需求分析與UML

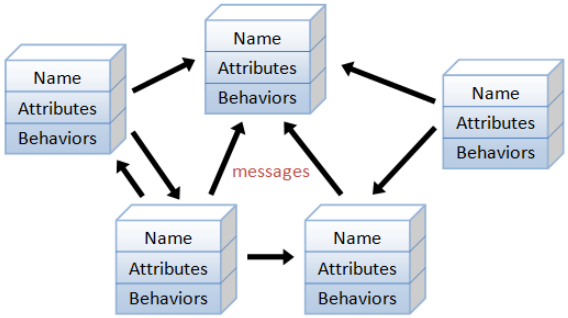
- 物件導向之系統開發過程是一種反覆的程序，主要包括需求分析、系統分析與設計、系統實作等階段。
- 其中，需求分析主要以使用案例圖作為表達工具；而系統分析與設計主要以類別圖、物件圖、循序圖、合作圖、狀態圖、活動圖、元件圖與部署圖等表達。
- 物件導向應用一些重要的觀念包括物件、類別、封裝、繼承與同名異式及超荷等，使物件導向系統有別於結構化之系統。由於物件導向技術之引進，使軟體之開發與維護更有效率，亦提升了程式的再用性與可維護性。

物件導向的基本觀念

- 物件導向的重要基本觀念
  - 物件
  - 類別
  - 封裝
  - 繼承

物件導向的基本觀念

- 物件是物件導向的基本思維單位，物件是一個具有狀態、行為與識別的實體或抽象化概念，且其行為會影響其狀態者。
- 物件包括名稱、屬性及操作(或稱方法)三部分



An object-oriented program consists of many well-encapsulated objects and interacting with each other by sending messages

物件導向的基本觀念

- 類別
  - 類別是具有相同結構及行為的物件所組成的集合。一個類別是一種定義、樣板或模型，它描述一群具有相同特徵（例如，屬性、操作、語意）的物件。
  - 類別是物件經分類或抽象化後所得的結果，也就是將物件抽象化，剔除物件間的差異而只考慮其相同的性質後，將這些物件組成一個群體稱為類別。例如，人。
  - 類別使用一組相同的屬性和操作來描述一個或多個物件，包含如何從類別中建立新的物件。類別有時亦稱為物件類型，類別中的任一物件稱為該類別之案例，因此物件與案例常被互用（視為同義詞）。

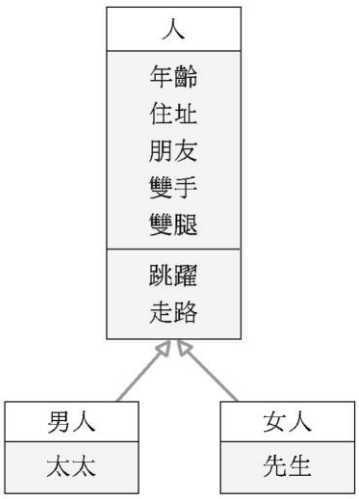
人
年齡 住址 朋友 雙手 雙腿
跳躍 走路

物件導向的基本觀念

- 封裝
  - 物件導向技術將資料及操作此資料的所有方法包裝成一個物件，稱之為「封裝」。封裝所形成的物件，其結構可分為兩部分，一是定義物件外觀行為的介面部分；另一則是存放抽象化的結果以及如何達成外觀行為的實作部分。
  - 封裝將物件的實作細節隱藏，使其與外界環境隔離，而只允許該物件所包含之操作修改其資訊，稱為資訊隱藏。物件之操作執行物件之行為，並可修改物件之資訊。
  - 使用物件時，僅需知道物件提供何種操作，而不需知道其內部之資訊或行為是如何表達或執行。外界之物件僅能透過訊息傳遞，要求該物件提供服務，而無法逕自改變該物件之資料內容。
  - 封裝之特性使物件導向的系統較容易維護。

物件導向的基本觀念

- 繼承
  - 繼承是類別間之關係，在此關係某類別之資料結構與行為可供其係中之類別分享。
  - 繼承之特性可利用一般化與特殊的原則，萃取相關子類別的相同性和操作，並將之歸類為一個父別來達成。
  - 在物件導向技術中，繼承概念之用對軟體工程有革命性之影響，達成程式碼再用與減少重複描述產生可靠度較高的軟體等。



需求分析與UML

- 開發一個系統通常為團體工作，必須多人合作方能完成任務，在開發過程當中需要使用文字、符號及圖型輔助說明設計的原意，使用UML是目前系統開最常見的方式
- UML是一種規格化、視覺化及文件化的軟體塑模語言，在系統塑模的實作面，需求分析主要以使用案例圖、活動圖、資料辭典等作為表達工具；而系統分析與設計則主要以類別圖、物件圖、循序圖、合作圖、狀態圖、活動圖、元件圖及部署圖等作為表達工具。

