Use Case

FIT9138 IS Analysis, Design and Systems Thinking

User Stories

- **Scenarios are stories**
- User stories are used to capture **system requirements**
- They briefly describe:

  functional features

  a work-related task done by a user **to achieve a goal** or result

  what the system will do to support **a type of user or role**

Functional and Non-Functional System Requirements

- Functional requirements: activities that the system must perform (based on business rules and processes)

  E.g., for a payroll system, calculate taxes or GST

- Non-functional requirements: characteristics that the system must provide or support

  Usability, performance, reliability and security

MONASH University

FURPS

- **F**unctional

  the business functions/tasks that users must perform using the system

- **U**sability
- **R**eliability
- **P**erformance
- **S**upportability
- Security (Satzinger et al. 2015)

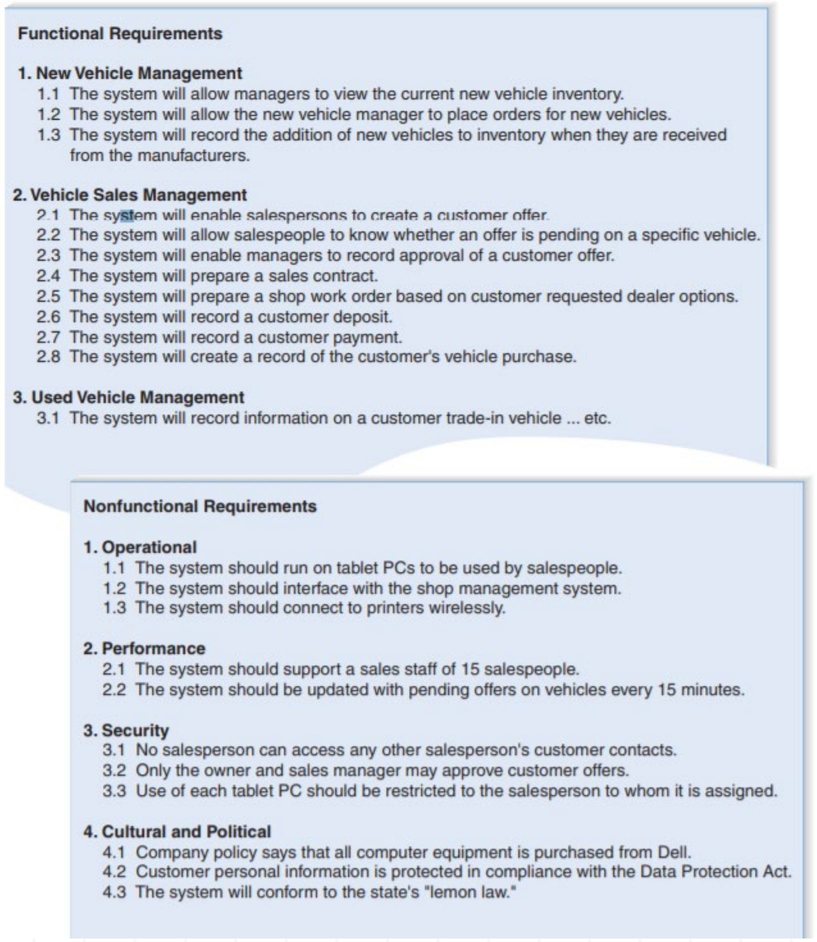
MONASH University

Grady and Caswell (1987)

FURPS+

- Design requirements
- Implementation requirements
- Interface requirements
- Physical requirements

MONASH University



Functional Requirements

1. New Vehicle Management
- 1.1 The system will allow managers to view the current new vehicle inventory.
- 1.2 The system will allow the new vehicle manager to place orders for new vehicles.
- 1.3 The system will record the addition of new vehicles to inventory when they are received from the manufacturers.

2. Vehicle Sales Management
- 2.1 The system will enable salespersons to create a customer offer.
- 2.2 The system will allow salespeople to know whether an offer is pending on a specific vehicle.
- 2.3 The system will enable managers to record approval of a customer offer.
- 2.4 The system will prepare a sales contract.
- 2.5 The system will prepare a shop work order based on customer requested dealer options.
- 2.6 The system will record a customer deposit.
- 2.7 The system will record a customer payment.
- 2.8 The system will create a record of the customer's vehicle purchase.

