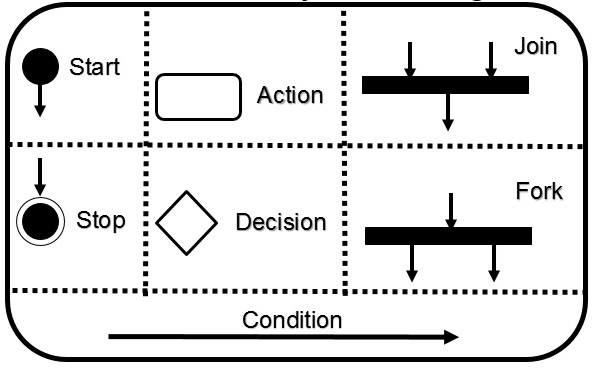


**Data Flow Diagram**

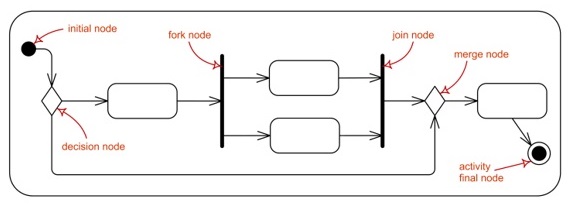
|  |
| --- |
| To order food: |
| A CUSTOMER(EXTERNAL ENTITY) will ORDER FOOD(PROCESS);  The ORDER FOOD process will send an ORDER to the KITCHEN(EXTERNAL ENTITY);  The ORDER FOOD process will send an ORDER to the ORDER(DATA STORE);  The ORDER FOOD process will send an INVENTORY DETAILS to the INVENTORY(DATA STORE);  The INVENTORY data store sends the INVENTORY DETAILS to the GENERATE REPORTS(PROCESS);  The ORDER data store sends the ORDER to the GENERATE REPORTS(PROCESS);  The GENERATE REPORTS process sends REPORTS to the MANAGER(EXTERNAL ENTITY);  The MANAGER external entity sends the INVENTORY ORDER to the ORDER INVENTORY(PROCESS);  The ORDER INVENTORY process send INVENTORY DETAIL back to the INVENTORY(DATA STORE);  The ORDER INVENTORY process send INVENTORY ORDER to the SUPPLIER(EXTERNAL ENTITY); |

|  |
| --- |
| To rent videos: |
| A CUSTOMER(EXTERNAL ENTITY) send a VIDEO REQUEST to RENT VIDEO(PROCESS);  The RENT VIDEO process will send a BILL to the CUSTOMER(EXTERNAL ENTITY);  The RENT VIDEO process will send RENTAL INFO to the RENTAL(DATA STORE);  A CUSTOMER(EXTERNAL ENTITY) send VIDEO RENTAL INFO to RETURN VIDEO(PROCESS);  The RETURN VIDEO process send a RETURN RECEIPT to the CUSTOMER(EXTERNAL ENTITY);  The RETURN VIDEO process send VIDEO INFO to the VIDEO LIBRARY(DATA STORE);  The RETURN VIDEO process send RENTAL INFO to the RENTAL(DATA STORE);  The VIDEO LIBRARY data store send VIDEO INFO to RENT VIDEO(PROCESS);  The RENTAL data store send RENTAL INFO to GENERATE RENTAL REPORT(PROCESS);  The GENERATE RENTAL REPORT process sends a RENTAL REPORT to the MANAGER(EXTERNAL ENTITY) ; |

