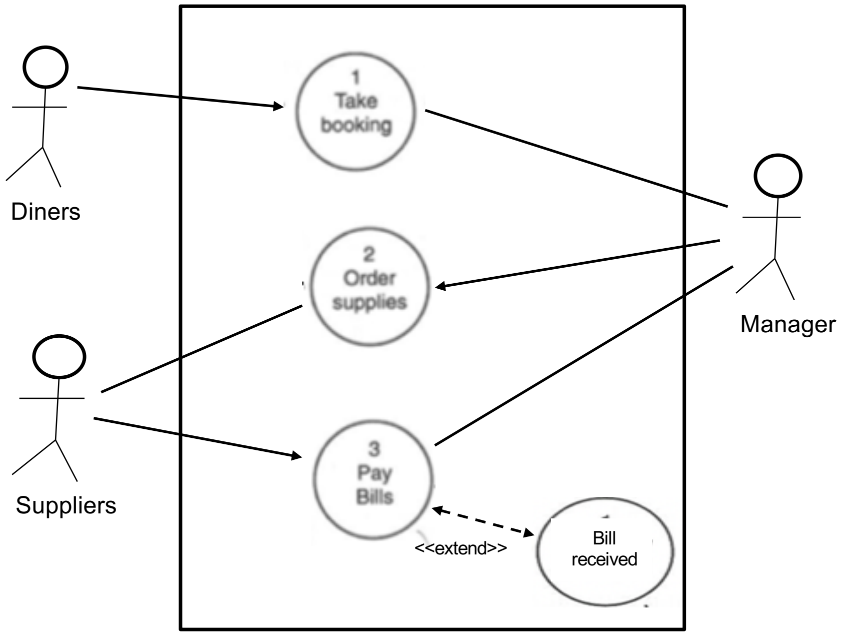
The Social Club Committee meets to plan activities and these are added by the secretary to the activities file. A monthly newsletter listing all coming activities is compiled and sent to each member. Advertisements are also sent to the local newspaper to inform the general public of the social club's coming activities.

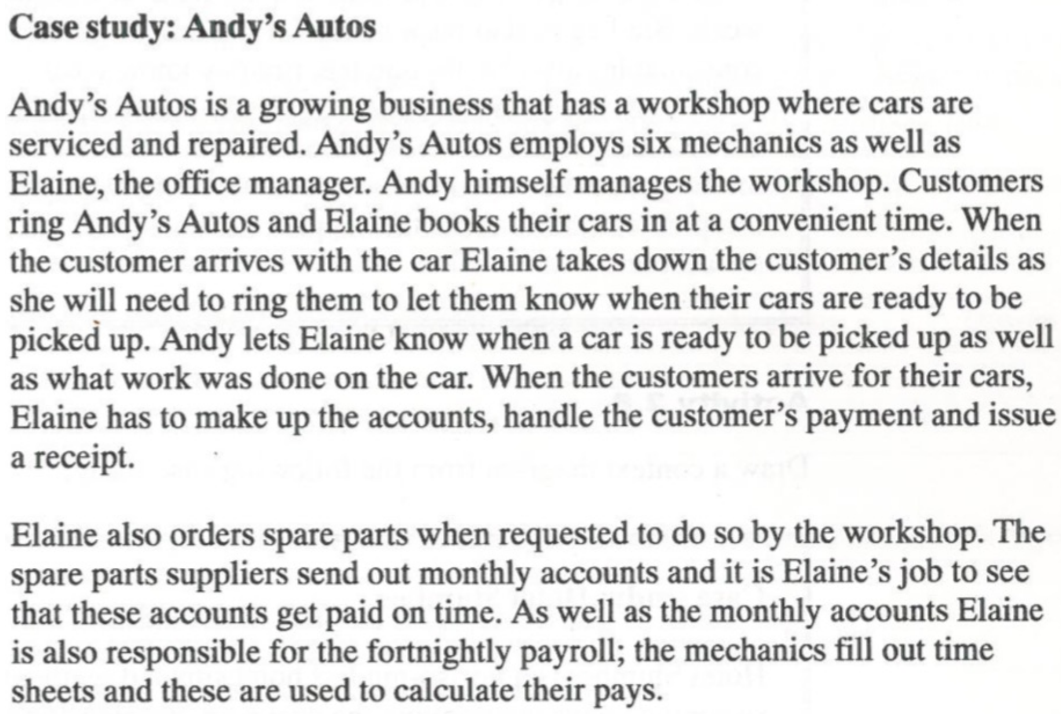




Case Study: Pilar's Pizza Palace

Pilar's Pizza Palace is situated in a busy harbourside shopping complex. It takes bookings from diners for lunch or dinner. Pilar does all the ordering and will only use the freshest of ingredients in her dishes. She uses a few chosen suppliers, orders daily over the phone and pays her accounts at the end of each month when her suppliers send her a bill.