3. Used Vehicle Management
- 3.1 The system will record information on a customer trade-in vehicle ... etc.

Nonfunctional Requirements

1. Operational
- 1.1 The system should run on tablet PCs to be used by salespeople.
- 1.2 The system should interface with the shop management system.
- 1.3 The system should connect to printers wirelessly.

2. Performance
- 2.1 The system should support a sales staff of 15 salespeople.
- 2.2 The system should be updated with pending offers on vehicles every 15 minutes.

3. Security
- 3.1 No salesperson can access any other salesperson's customer contacts.
- 3.2 Only the owner and sales manager may approve customer offers.
- 3.3 Use of each tablet PC should be restricted to the salesperson to whom it is assigned.

4. Cultural and Political
- 4.1 Company policy says that all computer equipment is purchased from Dell.
- 4.2 Customer personal information is protected in compliance with the Data Protection Act.
- 4.3 The system will conform to the state's "lemon law."

Source: Dennis, Wixom, Roth (2012)

User Stories' Format

the **"role-feature-reason"** format:

As a <type of user>, **I want** <a system feature or to perform an action> **so that I can** <achieve a goal or a benefit>

Examples

- **As a** potential TV buyer, **I want** to read product reviews, **so that I can** decide which TV to buy.

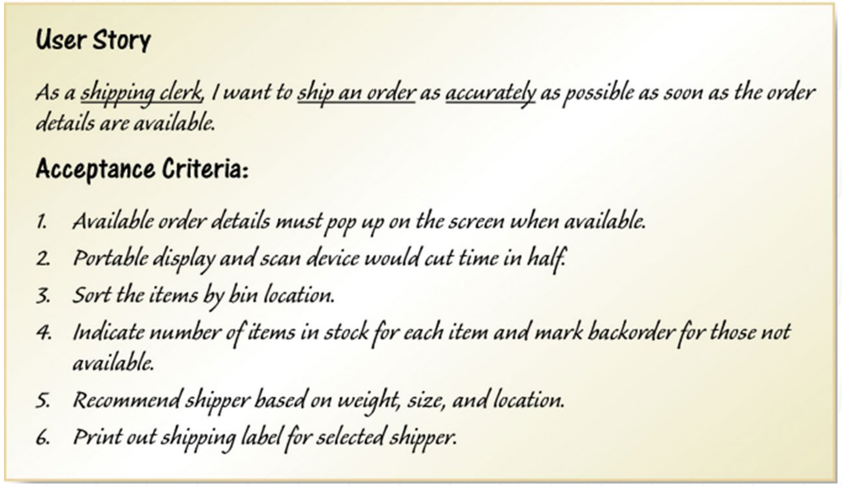
- **As a** hotel guest, **I want** to be able to cancel my reservation at any time, **so that I can** avoid losing all the money if an incident occurs.

- **As a** user of credit card app, **I want** to view the details of my transactions, **so that I can** easily check if there is any suspicious transaction.

User Stories

- **Acceptance Criteria:**

  Identify the features (conditions) that must be present at the completion of the task to satisfy the user
  Should be clear, concise and easy to understand



User Story

As a <u>shipping clerk</u>, I want to <u>ship an order</u> as <u>accurately</u> as possible as soon as the order details are available.

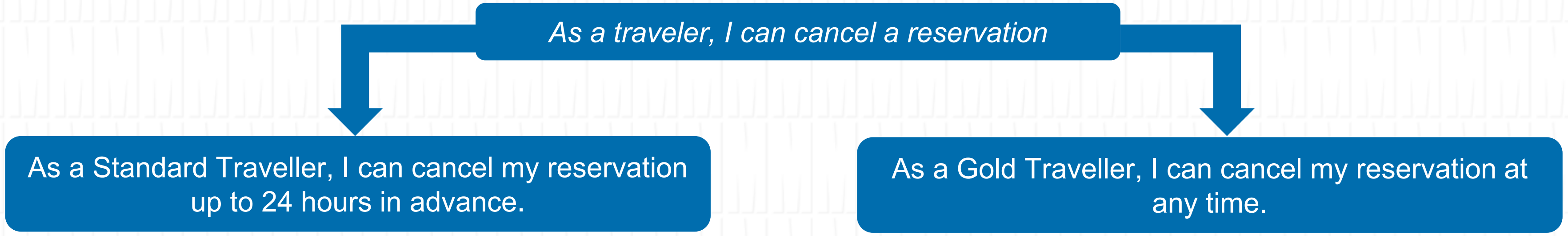
Acceptance Criteria:

