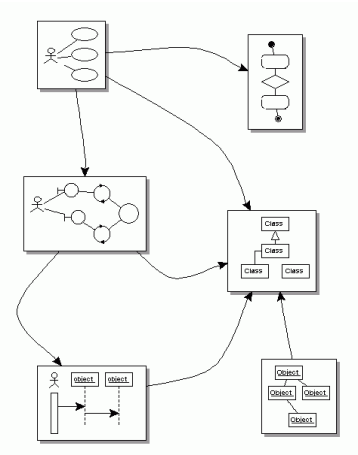
UML簡介_Use Case Diagram

資訊科技系
林偉川

UML簡介



2

UML

- UML是Unified Modeling Language的縮寫，中文翻譯為統一塑模語言。UML統合了物件導向方法論之各派不同的方法，提供了一致性的圖形語言做為開發系統的溝通媒介。
- UML是一種圖形語言

3

圖形的溝通工具

- 對於系統的使用者，可以用UML來表達系統的功能，讓使用者對於系統所能從事的工作有一個高層次的了解。
- 對於系統分析師，可以使用UML來做為討論系統架構的工具。
- 對於軟體工程師，可以利用UML來從事物件導向的系統分析與設計，塑模出物件之間的靜態以及動態關係。
- 系統的管理者也可以藉由UML來表達硬體或是軟體元件的佈署與配置情形。

4

塑模(Modeling)

- Why? 那麼塑模(Modeling)的意思呢？顧名思義，塑模就是塑造模型。為什麼要塑造模型？
- 應該都曾看過樣品屋, 可能曾在建築公司見過，或者是從電視上看到。為什麼建築師要蓋那些樣品屋？它又不能住人。同樣出現在許多的不同的工程領域中。為什麼飛機製造商要建造飛機的模型？為什麼造橋的工程師也要蓋橋的模型？這些模型到底有什麼做用？

5

塑模的原因

- 工程師建造模型來驗證設計不會出問題。並且，如果設計真的出了問題，損失的只是模型。飛機製造商藉由飛機的模型以及風動的測試來看看設計出來的飛機可不可以飛；造橋的工程師可以利用模型來測試橋樑可以承受的強風或是重量而不至於斷裂。建築師利用樣品屋可以知道到底有沒有人會想要買他設計出來的房子。

6

塑模的好處

- 建構模型比建構實物來得較容易，也較便宜。
- 模型可以用來模擬。如果出現了錯誤，也不至於造成重大的損失。
- 使用模型可以幫助學習。
- 使用模型是一種有效的溝通方式。
- 使用模型，可以用來表達不同層次的細節。

7

軟體塑模

- 透過塑模，可以對於即將開發的系統有更好的了解；利用塑模，可以幫助預見將來在系統開發時所會面臨到的問題，及早做修正。

8

塑模可以達成的四個目標

Booch等在書中指出了塑模可以達成的四個目標

- 模型幫助視覺化一個系統
- 模型允許詳述一個系統的結構或是行為
- 模型給出了指引建構系統的一個樣板
- 模型記錄所做的決定

9

4+1觀點

- 4+1觀點最早是由Philippe Kruchten於1995年在一篇論文中所提出。所謂的4+1觀點是用來做為塑模系統架構的一個藍圖。Booch等在他們所出版的UML使用手冊中定義了UML中的4+1觀點

10

使用案例觀點(Use Case View)

- 從系統外部的使用者角度，表達系統所提供的功能。(4+1中的1指的是使用案例觀點。)

11

設計觀點(Design View)

- 描繪出系統的靜態結構以及動態行為，以做為系統所應提供之功能的解答。因此，設計觀點圍繞在類別，介面以及物件的合作等等設計問題上。

12

處理流程觀點(Process View)

- 描繪出組成系統的平行以及同步機制之執行緒(thread)以及程序(process)。這個觀點強調系統的功能、延展性等非功能性需求。

13

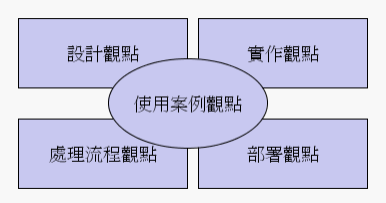
實作觀點(Implementation View)

- 以模組或元件來顯示設計觀點的物件是在那一個模組或元件中實作。

14

部署觀點(Deployment View)