需求分析與UML

- 使用案例圖
  - 在需求分析階段用來描述用戶需求，從用戶角度描述系統的功能，並指出各功能的執行者，強調誰在使用系統，系統完成什麼功能。
  - 描述系統的行為者與系統間之互動行為與關係。從內部觀點來看，可描述系統做什麼？從外部觀點來看，可描述行為者與系統如何互動
- 類別圖
  - 定義系統中的類別，描述類別的內部結構以及類別與類別的關係，主要用於描述系統靜態結構
- 物件圖
  - 物件圖是類別圖的一個實例，描述了系統在具體時間點上所包含的物件以及物件之間的關係

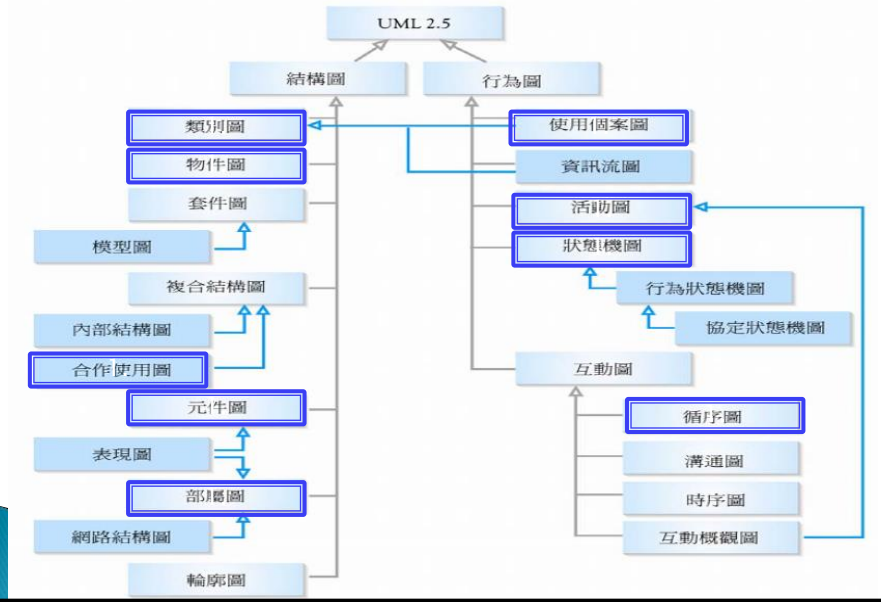
需求分析與UML

- 狀態圖
  - 用以表達物件在其生命週期中的狀態變化。描述物件所有可能的狀態以及事件發生時的狀態轉移條件。
- 活動圖
  - 活動圖是狀態圖的一種變異，用來表達涉及於執行某一作業行為中之活動。
- 循序圖
  - 用以描述系統運作時物件間的互動行為，且著重以時間之先後順序為主軸，以表達物件間的訊息傳遞與處理程序。一個循序圖會有一個與之對應的合作圖，但表達的重點與方式不同。

需求分析與UML

- 合作圖
  - 描述相關物件間之和合作關係，並能同時展現物件間的資料流程、控制流程與訊息傳遞的活動。
- 元件圖
  - 用以說明系統設計過程各類別與物件的配置，以及敘述軟體元件間的組織架構和相依關係。
- 部署圖
  - 用來呈現系統實作的模型，所謂的實作包含了各軟體元件的結構以及執行時（runtime）的元件布置情形。

需求分析與UML



UML開發工具

- IBM Rational Software Architect 是一個用於建模和開發的工具。它可使用統一建模語言 (UML) 來建模(畫UML圖)，並可以將設計的UML圖轉換成為程式碼(Java或 C++)，亦具備反向工程將程式碼轉換成UML圖的機制，可幫助簡化應用程式和服務的創建過程，提高生產力並加速開發。
- Enterprise Architect 是由Sparx Systems公司開發的 Shareware 軟體。提供軟體的正反向工程 (類別圖與程式碼的正反向對映)與自動化文件產出功能。