Activity Diagram



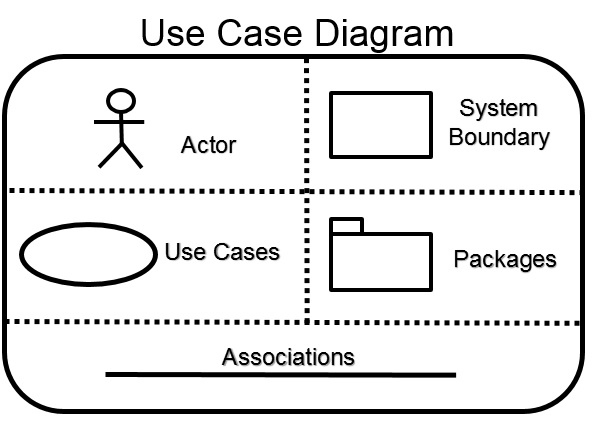
Activity Diagram

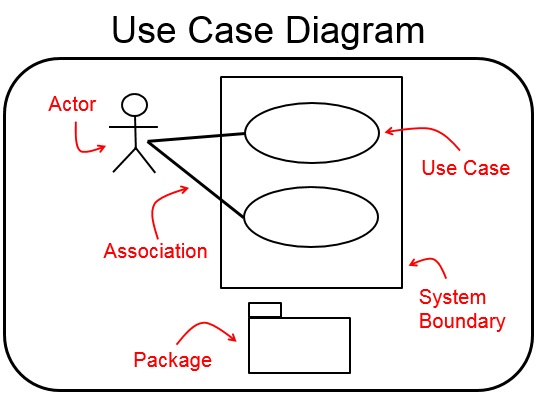


**Activity Diagram**

|  |
| --- |
| To enroll in college: |
| Fill Out Enrollment Forms;  IF the forms are filled out incorrectly  THEN IF it’s a big problem  THEN IF there is help available  THEN “Obtain Help”;  ELSE Return to “Fill Out Enrollment Forms”;  ENDIF;  ELSE it’s a trivial problem so “Enrol in University”  END IF;  ELSE “Enrol in University”  END IF;  And happening in parallel(Concurrently):   * Attend university * Enrol in Seminar(s) 🡪 Make Initial Tuition Payment |

|  |
| --- |
| To check-in at the airport: |
| Show Your Ticket at the Counter;  Verify the Ticket;  IF the Ticket is NOT OK  THEN Refer to CUSTOMER SERVICES;  EXIT;  END IF;  Check in Luggage;  Accept Luggage;  IF Luggage is Subject to Fee  THEN Pay Fee;  END IF;  Issue Boarding Pass; |

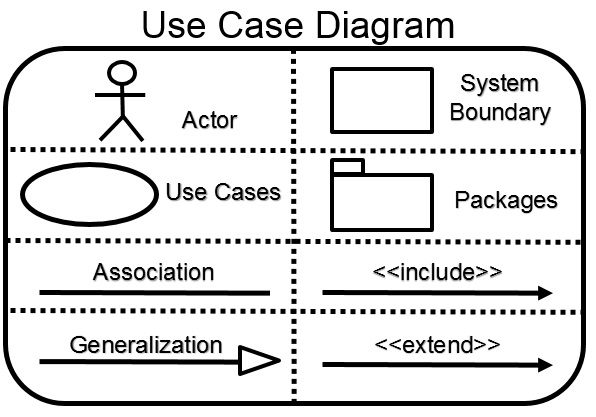


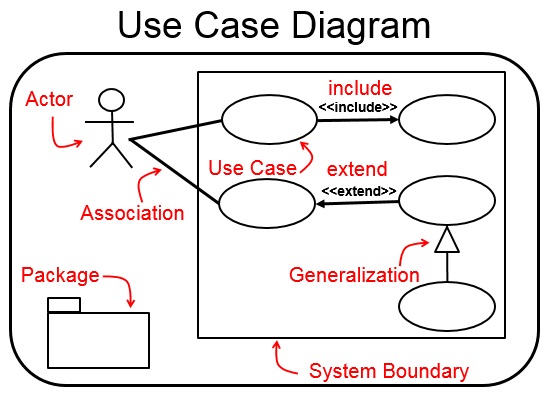


**Use Case Diagram**

|  |
| --- |
| A "Man with Chainsaw" is either: |
| - Wearing a SCARY Halloween costume  - a VIOLENT and SCARY Zombie Defender  - a VIOLENT and SCARY Psycho Killer  - has an OCCUPATION of Lumberjack  - has an OCCUPATION of Juggler |