- 描繪系統在執行時，各個組成元件的實際佈置與安裝。強調系統執行環境的硬體拓樸結構。



15

4+1觀點

- 對於4+1觀點中的每一個觀點，可以利用UML所提供的九種圖形來表達。對於每一個不同的觀點，又可區分為靜態及動態面，因此，這九種UML圖形在各觀點的應用時機可以利用下面表格來做一個歸納整理。

16

	靜態模型	動態模型
使用案例觀點	使用案例圖	互動圖、狀態圖、活動圖
設計觀點	類別圖、物件圖	互動圖、狀態圖、活動圖
處理流程觀點	類別圖、物件圖	互動圖、狀態圖、活動圖
實作觀點	元件圖	互動圖、狀態圖、活動圖
部署觀點	部屬圖	互動圖、狀態圖、活動圖

17

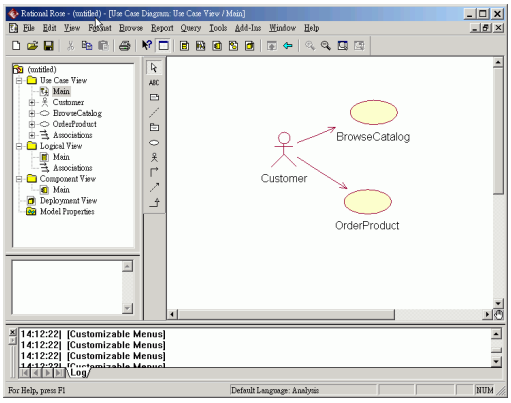
4+1觀點

- 利用不同的觀點來看系統是了解系統架構相當有效的方式。而利用RUP (Rational Unified Process)所提出之各種不同觀點，基本上可以將系統以下列的方式來看之
  - 功能觀點
  - 靜態觀點
  - 動態觀點
  - 部署觀點