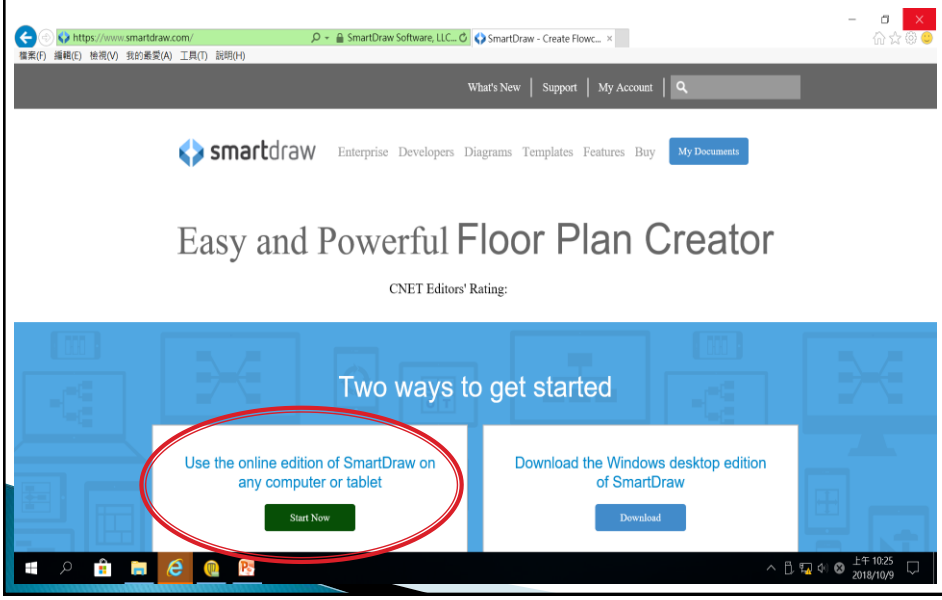
UML繪圖軟體

- Visio是Microsoft發行的一種繪圖軟體，提供UML圖形繪製
- creately 是一個線上圖表製作軟體，擁有流程圖、心智圖、資訊圖表的專業有創意圖表模板範本，可以在製作圖表時直接套用
- Diagram Designer 是一款簡單的向量繪圖工具，可用來創作流程圖、圖表、UML類別圖、簡易插圖、電路圖等圖形
- ArgoUML是由tigris團隊發行的一套免費UML開發工具，提供創建符合UML格式的文件

UML繪圖軟體

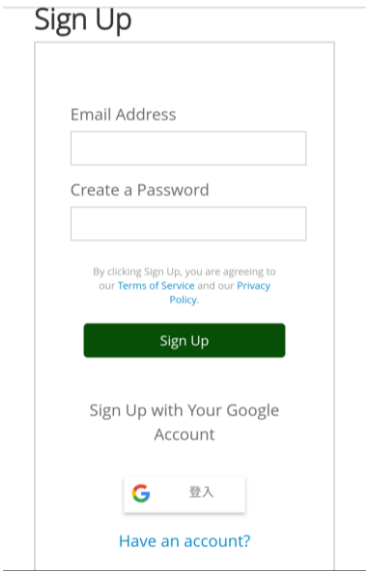
- **SmartDraw**是一套由SmartDraw.com開發的繪圖軟體，可以用來畫流程圖、時間圖等不同形式的商業圖表。
- **SmartDraw線上軟體**
  https://www.smartdraw.com/

UML繪圖軟體



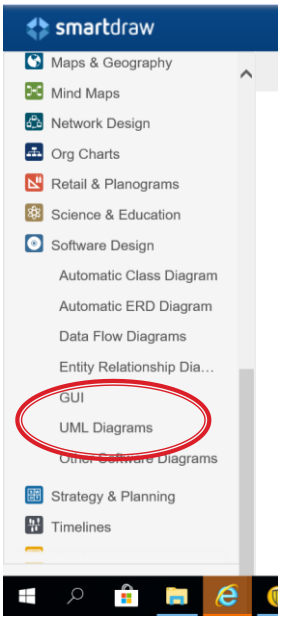
UML繪圖軟體

- 七天免費試用期
- 使用google 帳號登入



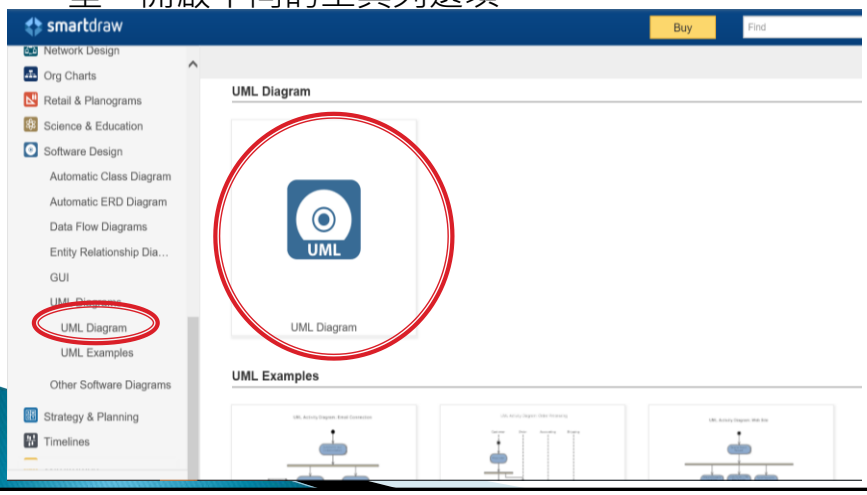
UML繪圖軟體

- 當執行SmartDraw後，就會先開啟流程圖的類型選項，其中有一般流程圖、地圖、網路圖、組織圖、軟體設計圖等等的類型
- 使用SmartDraw可繪製UML各種圖形，亦可套用範例檔製作



UML繪圖軟體

- 選取UML Diagram，SmartDraw會依所選擇的類型，開啟不同的工具列選項。

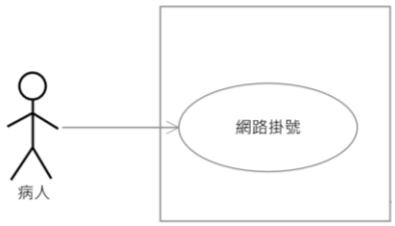


UML繪圖軟體

- 進到UML視窗後：
  - 只要拖曳工具列上的圖形鈕，就可新增流程圖的圖形。
  - 在任一圖形上按兩下滑鼠左鍵，就可輸入文字，中文字也可接受。
  - 先在圖形上按一下滑鼠左鍵就會在圖形邊緣出現黑色小點，在其上拖曳滑鼠左鍵，就可以做放大縮小的動作，在內部的圈圈做拖曳的動作，就可以旋轉該物件。
  - 在圖形上按一下滑鼠右鍵就可拉出修改該圖形的選單。