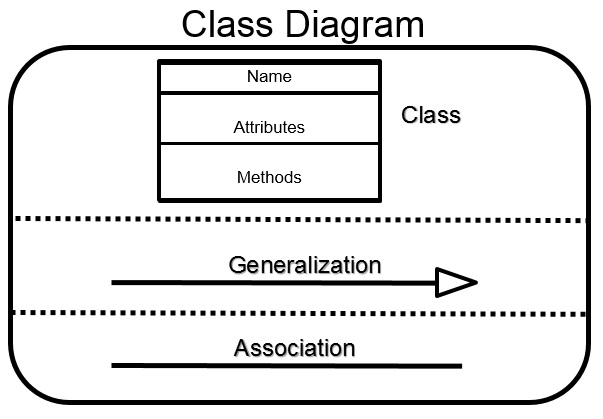
|  |
| --- |
| The "Passenger Service“ system: |
| The PASSENGER actor can use the "Passenger Service“ system to choose between:  (1) Check-In (with the help of a CHECK-IN REPRESENTATIVE actor)  (2) Automated Check-In  (3) Express Check-In  (4) Boarding  Both the CUSTOMS OF THE DESTINATION AIRPORT actor and the BAGGAGE TRANSPORTATION actor will be “Requesting a Passenger List” from the “Passenger Service” system. |

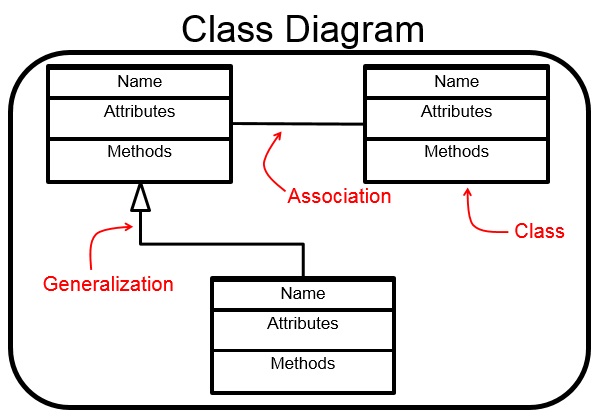




|  |
| --- |
| The ON-LINE SHOPPING System: |
| There is a general WEB CUSTOMER actor, and two instances of it, the REGISTERED CUSTOMER actor and the NEW CUSTOMER actor.  When using the ONLINE SHOPPING system, a REGISTERED CUSTOMER can "View Items" and "Make a Purchase".  When using the ONLINE SHOPPING system, a NEW CUSTOMER can also "View Items", but they cannot yet "Make a Purchase" as they have to add their name to the "Client Register".  "Make a Purchase" is a Use Case that has to include two other use cases: "View Items" and "Checkout".  "View Items" requires two external actors, both an "Authentication" actor and an "Identity Provider“ actor. "Checkout" requires three external actors, an "Authentication" actor, an "Identity Provider“ actor, and a "Paypal" actor. "Client Register" requires the "Authentication" actor. |