18

UML工具

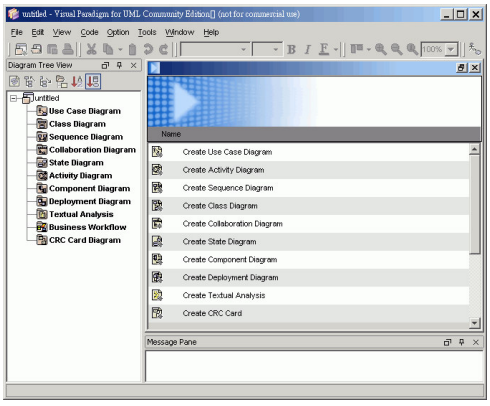
- Rational Rose



19

UML工具

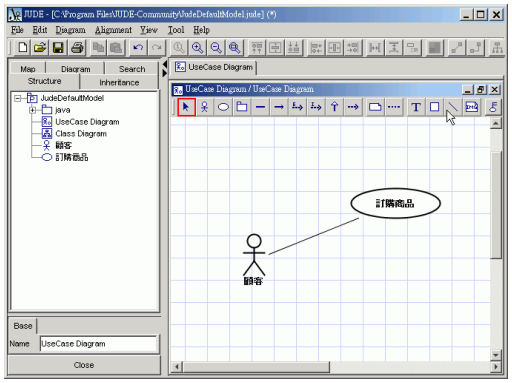
- Visual Paradigm



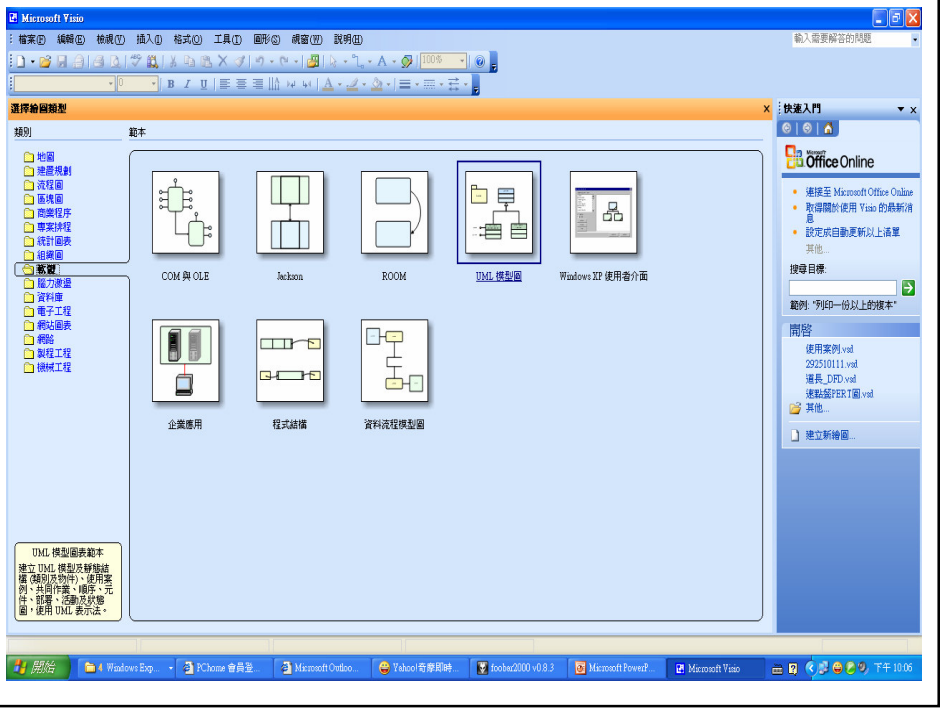
20

UML工具

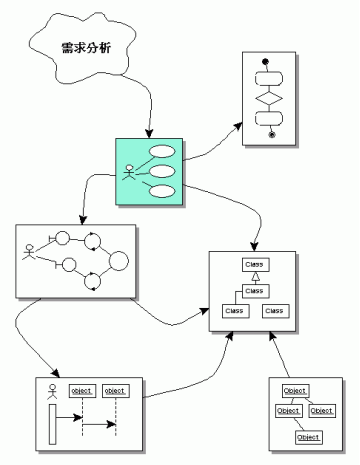
- JUDE



21



使用案例圖



23

4+1觀點

- 利用不同的觀點來看系統是了解系統架構相當有效的方式。而利用RUP所提出之各種不同觀點，基本上可以將系統以下列的方式來看之
  - 功能觀點
  - 靜態觀點
  - 動態觀點
  - 部署觀點

24

目的

- 使用案例圖的目的
  - 塑模出系統應該做什麼(what)，而不是如何(how)做。
  - 從高層的角度來看系統的功能。
  - 從系統的外部來看系統的用途。

25

使用案例的意思

- 什麼是使用案例(Use Case)？從字意上來說，使用案例指的是可以使用(Use)系統來處理的個案(Case)。這句話可以換個方式來講：系統提供的功能是什麼(what)？因為系統可以處理的事情表示系統所具有的功能

26

塑模系統的功能

- 使用案例圖用來塑模系統的功能、系統的用途。
- 使用案例圖描述系統的行為。
- 使用案例圖的繪製工作就是要從需求分析文件中找出actor、使用案例以及使用案例之間的關係。

27

符號介紹

- 演員(actor)

  - 演員這個字是從英文直接翻譯過來。在UML中，與系統互動的使用者稱為actor。把actor當成是使用者、參與者、或是角色也無妨。在不同的討論環境下會相互交替使用不同的意思。一般來說，還是直接使用其英文字actor。

28

Actor代表角色

- Actor代表的是一個角色。它不一定是代表真正的人，更不會代表哪一個特定的人。它要表達的是一個與系統互動的角色。在戲劇的腳本中，一個角色可以由不同的人來擔任。如果玩過角色扮演遊戲，就應該體會出它所代表的意思了。
- 凡是會與系統互動的都可以是actor。一個使用案例可以跟許多個actor產生互動。一個使用案例不會單獨存在。它至少會跟一個actor有互動。

29

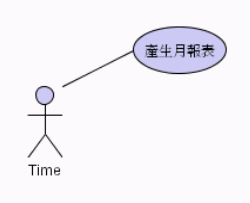
可能的Actor

- 常見的Actor
  - 使用者
  - 系統
  - 時間

30

Actor:時間(Time)

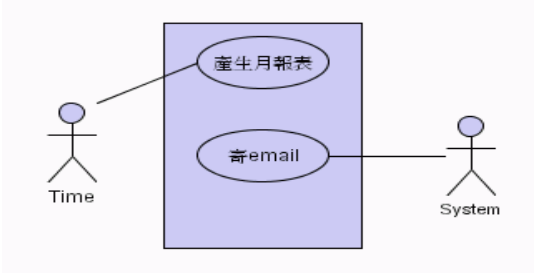
- 時間(Time)
  - 有些時候，驅動系統執行某項功能的原因是因為時間到了。例如，產生月報表這個功能。如果在系統需求文件中有記載著系統必須提供此功能， 並且要求這個事件是時間一到便會驅動系統自動來執行，那麼可以將這個actor命名為: Time。



