學會 **UML/OOAD**

這樣開始就對了

作者簡介

邱郁惠(271080@gmail.com)

畢業於東吳大學資訊科學系，研究 UML、OOA 十餘年，2007 年創辦 UML Blog(http://www.umltw.com)推廣 UML 技術，並專職於企業內訓、專案輔導、自辦課程、專欄寫作。

擔任過 NEC、華夏、中科院、百通、MISOO 物件教室、大竑、中華汽車、HSDc(2007)、資策會(2008)、台灣大哥大(2008)、意藍科技(2008)、新鼎(2008)、博客來(2009)、網飛訊(2010)、文化大學推廣部(2010)、北區職訓局(2010)、PMI-TW 國際專案管理學會(2010)、巨鷗(2010)、三商電腦(2010)、秀傳醫療(2011)、翔生資訊(2011)、國際厚生(2011)、中華電信(2012)等公司的內訓講師及輔導顧問，也擔任過物件導向雜誌主編暨 UML/OOAD 專欄作家、RUN！PC 旗標資訊月刊(2008~2009)、以及 iThome 電腦報(2008~2012)專欄作家。

出版的繁體書有《寫給 SA 的 UML/MDA 實務手冊》(天瓏銷售第 1 名,已絕版)、《寫給 C++程式設計師的 UML 實務手冊》(天瓏銷售第 4 名,已絕版)、《UML-SystemC 晶片設計實務》(已絕版)、《OCUP/UML 初級認證攻略》(天瓏銷售第 14 名,已絕版)、《寫給 SA 的 UML/UseCase 實務手冊》(天瓏銷售第 10 名,已絕版)、《學會 UML/OOAD 這樣開始就對了》(金石堂預購第 1 名,已絕版)、《Visual Studio 2010/UML 黃金準則》、《SA 前進 UML 專案現場》、《IT 人，你如何表達？》(未出紙本版)。

同時，出版的簡體書有《系統分析員 UML 實務手冊》、《C++程式師 UML 實務手冊》、《SoC 設計實務手冊》、《UML 那些事兒》、《系統分析師 UML 用例實戰》、《UML 和 OOAD 快速入門》、《Visual Studio 2010 和 UML 黃金法則》、《系統分析師 UML 項目實戰》。

目前擁有 OCUP(OMG-Certified UML Professional)三級認證、PMP(Project Management Professional)認證、ITIL V3 Foundation 認證、IBM OOAD(Object Oriented Analysis and Design)認證、Scrum Master 認證，曾榮獲大陸頒予《優秀 IT 技術圖書原創作者》獎。

目錄

0

前言

※※

0.1 本書使用的技術

E 世代講求快速、輕薄，在系統開發上頭，也是如此。可是 UML2 有十四張圖，並不符合輕薄，通通用起來，也不快速。當然，UML 希望可以用在各種系統開發中，所以有理由厚重。但是，我們講求快速入門，因此只選用其中必用的三款圖：類別圖(class diagram)、用例圖(use case diagram)和循序圖(sequence diagram)。

不過，UML 這三款圖不太夠，所以我們搭配了其他技術，如下：

1. 交易樣式(transaction patterns)—套用交易樣式，快速繪製出類別圖。
2. 用例敘述(use case description)—針對用例圖中的每個用例，以文字方式描述用例的執行流程。
3. BCE 樣式(Boundary-Control-Entity patterns)—套用 BCE 樣式，協助繪製出循序圖。

此外，UML 本身只是個單純的圖形語言，並不包含分析設計步驟，所以本書提出了一套無接縫的分析設計步驟。首先，由分析師交付一套分析階段的類別圖、用例圖文和循序圖的文件給設計師；接著，設計師才依據這套分析文件，添加跟實作技術有關的設計內容，產出另一套類別圖、用例圖文和循序圖的設計文件給程序員。

0.2 民宿聯合訂房系統

本書以「民宿聯合訂房系統」爲主要範例，在講述任何概念時，如果沒有特別說明的話，都以這個範例爲主。聯合訂房系統的服務非常明確，

會員可以上網向多家民宿訂房。加入會員的房客會是這個系統的主要使用者。爲了簡化這個範例，所以我們剔除了後台的管理機制，也就是說，這個系統並沒有包含後台的管理功能。

0.3 如何閱讀本書

本書內文共分六章，前三章講分析，後三章講設計。如果，您是分析師，爲求快速、省時，可以不讀後三章的設計章節。當然，在時間允許之的情況下，還是會建議分析師閱讀後三章的設計章節，這樣會更懂得如何跟設計師溝通。

但是，您要是設計師的話，一定得閱讀前三章的分析章節，因爲關於 UML、交易樣式和 BCE 樣式的概念，分析師也需要學習，所以會在分析章節先講述，您要是跳過前三章的話，可能會有點不明就裡。

1
(A1)類別圖

※※

1.1 類別圖

類別圖(class diagram)用來表達系統內部的靜態結構(static structure)。具體來說，開發人員可以透過類別圖的設計，來將數以萬行的程式碼分門別類，構成了系統內部的靜態結構。

過去，開發人員在寫程式時，需要切模組(module)、定功能(function)、宣告變數，這些動作在物件導向(Object-Oriented)技術中，一樣都沒少。

但是，觀念上有兩個顯著的改變：

1. 新術語—模組變類別(class)、功能變操作(operation)、變數變屬性(attribute)。新術語並不是舊酒換新瓶，而是在切類別、定操作、宣告屬性上頭，有新的切法。
2. 新切法—以前的做法是以功能的角度，把大功能、大流程切成數個模組；再把功能模組切成小功能、小流程，定出功能；然後在編寫功能時，宣告所需的變數。新的切法是，拿使用者的領域術語當類別，然後定出相關的操作和屬性，封裝在同一個類別中。

所以，再回過頭來看，系統的內部結構是由一個個的類別所組成，類別內部有操作和屬性，類別和類別之間有靜態關係(static relationship)。由於，類別裡頭同時包含了靜態資料(屬性)，資料之間會有關聯的需要，這種以資料爲主的關聯，即爲「靜態關係」。也就是說，類別圖不僅規範了程式碼，其實還同時規範了資料庫的資料結構。