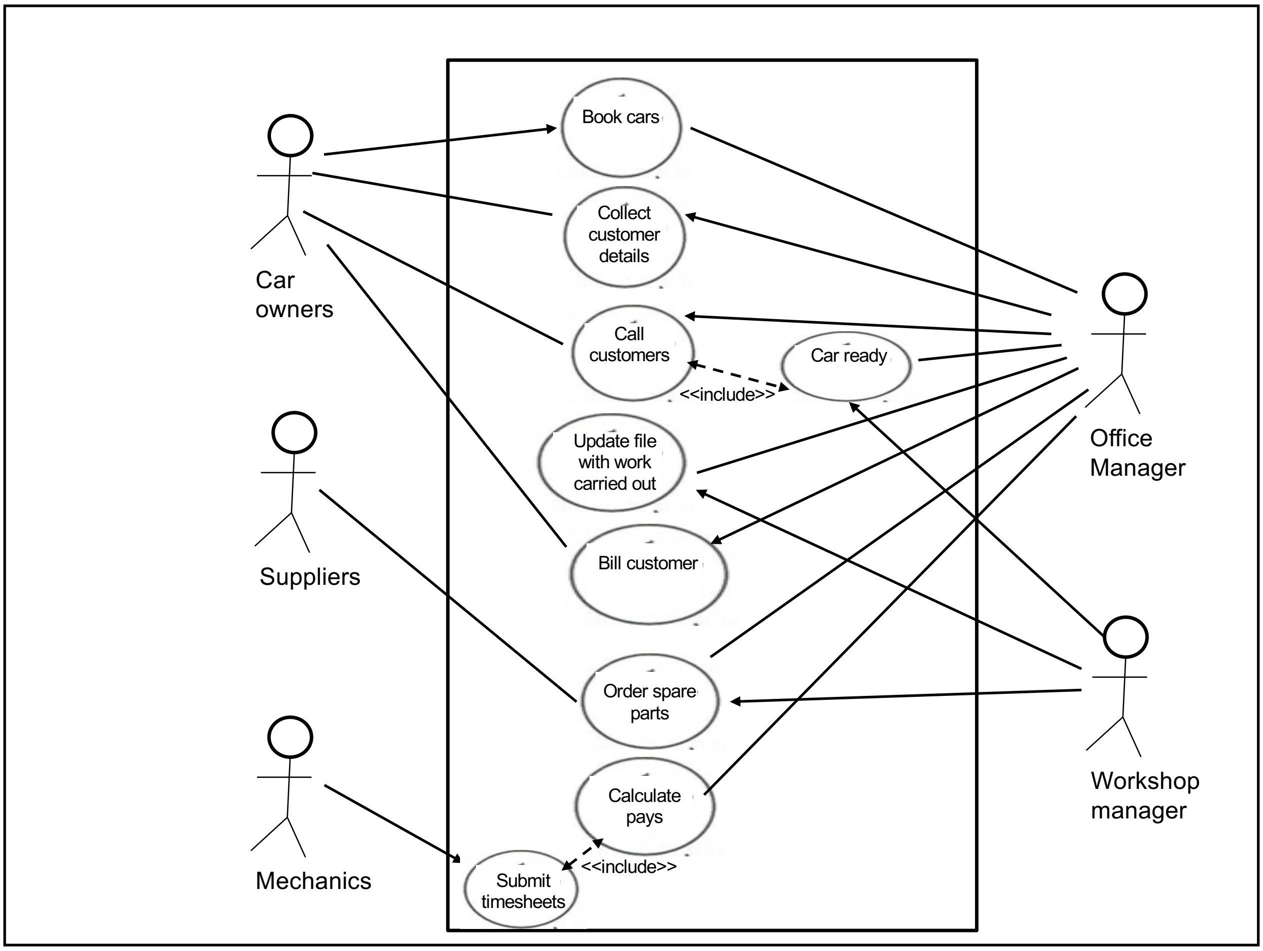




Case study: Andy's Autos

Andy's Autos is a growing business that has a workshop where cars are serviced and repaired. Andy's Autos employs six mechanics as well as Elaine, the office manager. Andy himself manages the workshop. Customers ring Andy's Autos and Elaine books their cars in at a convenient time. When the customer arrives with the car Elaine takes down the customer's details as she will need to ring them to let them know when their cars are ready to be picked up. Andy lets Elaine know when a car is ready to be picked up as well as what work was done on the car. When the customers arrive for their cars, Elaine has to make up the accounts, handle the customer's payment and issue a receipt.

Elaine also orders spare parts when requested to do so by the workshop. The spare parts suppliers send out monthly accounts and it is Elaine's job to see that these accounts get paid on time. As well as the monthly accounts Elaine is also responsible for the fortnightly payroll; the mechanics fill out time sheets and these are used to calculate their pays.


Car owners
Suppliers
Mechanics
Office Manager
Workshop manager
Book cars
Collect customer details
Call customers
Car ready
<<include>>
Update file with work carried out
Bill customer
Order spare parts
Calculate pays
<<include>>
Submit timesheets