1. *Available order details must pop up on the screen when available.*
2. *Portable display and scan device would cut time in half.*
3. *Sort the items by bin location.*
4. *Indicate number of items in stock for each item and mark backorder for those not available.*
5. *Recommend shipper based on weight, size, and location.*
6. *Print out shipping label for selected shipper.*

MONASH University

Different Levels of Details - Epics

- User stories can be grouped into an epic
- User stories can be split into smaller user stories


As a traveler, I can cancel a reservation
As a Standard Traveller, I can cancel my reservation up to 24 hours in advance.
As a Gold Traveller, I can cancel my reservation at any time.

MONASH University

Use Cases

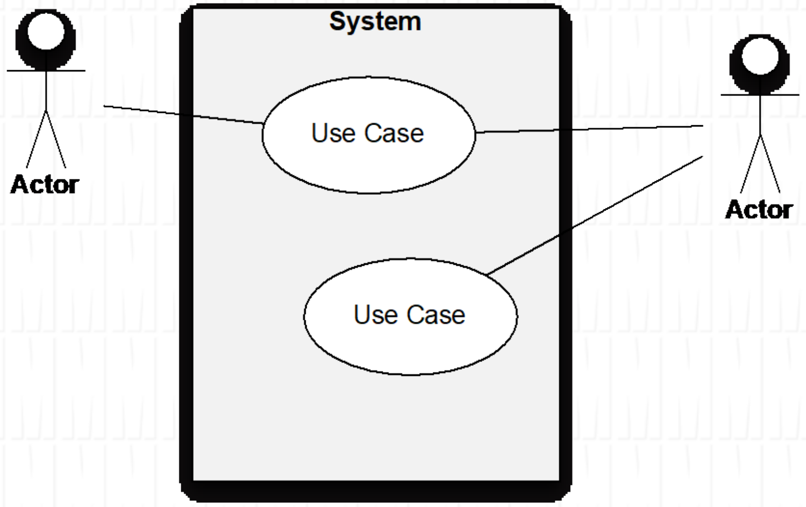
- **Use cases** are used to represent:
    - the **goals** that the **actors** try to achieve by using the system
    - **key activities** or tasks within a system
    - **functional requirements** of a system
    - the system's **reactions** and **responses**
        - E.g., create customer account, update details, add product

- **Actors** (users) are outside the system boundary box
    - An actor can be a person or another system
    - Usually, the role played by the user

MONASH University

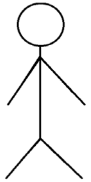
Use Case Diagram

- Graphical models to summarise and describe information about the use cases, actors and their interactions with the system
- It uses sufficiently descriptive verbs for use cases, and order them logically



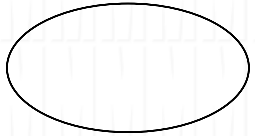
Use Case – Main Elements

- The actors that the system you are describing interacts with



- The rectangular: the system boundary

- The use cases, or activities/services that the system knows how to perform



- The lines that represent interactions between these actors and use cases

Actors

- The primary actor is the actor who initiates and triggers the use case and could receive a result
- The secondary actor responds or reacts to the use case or can benefit from the result
  - They can help the primary actor to achieve their goal