使用案例圖

- 使用案例圖是代表一個特定的企業功能或處理工作的步驟；一名參與者，對系統發出執行一個功能的請求，而啟動一個使用案例
- 例如在一個醫療辦公室系統中，一個「病人」，亦即參與者，可以做「網路掛號」，「網路掛號」即為使用案例，將「病人」與「網路掛號」間關係建立起來，即構成一個基本的使用案例圖

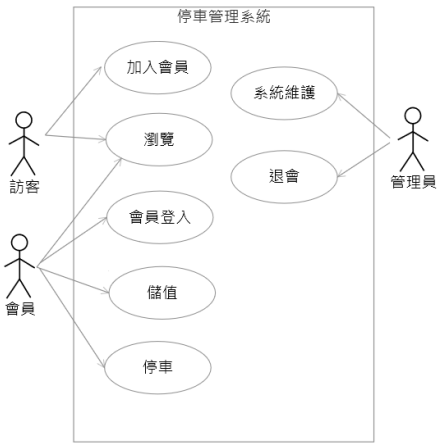


使用案例圖

- 參與者(演員)
  - 是環境中的人或事物(硬體設備、外部系統)，透過關係來描述與使用案例之間的互動，文字則描述參與者所扮演的角色
  - 辨識出系統的關鍵人員
- 使用案例
  - 一個使用案例是使用者透過介面可要求系統完成某一特定工作，可滿足使用者需求的功能
  - 辨識出系統的關鍵人員
- 系統邊界
  - 描述系統的範圍
  - 確認那些是這個系統要做的，避免客戶持續增加需求

使用案例圖

- 從使用者的角度描述停車管理系統的功能，並指出各功能也那些使用者，與系統間之互動行為與關係
- 使用案例圖是高階的需求描述，若要與使用者做進一步的需求確認，需要將每個使用案例做詳細的描述(Use Case Scenario Discription)



使用案例圖

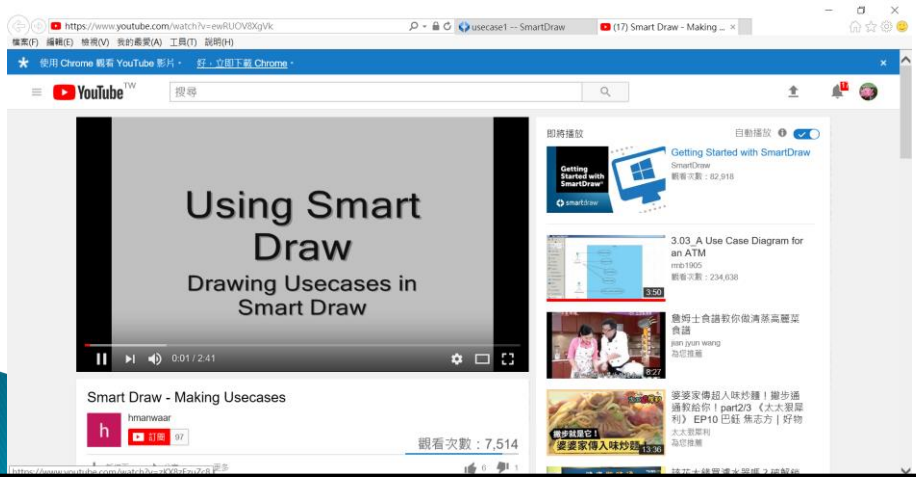
加入會員 使用案例情節敘述

- 名稱：加入會員
- 參與者：訪客
- 說明：

1.訪客閱讀會員條款
2.如果訪客同意會員條款
2.1訪客填寫會員資料
2.2系統偵測訪客是新會員還是舊會員
2.3法務徵信會員信用狀況
2.4系統寄發附有密碼的mail給訪客
2.5業務寄發卡片及會員相關文件
否則
2.6結束系統

繪製使用案例圖

- 教學影片
  https://www.youtube.com/watch?v=ewRUOV8XgVk



繪製使用案例圖

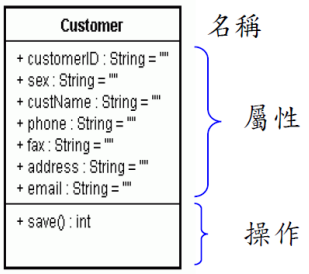
- 在Smart Panel標籤下分別選取Use Case、Actor-1圖形，拖曳至Page工作區適當位置
- 在任一圖形上按兩下滑鼠左鍵，就可輸入文字，並調整字體大小至16
- 上方功能區選取[Line]，繪製Actor與Use Case間的關係
- 上方功能區選取[Shape]，拖曳至Page工作區適當位置，選取[Design][Send to Back]，繪製系統邊界
- 上方功能區選取[File][Save as]，儲存檔案