在 UML 中，類別圖的元素相當繁雜，分析師當然不需要全學，所以在接下來的小節中，我們僅談論分析師必學的元素。

1.2 分析師必學元素

1.2.1 類別

想像一下，系統數以萬行的程式碼被分爲一個個的區塊，每一個區塊即爲一個類別。每一個類別中，包含有操作和屬性；操作內部放置邏輯運算相關的程式碼，屬性則爲所需的區域變數。

分析師不能自己隨意定義類別，必須尋找領域術語做爲類別名稱。比方說，一談到訂房，我們就會想到打算預訂什麼樣的房間，這裡會找到兩個領域概念：

1. 房間—真正住進去，特定房號的房間。
2. 房型—顧客在訂房時，通常是預定某個房型的房間。

類別的圖示爲矩形，通常矩形內部會分爲三格：頂格放置類別名稱、中格放置屬性、底格放置操作，如圖 1-1 所示。說明如下：

- 屬性—屬性是資料項目，所以需要指定它的型別(type)，屬性名稱後接冒號隔開型別。如果，屬性有預設值的話，也可以在型別後頭加上等號，然後標出預設值。
- 操作—操作名稱後頭以小括號帶輸入參數，然後可以在小括號後接冒號，標出回傳參數的型別。
- 能見度—特別注意到，屬性名稱前頭有個減號、操作名稱前頭也

有個加號，這個符號稱爲「能見度」(visibility)。由於，屬性和操作被封裝在類別中，所以需要使用能見度的來標示它們的存取等級。UML2 設置了四種能見度，分析師只要學「私有」(private)和「公開」(public)兩種，就可以了。

- 私有—屬性的能見度通常會設成私有等級，除了所屬類別內部的操作可以直接存取之外，其他類別內部的操作不可以存取私有等級的屬性。
- 公開—操作的能見度通常爲設爲公開等級，不僅所屬類別內部的其他操作可以調用(call)，其他類別內部的操作也可以調用公開等級的操作。

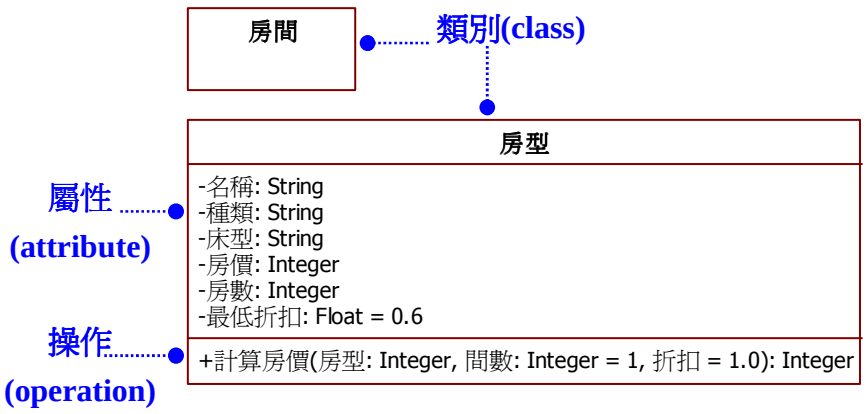

圖 1-1: 類別

1.2.2 結合關係

很多分析師都有設計「實體關聯圖」(Entity-Relationship Diagram,ERD)

的經驗，我們會在資料表(table)之間建立關係，這樣才能關聯不同資料表內的資料。

比較一下，資料表跟類別最大的差別在於，資料表只包含資料，但是類別同時包含了資料(屬性)和操作。也就是說，類別其實具備的資料表的靜態結構特性，但是又多了一份動態行爲特性。至於，原先在資料表之前的關聯，對應到類別之間，則稱爲「結合關係」(association)。

分析師可以善用以前實體關聯圖的概念，來認識類別圖，簡單對照如表 1-1 所示。雖然，紀錄的概念可以對應到物件，但只能對應到半個物件，因爲一個物件同時含有屬性值和操作，但是一筆紀錄只含有欄位值。

實體關聯圖(關聯式資料庫)	**類別圖(物件導向)**
資料表(table)	類別(class)
紀錄(record)	物件(object)
欄位(field)	屬性(attribute)
無	操作(operation)
關聯(relationship)	結合(association)

表 1-1: 實體關聯圖與類別圖

至於，資料表之間的關聯有分爲一對一關聯、一對多關聯、多對多關聯，用來表示一筆紀錄能夠關聯到另一個資料表中的幾筆紀錄。類別之間的結合關係中，同樣也有這樣的概念，稱之爲「個體數目」(multiplicity)，用來表示一個物件能夠結合到另一個類別中的幾個物件。

結合關係的圖示是實線，兩個結合端點可以標示個體數目。個體數目的下限爲 0，上限是無限大(*)，下限標示在前面，上限標示在後面，兩數

字之間使用兩個點點(..)隔開。請看圖 1-2 的例子，表達了一個房型可能連結一到多個房間，而一個房間則一定被限定連結到某一個房型。

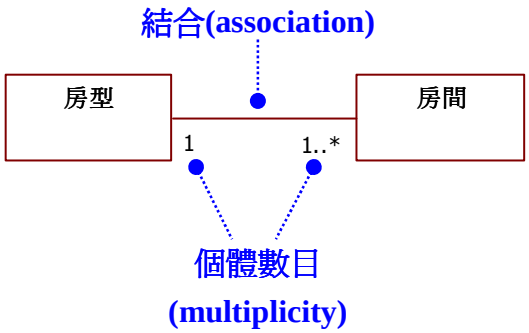

圖 1-2: 結合關係

在多對多的處理上，如同關聯式資料庫的經驗，由於處理起來太複雜，所以通常會將多對多拆解成兩個一對多的結構。比方說，一次訂房可能預訂多種房型，而每一種房型也可能跟多個不同的訂房事件有關，兩者之間形成多對多的結合關係，如圖 1-3 所示。