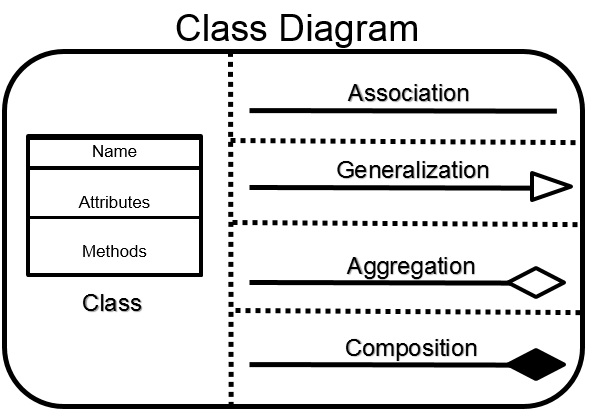
|  |
| --- |
| The BANK system: |
| In the BANK SYSTEM, there is a CUSTOMER actor who can do the following:  (1) "Open an Account" (with the help of a BANK EMPLOYEE actor)  (2) "Deposit Funds“, which will include "Update the Balance“.  (3) "Withdraw Funds", which will “include Update the Balance".  The "Deposit Funds“ will check if the balance exceeds £10,000, and based on that CONDITION, a “Calculate Bonus” will occur. Also if the Account is over 55 years , a “Calculate Bonus” will occur.  As well as a regular CUSTOMER, there is a FOREIGN EXCHANGE CUSTOMER, who can do everything a CUSTOMER can do, as well as bring able to "Convert Currency". |

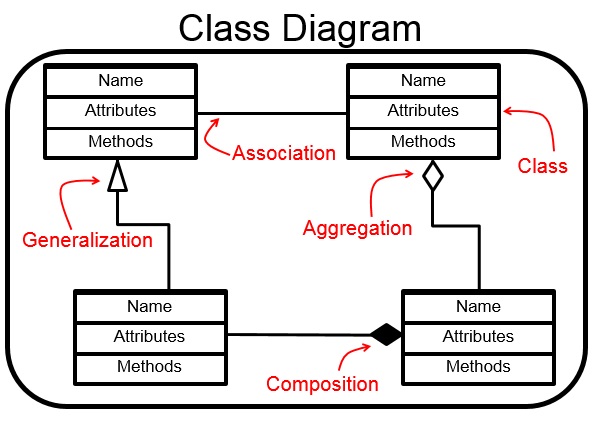




|  |
| --- |
| The ORDER PROCESSING System: |
| * CUSTOMER(a:name, address, m:creditRating()) * ORDER(a:date received, isPrepaid, number, price, m:dispatch(), close()) * CORPORATE CUSTOMER(a:contactName, creditRating, creditLimit, m:remind(), billForMonth()) * PERSONAL CUSTOMER(a:creditCard#, m:) * CUSTOMER (1 –– n) ORDER * CUSTOMER 🡨 CORPORATE CUSTOMER * CUSTOMER 🡨 PERSONAL CUSTOMER |

|  |
| --- |
| The STUDENT ENROLLMENT system: |
| * STUDENT(a:Name, Address, Phone, Email, Student#, AverageMark, m:isEligibleToEnroll(), getSeminarsTaken()) * SEMINAR(a:Name, Seminar#,Fees, m:addStudent(), dropStudent()) * ENROLLMENT(a:MarksRecieved m:getAverageToDate(), getFinalMark()) * PROFESSOR(a:Name, Address, Phone#, Email, Salary m:) * STUDENT (1 –– n) ENROLLMENT * SEMINAR (1 –– n) ENROLLMENT * STUDENT (n –– n) SEMINAR * PROFESSOR (1 –– n) SEMINAR |





|  |
| --- |
| The CUSTOMER TICKETS System: |
| * CUSTOMER(a:dateOfBirth, name m:) * TICKET(a:ticketCode, number m:) * COUPON(a:dateOfRedemption, Class, standBy, mealCode m:) * CUSTOMER (1 ------------ n) TICKET * COUPON (1,4 ---------<> 1) TICKET |

|  |
| --- |
| The ADDRESSBOOK system: |
| * ADDRESSBOOK(a:Introduction m:) * CONTACT(a:name, email m:) * ADDRESS (a:name, postalCode, city, country m:) * PHONE(a:internationalAreaCode, prefix, number m:) * CONTACT (1 ---------<\*> n) ADDRESSBOOK * CONTACT (1 –--main–-- 1) ADDRESS * CONTACT (1 –alternative– 1) ADDRESS * CONTACT (1 –--main–-- 1) PHONE * CONTACT (1 –alternative– 1) PHONE |