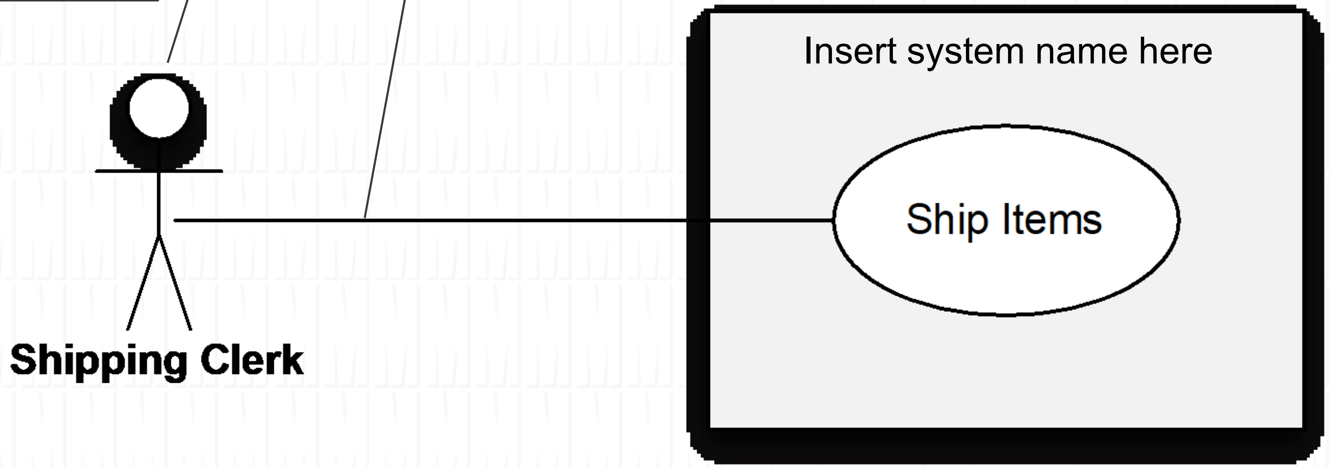
Simple Use Case with an Actor

Actor is a stick figure, usually meaning an actual person using the system

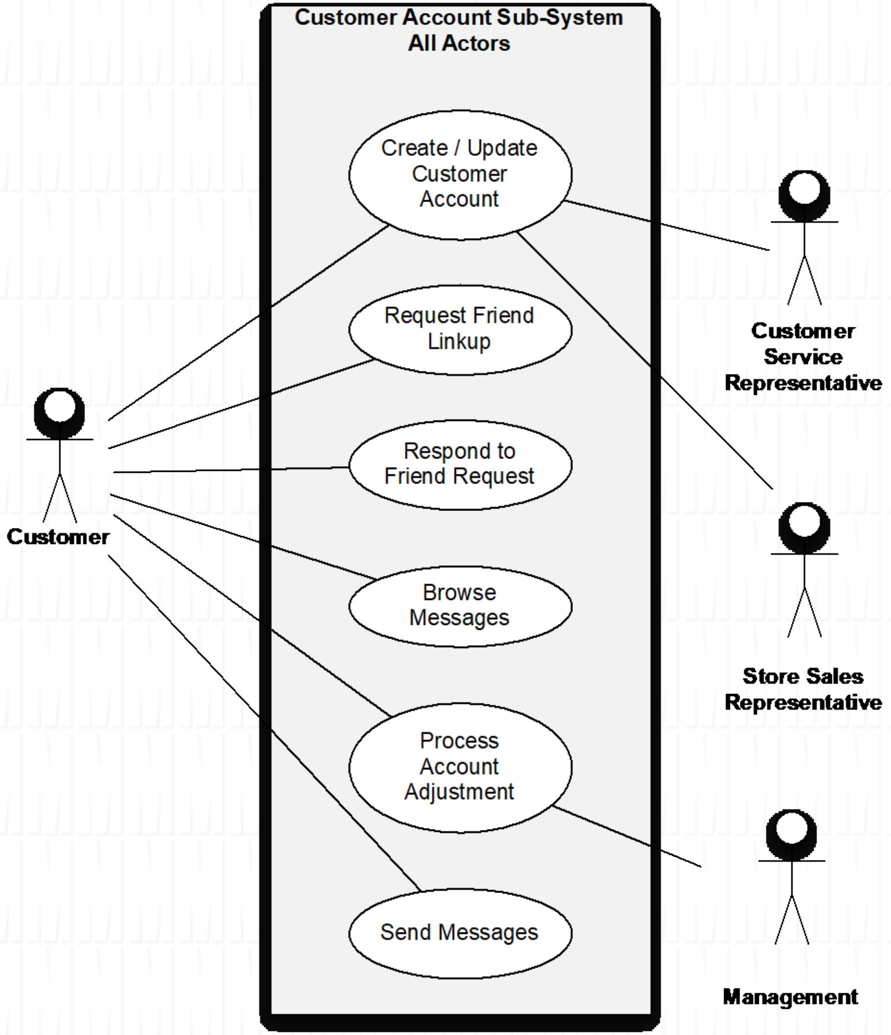
Relationship and association

Connecting line to show which actors participate in use cases

System boundary box (identifies the scope of the system)