類別圖

- 類別是打造物件的藍圖，由類別所建構出來的實例稱為物件(object)
- 一間房子的藍圖就比如是類別，而依據藍圖所蓋出來的房子就是實體
- 類別是物件經分類或抽象化後所得的結果，也就是將物件抽象化，剔除物件間的差異而只考慮其相同的性質後，將這些物件組成一個群體稱為類別。

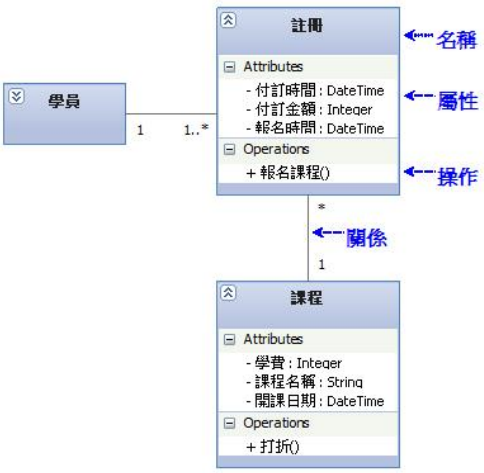
類別圖

- 類別用來描述一群具有相同屬性(attribute)與操作(operation)的物件。在UML中，類別是以一個長方形來表示， 且此長方形被分為三個部份。第一部分表示類別的名稱。第二部份表示類別所具有的屬性。第三部份則是表示類別所具有的操作。



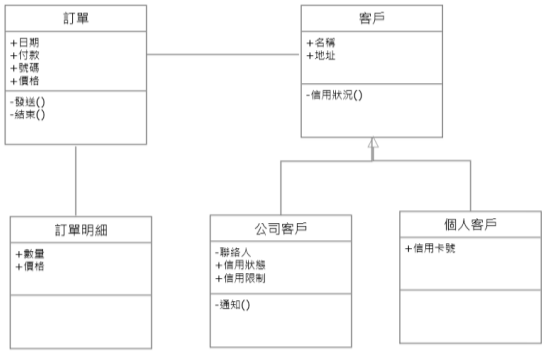
類別圖

- 類別圖主要是用以表示系統存在之類別及各類別間的靜態資料結構與邏輯關係
- 通常是透過檢視使用案例，描繪出出現的類別及與使用案例相關的操作
- 類別間的關係有：相依關係、一般化關係、結合關係、實現化關係



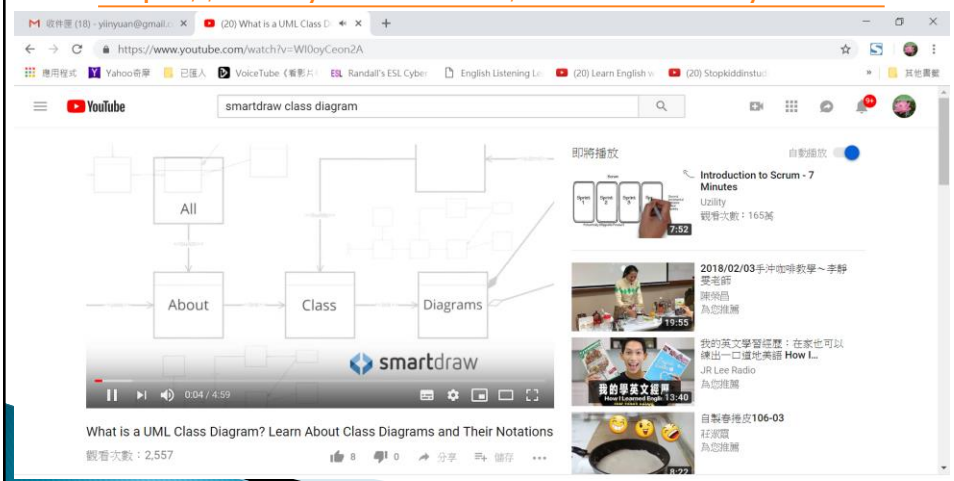
類別圖

- 客戶與公司客戶間，客戶與個人客戶間具有繼承關係，用實現的空心三角形箭頭來表示
- 訂單與客戶間、訂單與訂單明細間具有結合關係，優方彼此都知道對方的存在，用實線來表示



繪製類別圖

- 教學影片
  https://www.youtube.com/watch?v=WI0oyCeon2A

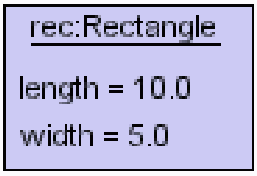


繪製類別圖

- 在Smart Panel標籤下分別選取Class-1圖形，拖曳至Page工作區適當位置
- 在Class-1圖形上按兩下滑鼠左鍵，就可輸入名稱；屬性及操作之文字，並調整字體大小
- 上方功能區選取[Line]，繪製Class與Class間的關聯
- 上方功能區選取[Line][Arrowheads]可調整箭頭樣式
- 上方功能區選取[File][Save as]，儲存檔案