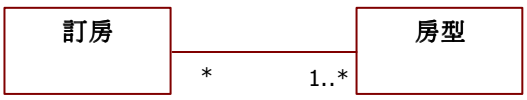

圖 1-3: 多對多的個體數目

因爲，多對多不好處理，所以我們將圖 1-3 拆解成兩個一對多的結合關係，如圖 1-4 所示。一次訂房包含多個訂房明細，每一個訂房明細則隸

屬於某一次的訂房交易中。再者，一個訂房明細會連結到一個房型，每一個房型可能出現在多個訂房明細中。

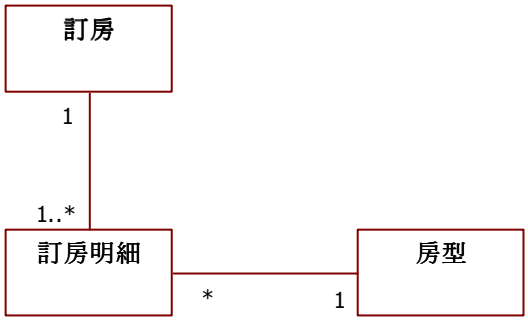

圖 1-4: 兩個一對多的個體數目

1.2.3 組合關係

再以圖 1-4 爲例，仔細比較訂房-訂房明細，以及訂房明細-房型，兩者之間的關係，不難發現，訂房-訂房明細之間的關係比較特別。訂房-訂房明細之間的關係有一種「整體-部分」(whole-part)的強烈關係，一旦代表整體的訂房物件被刪除的話，底下所連結的訂房明細物件都應該一併被刪除才對。但是，訂房明細-房型之間則沒有這般強烈的擁有權。

如果，分析師要表達兩種物件之間有這般強烈的擁有權時，可以在整體端標示實心菱形，未標示則代表部分端，如圖 1-5 所示。具備強烈整體-部分的結合關係，特別稱爲「組合關係」(composition relationship)。不過，組合關係很容易遭到濫用。切記，整體-部分在領域概念上，兩概念必須是不可分割、或者難以分割的。

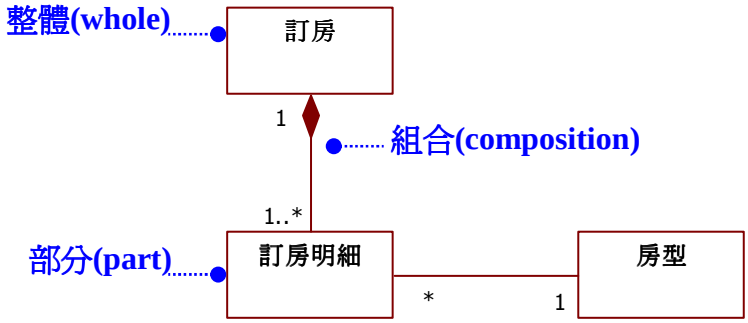

圖 1-5: 組合關係

1.3 交易樣式

領域概念何其多，建議分析師套用「交易樣式」(transaction patterns)迅速勾勒出類別圖的雛型。交易樣式由知名的 OOAD 大師 Peter Coad 所提出，您可以在網路上查到許多相關資料，主要出處爲《Object Models: Strategies, Patterns, & Applications》一書。

不過，我們在此處的引用上，多了一些限制，所以使用上會比 Peter Coad 所提出的交易樣式更狹隘一些，但是並沒有改變 Peter Coad 的原意。我在《寫給 SA 的 UML/MDA 實務手冊》和《寫給 SA 的 UML/UseCase 實務手冊》這兩本書中，特別是後者，都有詳細解釋過交易樣式，有興趣的話，也很推薦您找來讀一讀。

1.3.1 交易與人地物

顧名思義，交易樣式強調以「交易」(Transaction)爲中心，串起跟交易

相關的交易明細(TransactionLineItem)、關係人(Participant)、地點(Place)、物品(Item)，如圖 1-6 所示。

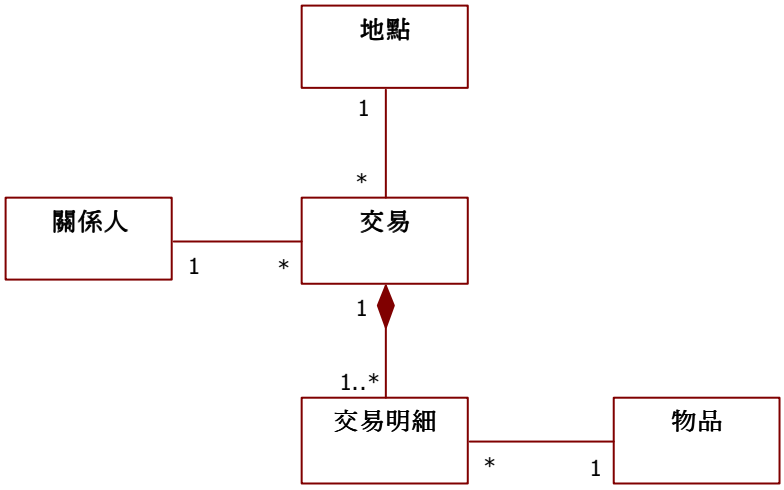

圖 1-6: 交易樣式(核心)

不過，此處對「交易」的定義比較廣義，並不侷限於一般的商業交易，舉凡系統「必須記錄的事件」都是我們的候選交易。因此，交易可能是一個短暫的時間片刻，也可能是一小段時間。