31

Actor:系統(System)

- 系統(System)
  - 另外，在某些情況下，一些特定的功能是由系統自己來驅動。例如，當一筆訂單確認後，系統必須主動寄發email給顧客這個事件。對於這個事件，系統(system)就是actor。



32

角色的找法

- 對於角色的找法，嘗試著回答下列的問題
  - 誰要使用到此系統？
  - 誰提供這些資訊？
  - 誰需要這些資訊？
  - 誰可以改變這些資訊？
  - 誰可以刪除這些資訊？
  - 再次強調，”誰”不一定是代表哪些人。

33

使用案例(Use Case)

- 使用案例基本的介紹：可以使用(Use)系統來處理的個案(Case)。這樣的解釋很類似Larman在書中對Use Case的定義：”They are stories or cases of using a system”。

34

事件與使用案例

- 事件表是用來表達系統應該提供的功能。因此，使用案例與事件會有對應上的關係。
- 如果需求文件是以事件表的方式寫成，可以直接就將事件名稱用到使用案例的名稱上。

35

使用案例的命名

- 使用案例的命名應該從使用者的觀點來描述。
- 如果事件描述不是從使用者的觀點，那麼不應該直接拷貝事件的名稱做為使用案例名稱。
- 並不是任何的動作都是使用案例！系統所執行的處理中有很多是使用者看不到的。

36

使用案例的命名

- 一個使用案例都有一個唯一的名稱，並且，使用案例的名稱均以動詞為開頭。這是從英文的語法結構來講。如果是用中文來描述使用案例，使用案例是代表系統能夠執行的功能，它代表的是動作、處理，因為使用案例描述系統的行為。

37

UML

- 在UML中，以一個橢圓形來表示使用案例。使用案例用來表示系統應提供的功能。
- 下圖顯示一個顧客(customer)以及一個使用案例 - 瀏覽目錄(Browse Catalog)。

38

使用案例找法

- 最基本的方式就是從事件表開始找起。
- 有時候，事件的描述太細了，可以將一系列相關的事件有組織地集合起來使之成為一個使用案例。
- 利用事件表，應該可以開始找出許多的使用案例。

39

使用案例找法

- 可能也要考慮到，有其他的系統需要跟本系統互動？
- 假設說在公司中的郵件系統是一個獨立的資訊系統，而所要建立的系統需要寄送郵件來通知顧客某些事情，那麼這兩個系統就會有互動。

40

描述的範圍

- 使用案例所描述的功能範圍可大可小。至於有沒有範圍的限制則沒有一定的規則。
- 儘量用actor可以看到的系統功能來進行塑模。所討論的actor必須是人，也就是使用者。

41

描述的例子

- “處理訂單”這個使用案例如果包含有“建立新訂單”，“提交訂單”，“更改訂單”，“處理訂單”這個使用案例就描述的太大了。
- “更改訂單的郵遞區號“這個使用案例就描述的太細了。

42

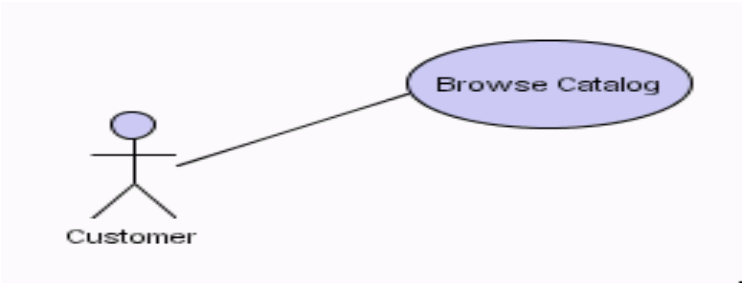
利用CRUD來捕捉Use Case

- CRUD指的就是建立(Create)、存取(Retrieve)、更新(Update)、刪除(Delete)這四個動作。這是從資料庫的觀點來看。一個資訊系統的主要功能就是在處理資料。而處理資料牽涉到的就是這四個基本動作。當系統執行這些CRUD的動作時，都會在結束後，給使用者一些有意義的回應，例如成功或是失敗的訊息。