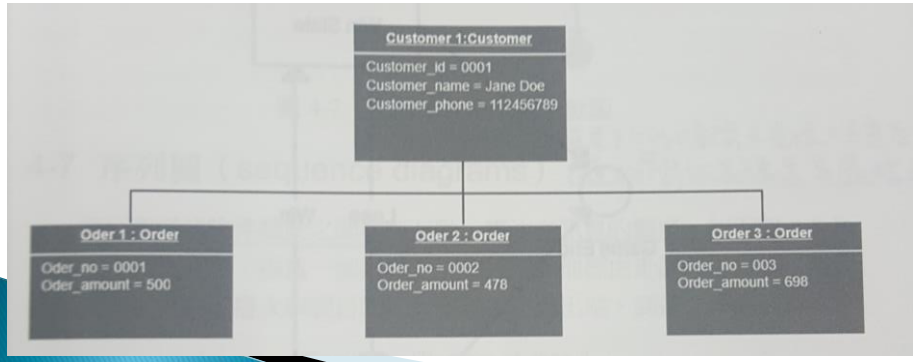
物件圖

- 物件圖就是類別圖的實例，其基本觀念與類別圖相同；物件圖描述的系統靜態結構式系統某一時間點的快照
- 物件圖包含參與的物件、相關屬性的屬性值、還有物件與物件之間的關係。例如下圖為表達一個長10.0，寬5.0的長方形物件，這個物件的名稱是rect，它的型態是長方形(Rectangle)



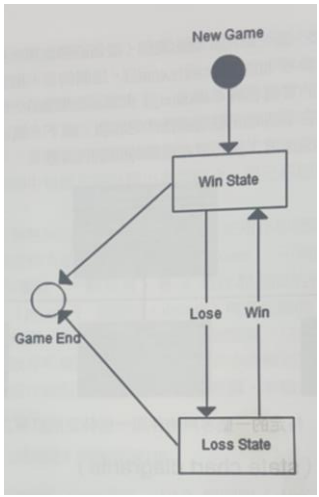
物件圖

- 客戶類別用一個實體名稱Customer_1來表達這個物件，也將其屬性賦予特定值，說明某個時點這個物件的狀態；這個客戶有三張訂單，也同時描述這三張訂單在這個時間點的狀態。



狀態圖

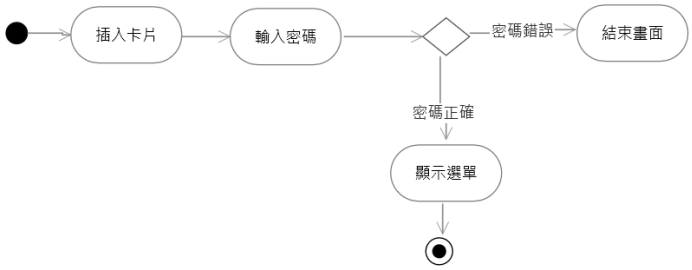
- 通常用來描述單一使用案例之內部的流程，或橫跨多個使用案例的流程
- 狀態圖非絕對必要，因大部分的系統流程在使用案例情節中就可以呈現
- 有助於釐清狀態之間轉移的變化情形



一個遊戲的狀態圖

活動圖

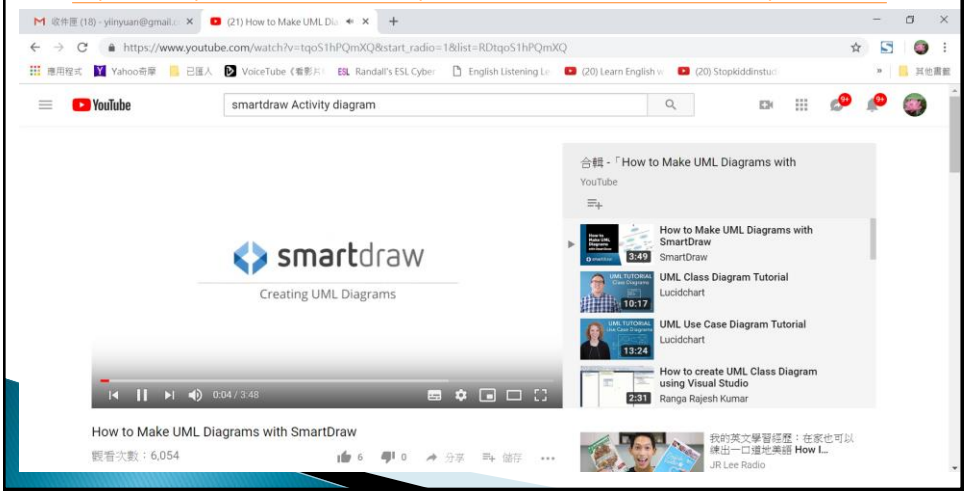
- 活動圖類似狀態圖，可用來描述某個使用案例或許多使用案例間或一個系統之作業流程或行為
- 活動圖亦可用表達某物件的操作、所涉及之循序或同步的活動
- 描述輸入密碼的活動圖