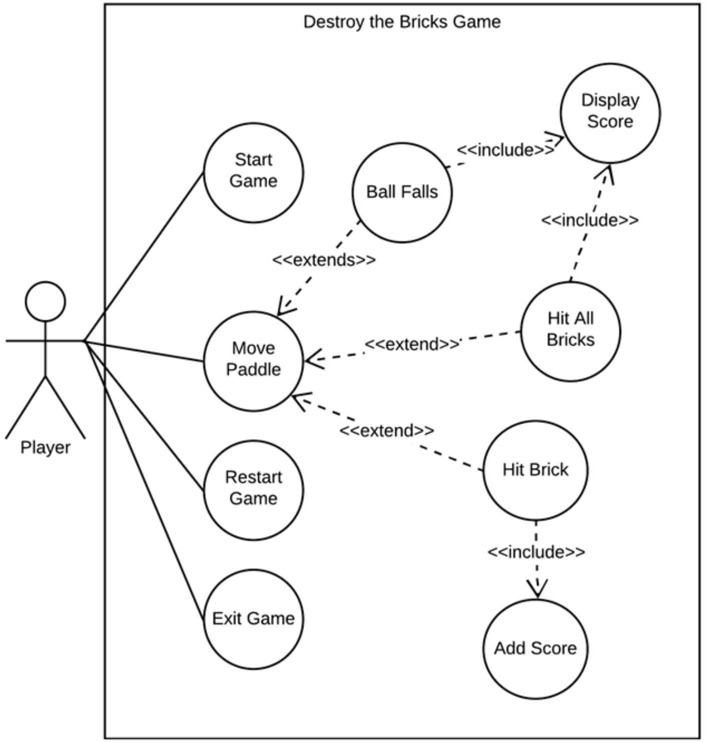


Use Cases for all Actors of Customer Account Sub-System


Customer Account Sub-System
All Actors
Create / Update Customer Account
Request Friend Linkup
Respond to Friend Request
Browse Messages
Process Account Adjustment
Send Messages
Customer
Customer Service Representative
Store Sales Representative
Management

MONASH University

Use Cases – Brick Game



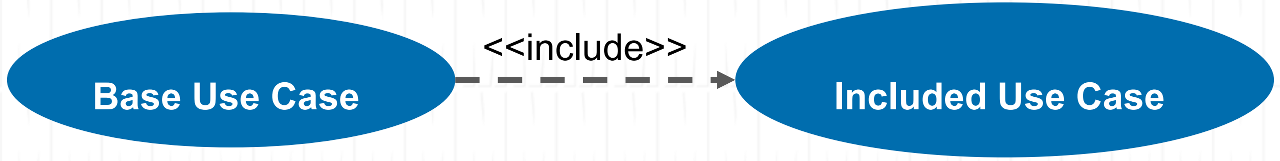
Relationships

- Association (the basic relationship)
- Include
- Extend
- Generalization

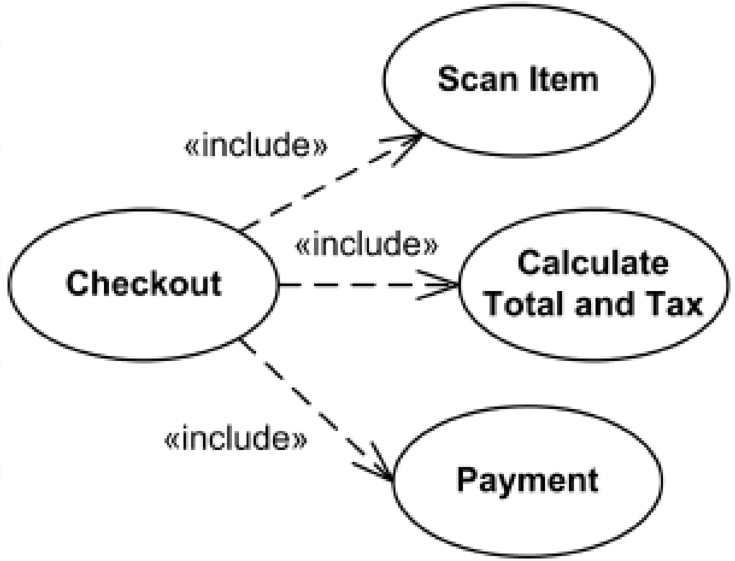
MONASH University

The <<include>> relationship

- Documents situation where one use case requires another use case in order to be completed
- The ‘include’ relationship shows that the dependency of one use case on another use case
- **Whenever a base use case is initiated, the included use case must be executed**
- It is represented using a dashed line from the base use case to another use case