43

連結線

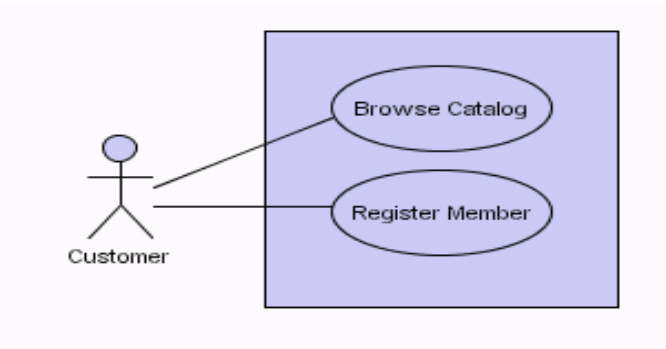
- 連結線是一條直線，它用來連結使用者及系統執行的功能。這條連結線表示某個角色的演員啟動了某個案例。
- 下圖顯示使用者瀏覽產品目錄。



44

系統

- 系統以一個方形為代表。用以表示系統的邊界。當開發的系統並不與其他系統有互動時，有時候會省略它。



45

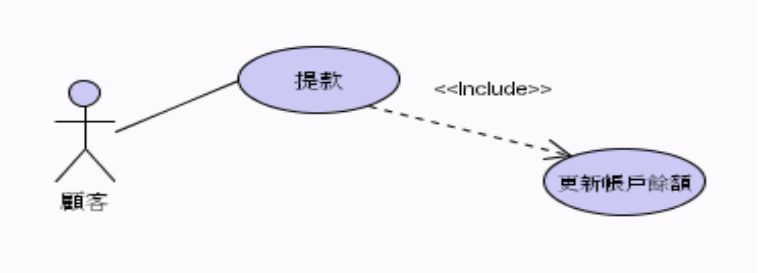
關係(Relationship)

- 在使用案例圖中，除了包含有演員，系統，使用案例，連結線之外，使用案例與使用案例之間也可以有關係的存在。
- 使用案例的關係中最常用到兩種：
  - 包含關係(include)
  - 延伸關係(extend)。

46

包含關係(include)

- <<include>>：包含關係有點像程式語言中的副程式。對於一個ATM的系統，”提款”為一個典型的使用案例。”提款”這個使用案例”一定”會包含有”更新餘額”這個使用案例。那麼可以將它們繪製如下：



47

包含關係(include)

- 如果一個使用案例A在其執行的過程中一定會使用到使用案例B，則案例A包含案例B。這個關係強調的是百分之一百的肯定。

48

包含關係的繪製方法

- 包含關係的表法為畫一條帶有箭頭的虛線，從案例A(包含者)連到案例B(被包含者)，並且在線上要標記出<<include>>。
- 在UML中，帶有箭頭的虛線表示相依的關係。所以上圖可以說成：”提款”使用案例依賴於”更新帳戶餘額”使用案例。因為A依賴B，所以A沒有B不行。所以，A一定會用到B。

49

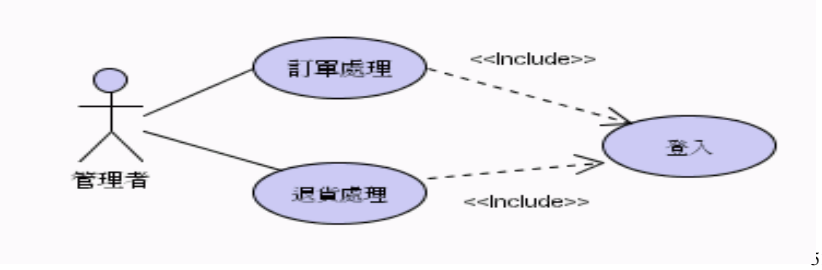
包含關係的好處

- 利用包含的關係就不需要在不同的地方重複描述同一個使用案例。
- 當發現許多個使用案例都共用一些相同的功能時，可以把此功能獨立出來，讓它自成一個案例，其他的使用案例只需要包含它就可以。

50

例子：登入

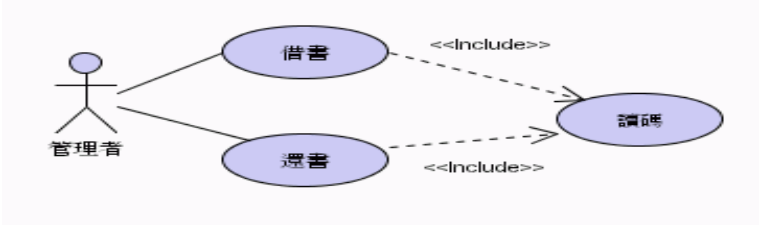
- 在一個購物系統中，管理者可以執行"訂單處理"以及"退貨處理"這兩個使用案例。在執行這兩個使用案例前，管理者必須要登入到系統並且通過查核才可以執行它們。面對這種需求描述，可以把它繪製如下。