所以，套了圖 1-6 的交易樣式之後，分析師大概可以很快得出如圖 1-7 所示的類別圖雛型了吧！

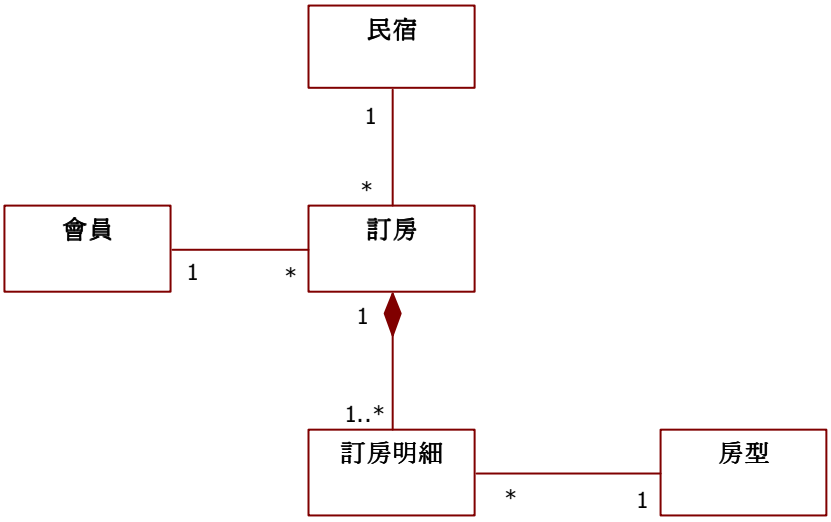

圖 1-7: 套用交易樣式

1.3.2 物品與特定物品

交易涉及的物品概念，可以細分成兩種：一種是具體的、特定的物品，另一種是針對一群同種類特定物品的描述或分類。在交易樣式中，則將這兩個概念分爲「特定物品」(SpecificItem)和「物品」(Item)，如圖 1-8 所示。

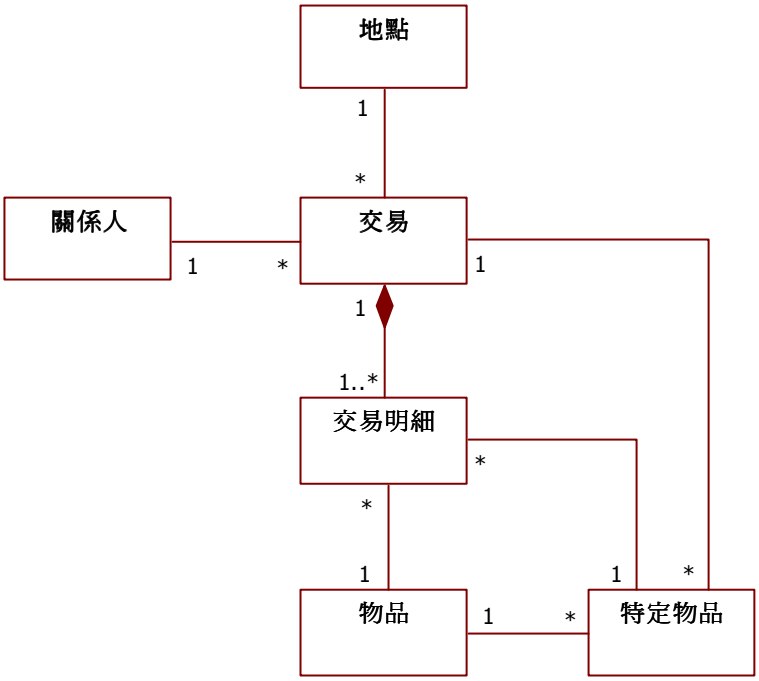

圖 1-8: 物品與特定物品

其實，我們在前面提到的「房型-房間」的概念，就是「物品-特定物品」的應用，如圖 1-9 所示。不過，在訂房的範例中，我們目前還無法確定是否可以直接套用「特定物品-交易明細」以及「特定物品-交易」。分析師要是遇到這種未確定的情況，可以標上像便利貼一樣的「註解」(comment)來提醒自己留意。

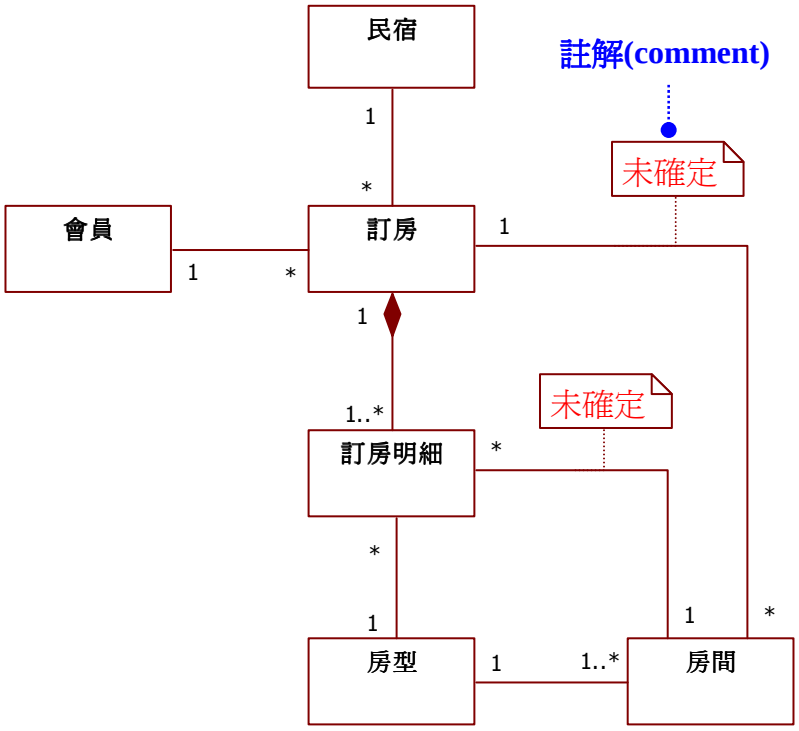

圖 1-9: 套用「物品-特定物品」

之所以無法確定圖 1-9 中，「房間-訂房明細」以及「房間-訂房」這兩條結合關係，最主要原因在於，這兩條關係都不是單純的一對多結合，而是如圖 1-10 所示的多對多結合。此處，我們先保留多對多結合，稍後再來討論細節。