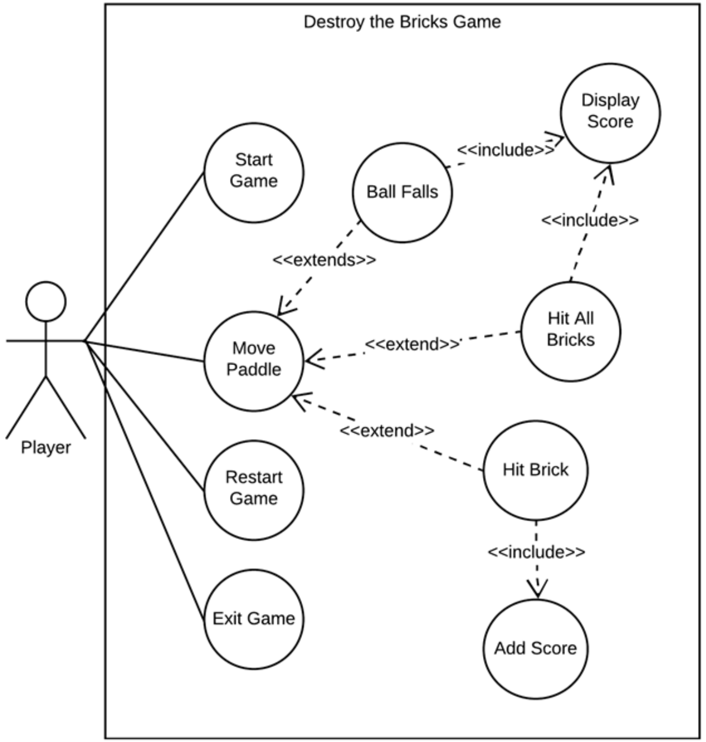
Example of <<include>> Use Cases



MONASH University

Source: https://www.uml-diagrams.org/use-case-include.html

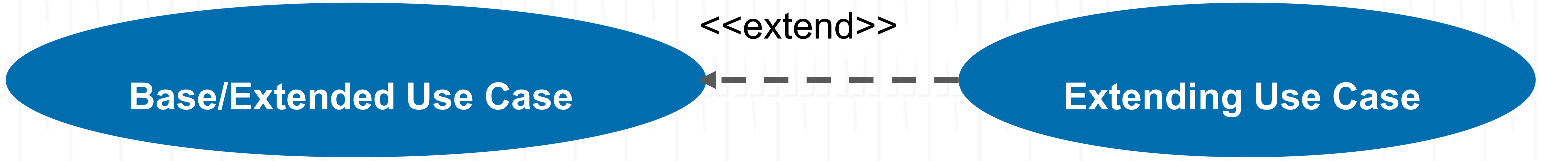
Use Cases – Brick Game



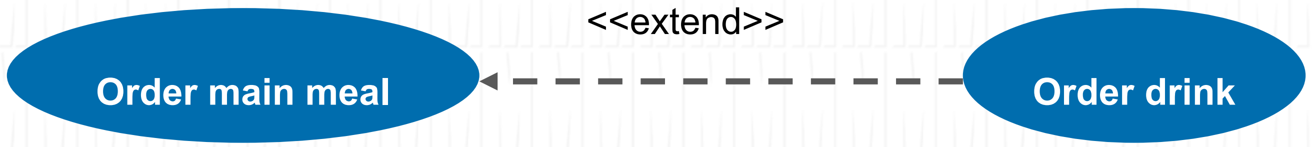
The <<extend>> relationship

- Extension is optional (happens sometimes)
- **The base use case is independent of the extending use case but the extending use case may not be meaningful by itself**

    E.g. in a parking pay station, after making the payment, you can obtain a receipt (Obtain Receipt extends Make Payment)