51

例子：讀碼

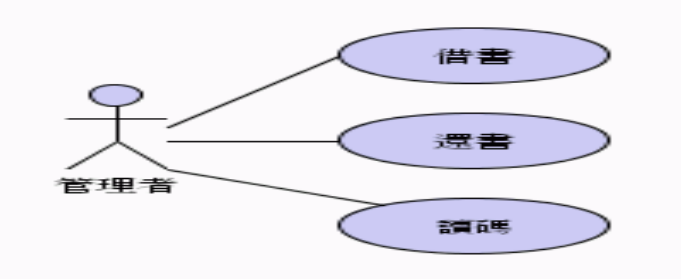
- 曾經到圖書館借過書、也一定還過書。是否有注意到不論是借書或是還書，圖書館的管理員都會掃描書上的條碼(假設圖書館採用讀碼機)？如果要為某個圖書館建置管理系統並且該圖書館也將採用讀碼機的話，使用案例圖中應該會有：



52

關係繪製的建議

- 先將所有的使用案例都劃上。
- 再討論案例與案例的關係。
- 在得出上面的使用案例圖之前，初步使用案例圖可能如下，也就是每個案例都是獨立的。



53

Stereotype

- include這個字的前後有加上<< >>
- 在UML中，它稱為stereotype。主要的作用是用來擴充UML的詞彙(vocabulary)，提供不修改UML而將其延伸的功能。

54

Stereotype

- UML適用的領域很廣。針對不同的領域及實際情況，使用者可以自行定義詞彙，進而擴充UML的功能。
- 定義詞彙的方式就是在字的兩旁加上<<詞彙>>。所以，stereotype是UML所提供的一種擴充語意的機制。UML本身有針對一些情形定義一些stereotype。<<include>>以及<<extend>>這兩個stereotype會出現在使用案例圖中。

55

Stereotype

- UML還提供了標記值 (tagged value)以及限制(constrain)做為其擴充機制。

56

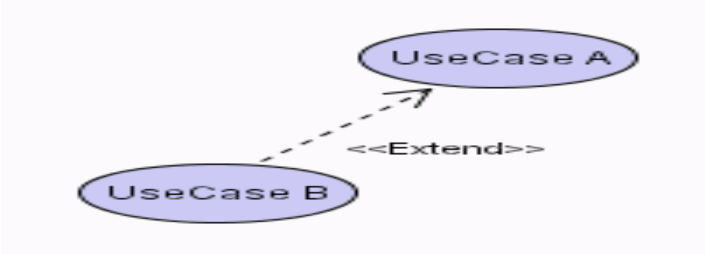
延伸關係(extend)

- <<extend>>：延伸關係。在某些情況或是條件下，一個使用案例的行為可以被另一個使用案例的行為所延伸。”某些情況或是條件”稱為延伸點(extension point)。
- 延伸的關係不太好理解。初學UML的人很容易把延伸跟包含混淆。如果用”安插”這兩個字來取代延伸，可能會比較容易去了解意義。

57

簡單的範例

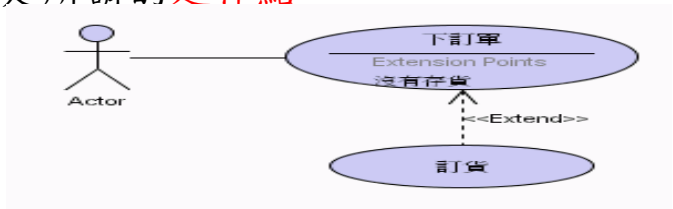
- 假設說有兩個使用案例 – 使用案例A以及使用案例B。使用案例B延伸使用案例A。
- 此延伸關係的表示方法為從使用案例B畫一條帶有箭頭的虛線指向使用案例A。也就是使用案例B安插到使用案例A。



58

延伸點

- 假設在需求描述中有：使用者可以下訂單。系統會去檢查庫存有無貨品。若是沒有存貨，系統要執行訂貨。從這個描述裡頭中，看出了兩個使用案例，一個是"下訂單"，另一個是"訂貨"。而敘述中的"檢查庫存有無貨品，若是..."這一段話就是條件，它就是所謂的延伸點。