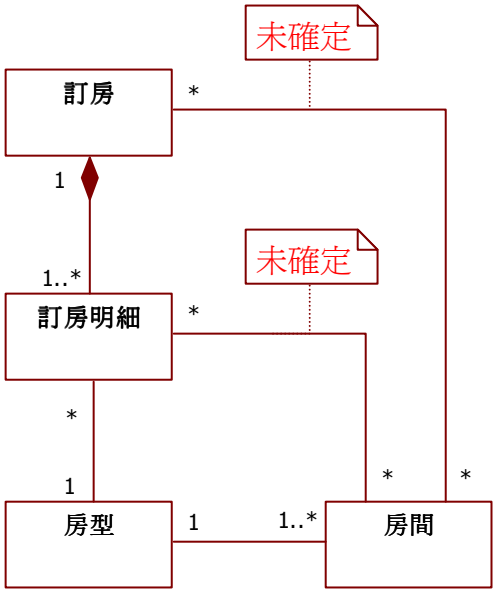

圖 1-10: 多對多的複雜狀況

1.3.3 後續交易

交易本身含有時間因子，所以如果拉出一條時間軸的話，會看到交易之後可能有「後續交易」(subsequentTransaction)，如圖 1-11 所示。

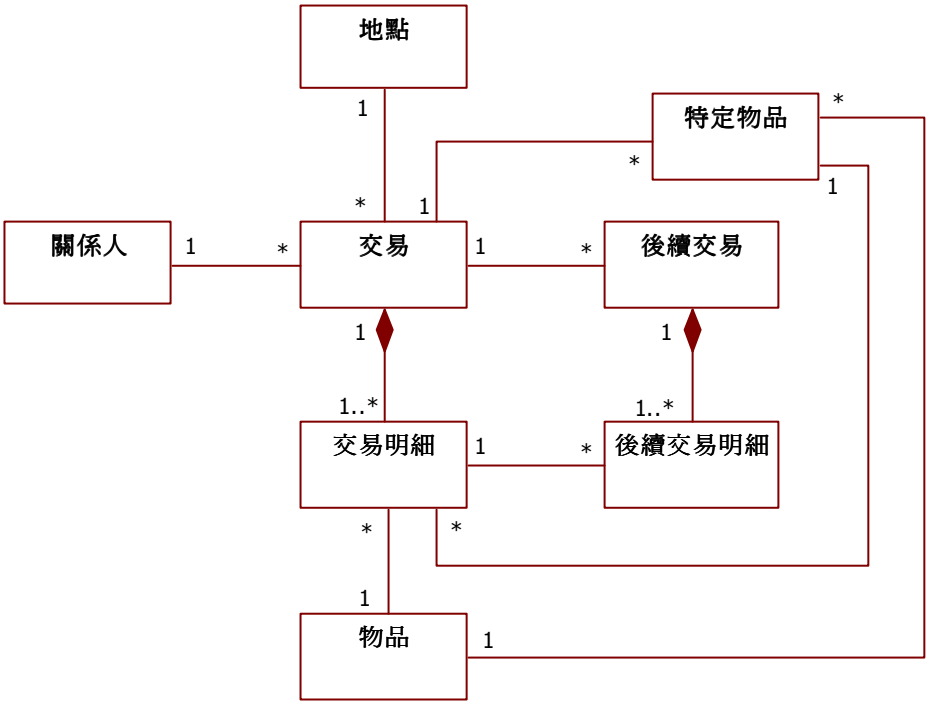

圖 1-11: 後續交易-後續交易明細

想想看，訂房成功之後，後續會發生什麼必須記錄的重要事件？我第一個想到的後續交易是「入住」(check in)，如圖 1-12 所示。

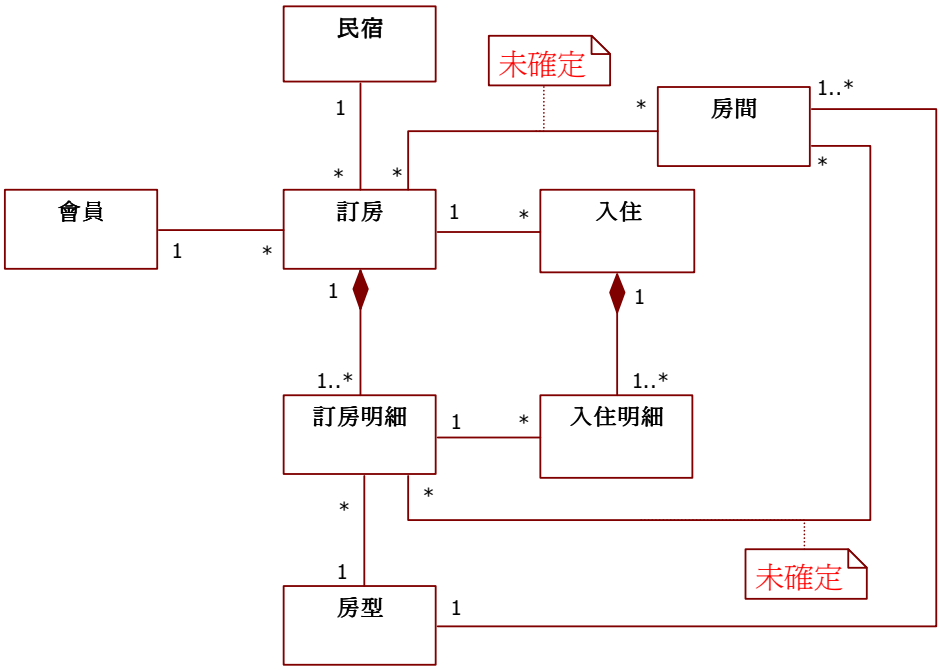

圖 1-12: 套用「後續交易-後續交易明細」

套用交易樣式不僅可以幫分析師快速產出類別圖雛型，更重要的是，可以協助分析師使用比較有系統性的方式去理解領域概念。比方說，套用交易樣式到目前爲止，好像對於「訂房-訂房明細」以及「入住-入住明細」領域概念一直沒有真正釐清，分析師可能開始覺得有必要釐清這兩個概念。

首先，我們先釐清訂房-訂房明細，如圖 1-13 所示，我們做下述規定，讓它看起來簡單一點：一次訂房可以同時訂多個房間，但是只能限定是同一個預訂日期。假設，這個會員要一次預訂多天的話，必須拆成多筆訂房交易。