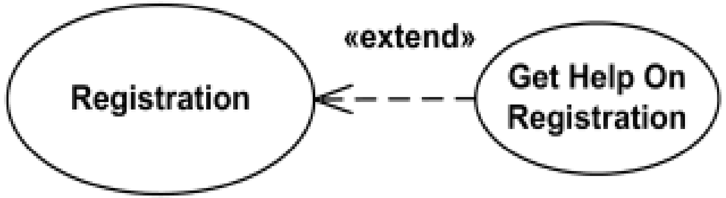
Example of <<extend>> Use Case

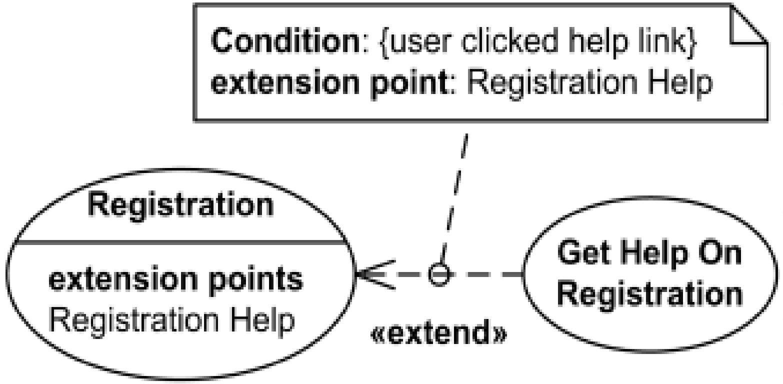


MONASH University

The <<extend>> relationship

- Registration use case is meaningful on its own, and it could be extended with optional Get Help On Registration use case
- Additional details can be provided to conditionally extend a use case

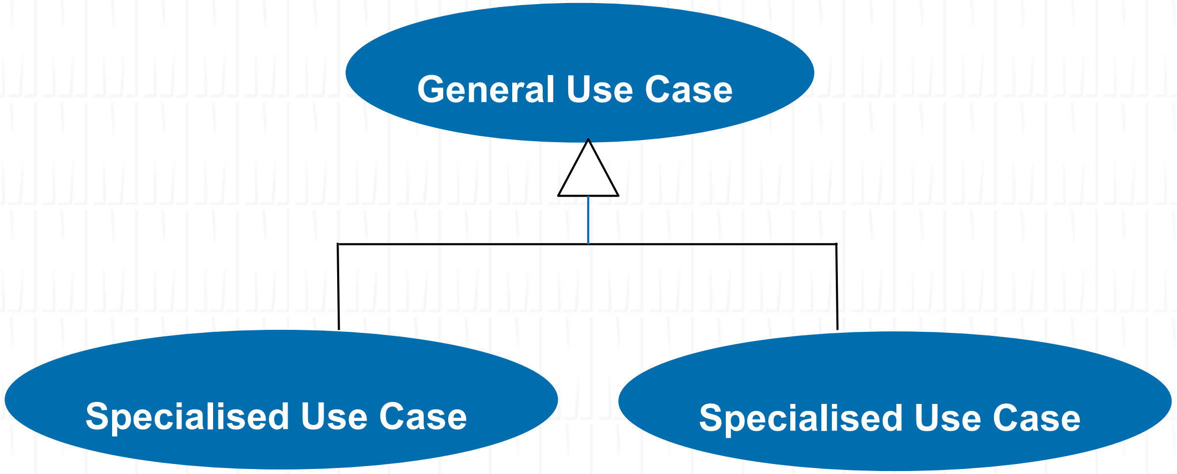




The Generalisation relationship

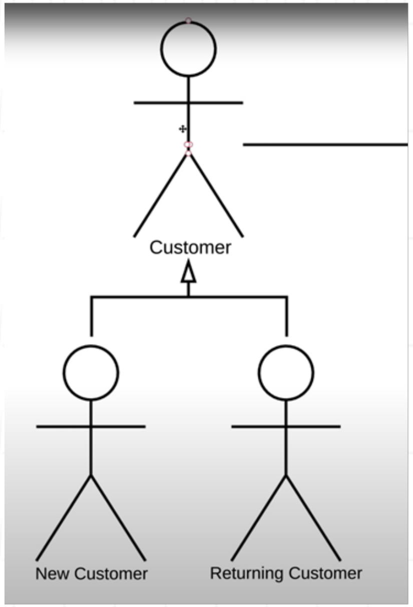
- It is similar to the concept of inheritance
- The specialised use cases share common behaviour of the general use case but each add different and specific behaviour