59

延伸點

- "訂貨"使用案例可以被安插在"下訂單"使用案例中。
- "訂貨"使用案例延伸了"下訂單"使用案例的功能。
- "訂貨"使用案例可以單獨存在。"訂貨"使用案例並不是一定會被叫用。只有當某些條件滿足時，才會被叫用。

60

注意事項

- <<extend>>跟程式語言中的extends完全沒有關係。如果用程式語言的extends來想使用案例的extend，那就大錯特錯了。

61

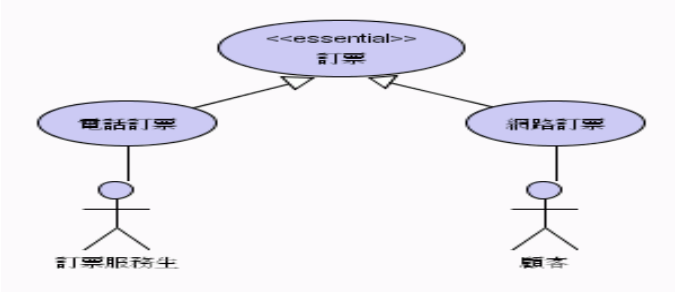
一般化關係(generalization)

- 使用案例圖中也可以表示一般化關係，使用案例的一般化關係與物件導向語言的一般化關係是類似的概念。

62

例子：一個訂票系統

- ”訂票”是一個很容易捕捉的使用案例。如果要細分訂票方式的種類，那麼訂票的方式可能是打電話來訂票，或者是直接在網路上訂票再來取票。那麼可以利用下列的方式來表達。



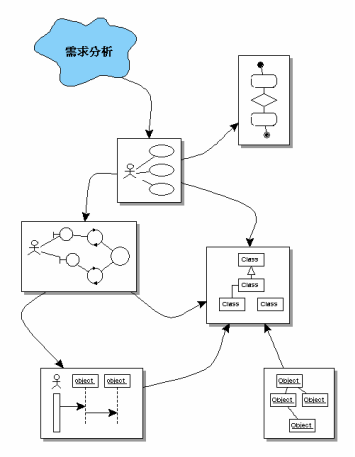
63

<< essential >>

- 這個使用案例圖說明了”電話訂票”以及”網路訂票”在本質(essential)上都是訂票。所以可以使用一般化來塑模這些案例們。並且在父使用案例加上<<essential>>這個stereotype來表明這個一般化的關係。

64

情節



65

使用案例

- 使用案例圖的繪製其實很簡單。它只牽涉到幾個非常基本的UML圖形，直覺上好像是畫幾個人型圖案、幾個橢圓，然後用直線把它們連一連，使用案例就完成了。當然，對於系統的高階動態描述絕不是繪製完使用案例圖就結束了。

66

Use Case 是一個開端

- 系統開發的過程中，使用案例圖只是一個開端。使用案例圖將會驅動整個系統開發的過程。而這也是RUP所強調的精神之一。
  那麼，接下來的工作要看的是使用案例描述及情節(scenario)。

67

使用案例描述

- 在Booch等所著的UML使用手冊這本書中，給出了下面的使用案例定義：

"…a description of set of sequences of actions, including variants, that a system performs that yield an observable result of value to an actor."

- 翻譯成中文，它的意思是：”一個系統執行並且產生可觀察且對使用者有價值的結果之一序列動作的描述”。

68

使用案例是動作的描述

- 使用案例是動作的描述，它還是一序列動作的描述。使用案例描述系統行為的執行步驟，並且包含過程中的可能發生狀況。使用案例的執行應該產生對於使用者有價值的結果。
- 使用案例不是只有一個橢圓形就沒了！它只表示了故事的開端。

69

系統行為的描述

- 對於系統行為的描述可以用一般敘述性的文字來表達。在描述的內容上，一方面採用使用者的觀點，也就是使用者從外面看得到的系統行為，另一方面則是描述系統的回應。對於描述，可能會有一個疑問，那就是描述的範圍是要到哪個程度？這個沒有一定的答案。不要在描述中談論設計細節或是特定的實作方法。

70

描述行為的例子

- 案例描述中可能會出現"系統顯示產品目錄給使用者"。
- 並沒有提到如何顯示、顯示在哪裡、顯示的格式、儲存產品目錄的資料庫是哪種資料庫等細節。產品目錄一定是存放在某個資料庫。上面那句話細緻化成"系統首先取出產品目錄。然後，系統將產品目錄顯示給使用者"。如何取出、從哪裡取出等等細節都不是目前關心的重點。只知道要去取出來，才有東西可以顯示給使用者看。