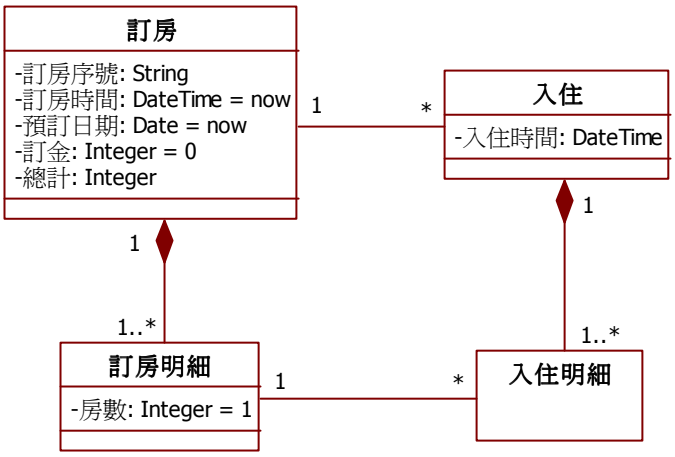

圖 1-13: 訂房與入住

再者，我們還規定一次入住事件對應一間房間。假設，會員預訂 12/20 兩個房間的話，那這次訂房就會對應到兩個入住事件。在這樣的規定之下，也就不再需要入住明細了，因此修改成圖 1-14 的樣子。