繪製活動圖

- 教學影片

https://www.youtube.com/watch?v=tqoS1hPQmXQ&start_radio=1&list=RDtqoS1hPQmXQ

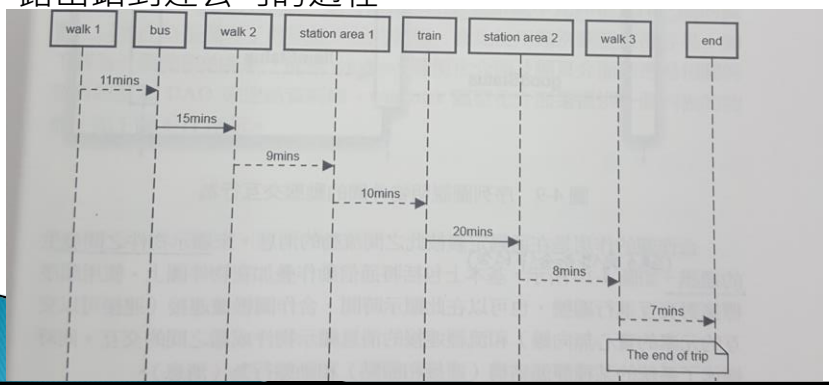


繪製活動圖

- 在Smart Panel標籤下分別選取Activity-1、Decision、Initial Node、Final Node圖形，拖曳至Page工作區適當位置
- 上方功能區選取[Line]，繪製圖形間的關係
- 在任一圖形上按兩下滑鼠左鍵，就可輸入文字，並調整字體大小至16
- 上方功能區選取[File][Save as]，儲存檔案

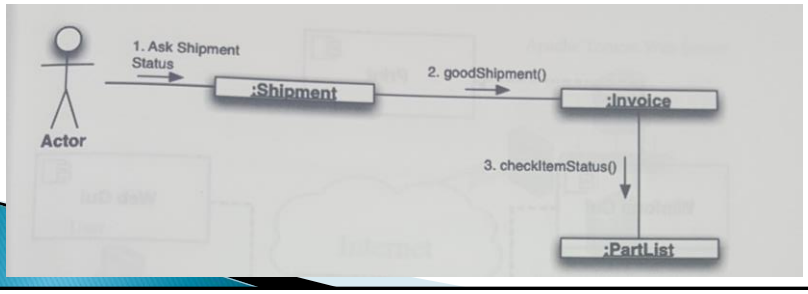
序列圖(循序圖)

- 序列圖描述物件彼此之間如何互動，著重以時間發生之先後順序來表達物件間的訊息傳遞之程序。
- 下圖為一個通勤上班的員工，經由走路、搭公車、再走入進入火車月台、搭火車到目的地火車站、走路出站到達公司的過程



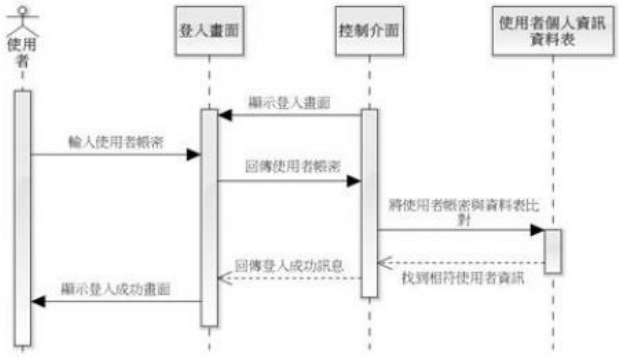
合作圖

- 合作圖和序列圖相同，都是在描述物件之間的動態交互行為。惟其重點在於顯示物件之間發生的通訊
- 合作圖的作用是在通過定義彼此之間流動的消息，來顯示物件之間發生的通訊
- 合作圖描述了系統的某種靜態結構(連接和節點)和動態行為(消息)