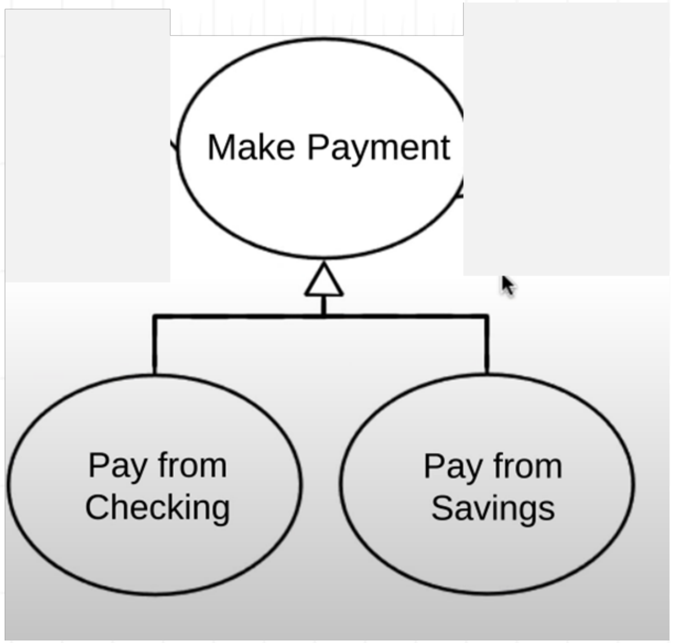
  E.g. Make Payment can have two specialised use cases of Pay by CC, Pay by PayPal



MONASH University

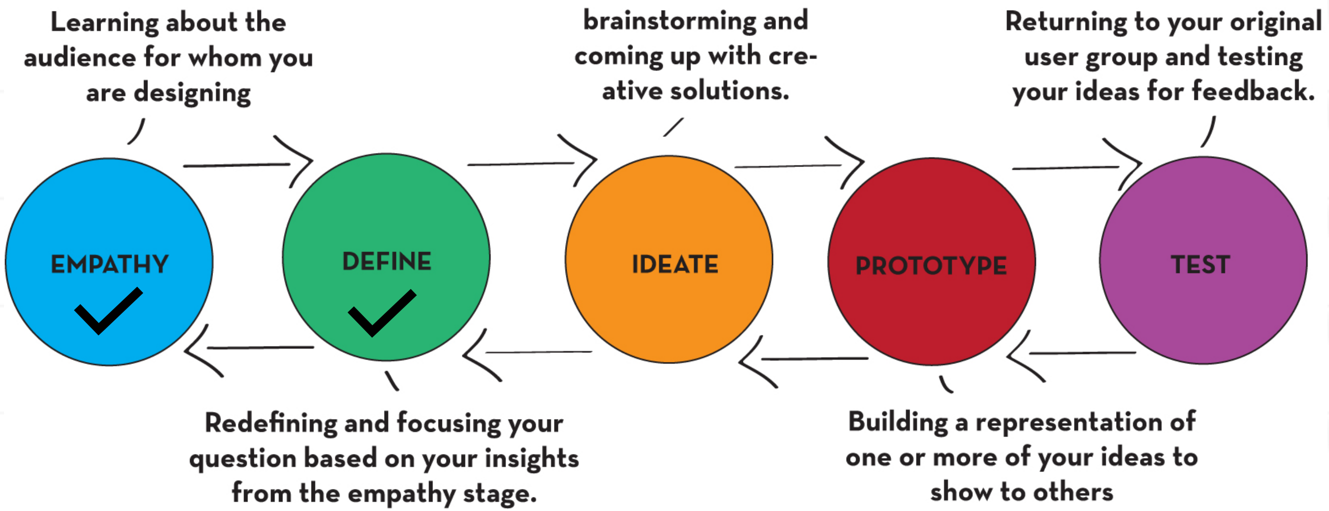
Example of Generalisation





Design Thinking – How?

Design thinking process is best thought of as a system of overlapping spaces rather than a sequence of orderly steps


Learning about the audience for whom you are designing
brainstorming and coming up with creative solutions.
Returning to your original user group and testing your ideas for feedback.
EMPATHY ✓
DEFINE ✓
IDEATE
PROTOTYPE
TEST
Redefining and focusing your question based on your insights from the empathy stage.
Building a representation of one or more of your ideas to show to others

MONASH University

http://www.mountvernonschool.org/upperschool/2012/03/design-thinking-summit-today/