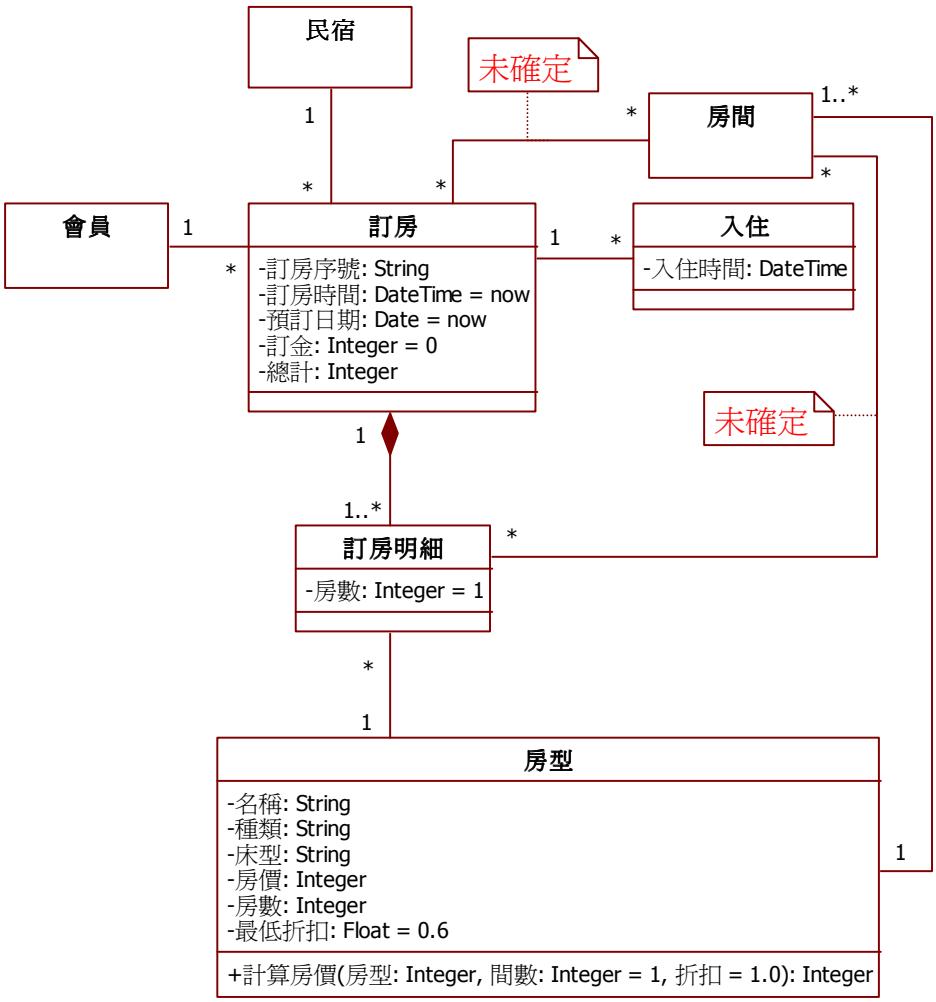

圖 1-14: 刪掉「入住明細」

1.3.4 行動者與關係人

最後，我們回過頭來看交易樣式中的關係人的概念。仔細探究起來，

關係人其實是一種身分、一種角色，在這個角色背後有一個真正的「行動者」(actor)。「行動者-關係人」之間的關係，就像是「演員-角色」，如圖 1-15 所示。

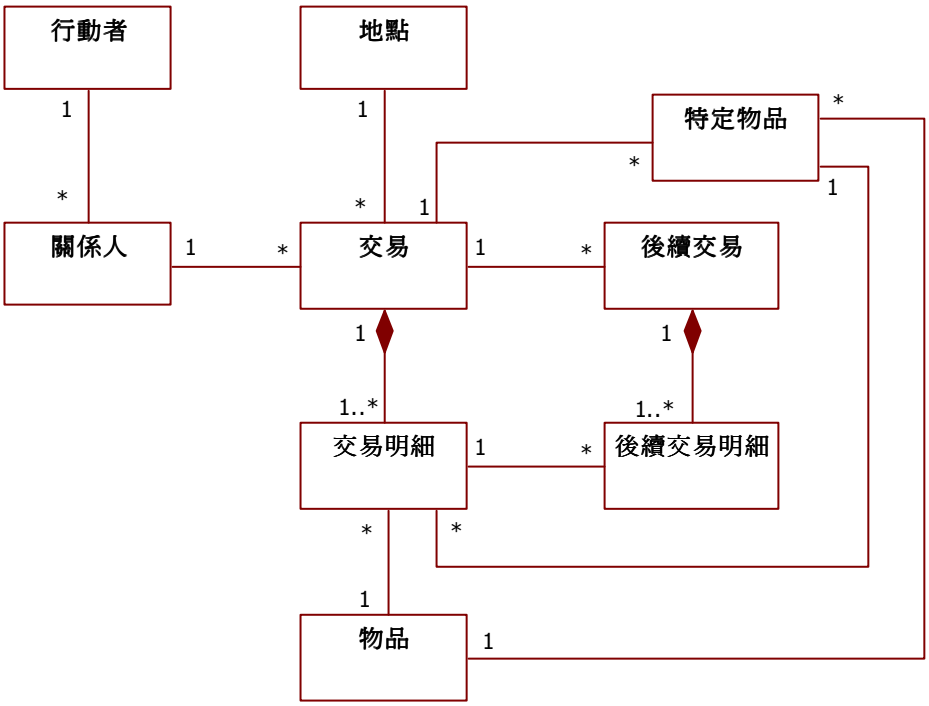

圖 1-15: 行動者-關係人

在訂房系統中，必須具備會員身份者才能夠訂房，所以「個人-會員」，就套用了「行動者-關係人」，如圖 1-16 所示。不過，我們還不確定一個人是否可以申請多個會員角色，所以標上了未確定的註解。

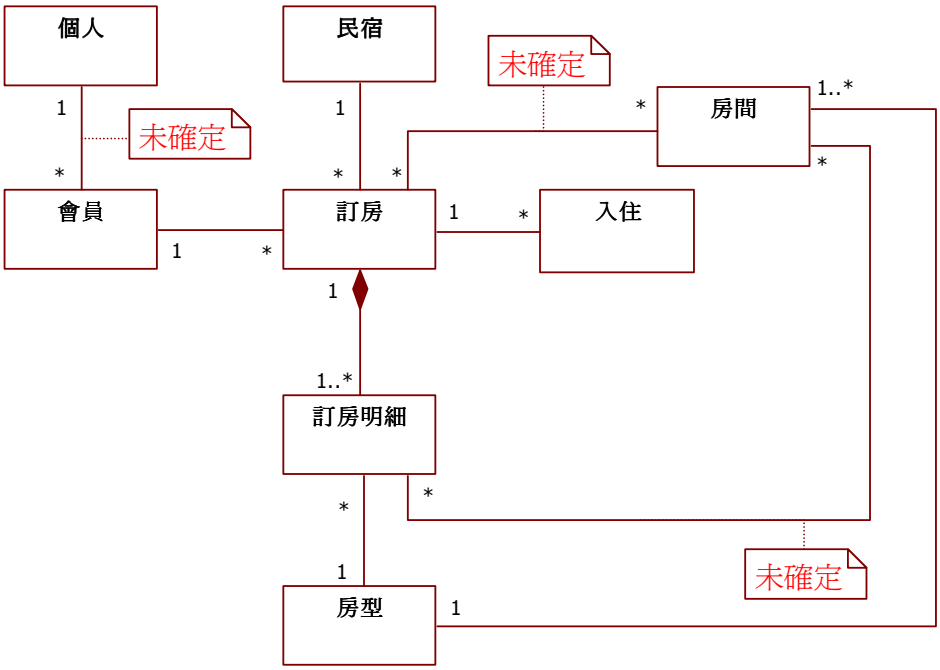

圖 1-16: 套用「行動者-關係人」

1.4 民宿聯合訂房系統

前面已經對類別圖討論出一點小雛型，到目前爲止獲得了，如圖 1-17 的類別圖雛型。

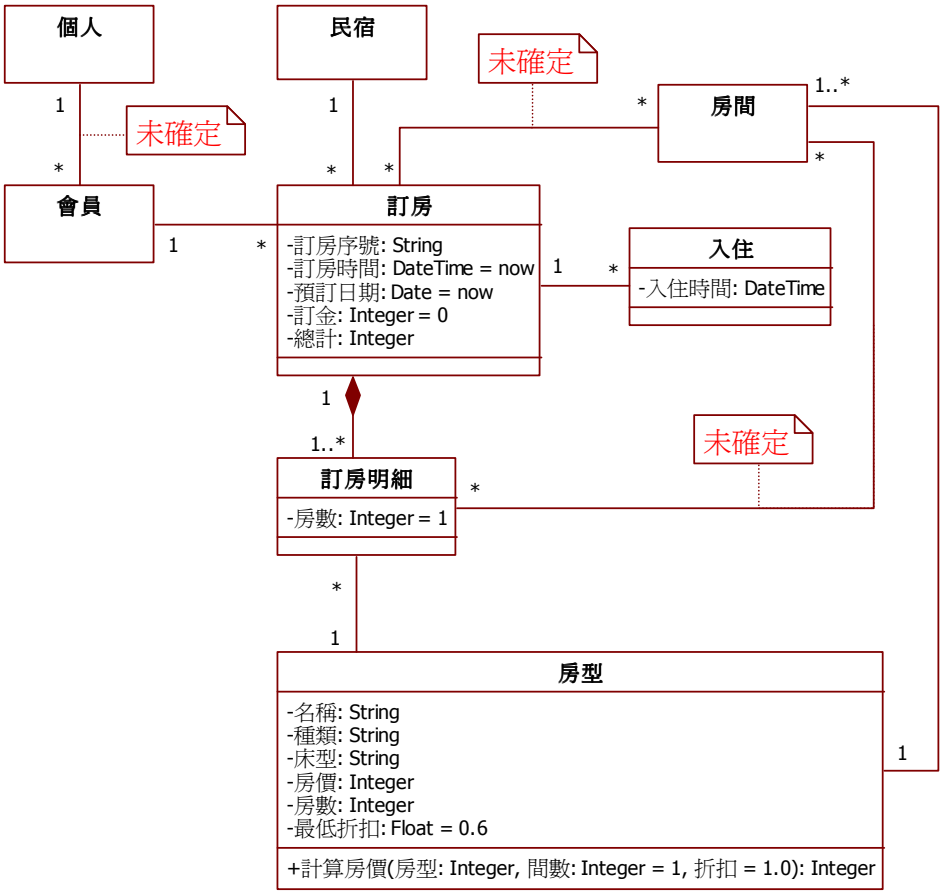

圖 1-17: 類別圖的雛型

原本在這個小節中，我們應該繼續來討論訂房系統的類別圖，不過我想先到此爲止，直接進入下一章來談用例圖文。分析師推進到用例圖文的時候，一定會對領域概念捕捉且理解更多，那時再把所獲得的回饋到類別圖中。