序列圖與合作圖之差異

- 循序圖有垂直象限，是依照表達訊息呼叫發生的時間順序，來描述呼叫的先後順序；循序途中另外還有，水平象限用來描述一個物件實體傳送資訊給哪一個物件實體。

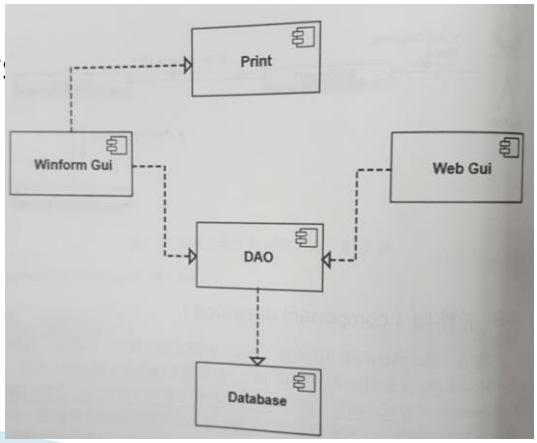


序列圖與合作圖之差異

- 合作圖較著重在表達物件間連結結構，並能同時展現物件間的訊息傳遞與處理程序

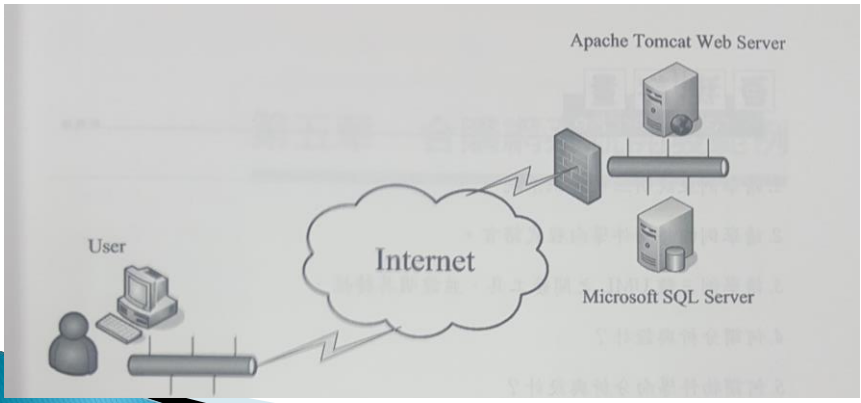
元件圖

- 用以顯示軟體系統中元件與元件之間的關係
- 以元件為基礎來了解軟體系統的架構，會比類別圖來的容易
- 右圖描述Windows圖形化介面及網頁介面都透過相關的資料庫引擎DAO來連接資料庫； Windows圖形化介面還提供一個列印功能



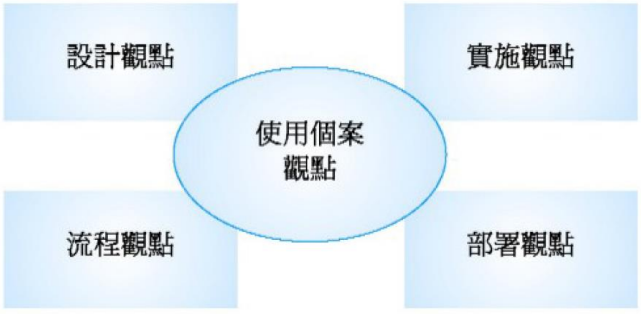
部署圖

- 部署圖描述系統運行的元件結構，顯示硬體元件的配置，並顯示軟體元件如何對應到這些節點上。



五個連鎖觀點的軟體系統架構

- 於系統塑模的概念面，以五個連鎖觀點的軟體系統架構，供參與者從不同的角度來檢視系統：



五個連鎖觀點的軟體系統架構

- 使用個案觀點 (Use Case View)
  - 由描述系統行為的使用個案組成，這些系統行為是以使用者、分析師、測試者的觀點來描述系統應做什麼及其操作情境，並不涉及實際系統的軟硬體架構。以使用個案觀點來描述高鐵站之自動售票系統為例，可說明系統如何支援使用者購票及購票之實際操作情境 (例如須先輸入購票資訊再行付款)。
- 設計觀點 (Design View)
  - 描述系統之類別、介面、合作情形及彼此間之互動方式。此觀點主要支援系統之功能面需求，意即著重表達系統應提供予使用者之服務。以設計觀點來描述高鐵站之自動售票系統，可說明此售票系統有選擇票種、目的地、車票張數、乘車日期時間與購票資訊等介面及車次資料等類別，並描述介面與類別間之互動及合作情形。

五個連鎖觀點的軟體系統架構

- 流程觀點 (Process View)
  - 用以描述系統內部的處理程序來說明系統之績效 (Performance)、產出 (Throughput) 與可擴充性 (Scalability)，而處理程序可能為平行或同步機制。以流程觀點描述高鐵站之自動售票系統，可說明乘客在購票過程中，系統讀取車次資料之處理程序；亦可說明若乘客選擇以信用卡付款，則系統與外部銀行連結之處理程序。
- 實施觀點 (Implementation View)
  - 表達系統版本的結構配置管理，描述系統由哪些獨立的元件與檔案組成，這些元件與檔案如何以不同方式組合可實際執行之系統。以高鐵站之自動售票系統為例，實施觀點可描述售票系統中所包含之元件，如計算車資之函式庫、列印車票之執行檔元件，並說明各元件於系統中之靜態結構關係。

五個連鎖觀點的軟體系統架構

- 部署觀點 (Deployment View)
  - 表達系統執行時，硬體拓樸上各節點 (Nodes) 之配置，並描述系統分析與設計的產出，及說明系統如何實作。以高鐵站之自動售票系統為例，此觀點可說明各站之自動售票機如何建置，例如售票軟體系統如何安裝至售票機、觸控式螢幕如何接收乘客輸入之購票資訊等。

五個連鎖觀點的軟體系統架構

表達工具	靜態面	動態面
使用個案觀點	➢使用個案圖 ➢活動圖	➢循序圖 ➢溝通圖 ➢時序圖 ➢互動概觀圖 ➢狀態圖 ➢活動圖
設計觀點	➢類別圖 ➢物件圖 ➢複合結構圖	
流程觀點	➢類別圖 ➢物件圖 ➢複合結構圖	
實施觀點	➢元件圖 ➢套件圖	
部署觀點	➢部署圖	