71

描述使用案例

描述使用案例時，敘述內容應該紀錄以下幾大項目：

- 使用案例如何開始。
- 使用案例如何結束。
- 使用者如何與系統互動。
- 互動的過程有什麼樣的訊息交換。
- 互動行為的正常過程，以及其他的過程。

72

描述的格式

- 使用案例描述的格式並沒有一定的規格。採用表格的方式。此表格分為左右兩個欄位。左欄記載著使用者(actor)的請求動作，右欄記載著系統的回應。

Actor 動作	系統回應

73

”使用者訂購音樂CD”

- ”使用者訂購音樂CD”這個使用案例來了解如何進行使用案例描述。假設顧客在之前已經利用系統所提供的搜尋功能找到了所想要購買的音樂CD摘要。系統顯示音樂CD摘要的網頁給顧客。接下來，可能會發生的情節把它寫成如下：

74

Actor 動作	系統回應
1. TUCBW: 顧客選擇其中的一個CD以查看更多的資訊	2. 系統提供顧客CD的詳細資訊
3. 顧客把CD加入到購物車	4. 系統顯示購物車內容
5. 顧客登入系統結帳	6. 系統驗證顧客的信用卡資料
	7. 系統儲存訂單交易資訊
	8. 系統寄發訂單確認函(email)給顧客
	9. TUCEW : 系統顯示訂單內容以及交易明細

75

描述的說明

- 用數字來表示執行的步驟。可以發覺到兩個奇怪的英文出現在第1步驟以及第9步驟。
- 步驟1的TUCBW是The Use Case Begin With的縮寫。它的意思是”這個使用案例開始於”。
- TUCEW則是The Use Case End With的縮寫，它的意思是”這個使用案例結束於”。

76

情節(scenario)

- 討論使用案例時，一定會提到情節。上面的描述就是一個情節的例子。
- 在使用案例中，所謂的情節是指使用案例的某單一執行路徑。
- 為什麼使用案例會有不同的情節？因為使用案例並不是只有單一的執行路徑。

77

上面的例子

- ”使用者訂購音樂CD”這個使用案例描述中的步驟。尤其是第6步驟： 系統驗證顧客的信用卡資料。

78

上面的例子

- 驗證信用卡時可能有兩種情況會產生：成功與失敗。成功的時候，使用案例會繼續第7個步驟。失敗時上面的表格中並沒有寫出。所以，使用案例是會有不同的執行路徑。上面的表格所描述的路徑稱為正常路徑。
- 對於使用案例的可能不同路徑可以用另一個表格來表示。有時候也可以只是文字敘述。

79

例外路徑：信用卡驗證失敗

Actor 動作	系統回應
1. TUCBW: 顧客選擇其中的一個CD以查看更多的資訊	2. 系統提供顧客CD的詳細資訊
3. 顧客把CD加入到購物車	4. 系統顯示購物車內容
5. 顧客登入系統結帳	6. 系統驗證顧客的信用卡資料
	7. TUCEW :驗證失敗，系統顯示錯誤訊息。

80

例外路徑：信用卡驗證失敗

例外路徑：信用卡驗證失敗

步驟6：系統驗證顧客信用卡資料，驗證失敗。

步驟7：系統顯示錯誤訊息。

81

購物系統例子

- 如果需求分析中對於"下訂單"使用案例的描述為：使用者藉由網路下訂單。系統接到了訂單，會去檢查庫存有無貨品。若是沒有存貨，系統要執行"訂貨"。那麼使用案例描述會是：

82

例外路徑：缺貨(用表格式)

Actor 動作	系統回應
…	…
5. 顧客登入系統結帳	6. 系統驗證顧客的信用卡資料
	7. 系統檢查貨品庫存量
	8. 系統執行"補訂貨"使用案例
	9. TUCEW：系統顯示缺貨訊息

83

使用案例文件

對於每一個使用案例應該有一份相對應之使用案例說明文件。從使用案例描述所建議的事項，使用案例文件可以包含有下列項目

- 使用案例名稱。
- 案例簡述。
- 參與角色。
- 前提。
- 成功條件。
- 主要路徑。
- 其他或是例外路徑。
- 詞彙表。
- 其他相關資料。

84