其實，別沉溺在任何一張圖中，單方向窮究一張圖是無法檢驗出分析

設計的好壞的。所以，我建議分析師應該迅速推進到下一張圖，就像齒輪組一樣，其中一個齒輪轉動，其實也會同時帶動其他齒輪的轉動，這樣才能夠檢驗出其他齒輪有沒有卡住，如圖 1-18 所示。

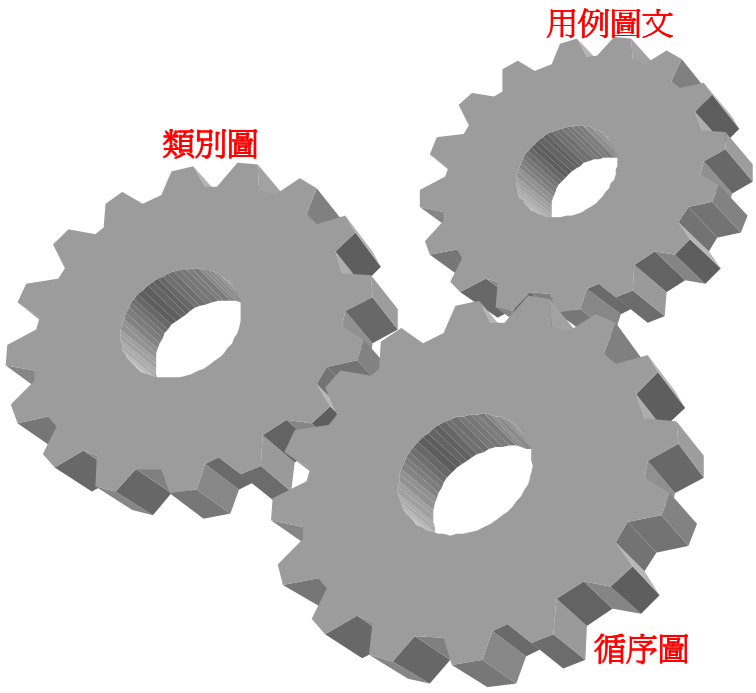

圖 1-18: 帶動彼此

2

(A2)用例圖文

※※

2.1 用例圖

在上一章的類別圖中，分析師學到了使用類別圖來表達系統內部的靜態結構；在本章的用例圖文中，分析師將學到透過用例圖文來表達系統對外提供的服務。

以類別圖的觀點，系統內部是由一個個的類別所組成；相較於用例圖的觀點，系統外部是由一個個的用例所組成。也就是說，類別是系統的裡子，用例是系統的面子，有了類別圖和用例圖之後，系統分析師就能夠兼顧系統的面子和裡子了。

知識要拿來用，才會成爲力量，所以我們一邊學習新的用例圖觀念，一邊要懂得應用並複習前面學到的類別圖觀念。因此，我們列出了表 2-1 來溫故而知新，對照一下類別圖與用例圖的觀念。

類別圖	用例圖
類別(class)	用例(use case)、參與者(actor)
結合(association)	包含(include)、擴充(extend)
系統的內觀(裡子)	系統的外觀(面子)
靜態結構	動態功能
穩定成長	變動迅速

表 2-1: 類別圖與用例圖

由於，類別圖是系統內部的結構，所以應該要能穩定成長。換言之，類別的結構應該要穩定，所以我們才會採用比較穩定的領域術語以及交易樣式；但是，穩定並非保守、僵化，我們希望穩定中透著彈性，這樣系統

內部結構才能夠在穩定中求成長。

至於，用例圖就不同了，因爲它表達系統對外提供給使用者的服務，就像樹上的杜鵑花一樣，能開多少就開多少，花開花謝，使用者有很多新功能可以用，開發人員有很多錢可以賺。

如同上一章的類別圖概念，我們對用例圖(use case diagram)的概念會分爲兩個部分來談：分析師必學的元素比較少，像是用例之間的包含關係(include relationship)、擴充關係(extend relationship)，我們就不談，也希望分析師少用，讓分析階段產出的用例圖越易懂，也才越能夠跟客戶溝通需求。

2.2 分析師必學元素

2.2.1 用例與參與者

既然，用例圖在表達系統對外提供的服務或功能，分析師大概就能合理地推想得到，用例圖上會有代表系統服務的「用例」(use case)，以及代表使用者的「參與者」(actor)，如圖 2-1 所示。

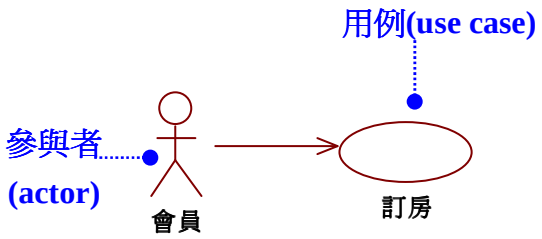

圖 2-1: 用例與參與者

由於，用例與參與者分別位於系統的內外，所以兩者之間隱含了系統範圍。分析師也可以在用例圖面上，明顯標示出系統方框。請看到圖 2-2，用例放置在大方框內部，參與者列在大方框外邊，大方框正式名稱爲「主體」(subject)，代表分析師必須細究的對象，也代表用例應用的環境。

此外，再看到圖 2-2 中，參與者與用例之間有個結合關係，帶箭頭的實線表示單向的結合關係。不過，此處的結合關係只是單純爲了指出參與者可以使用那個用例，分析師倒是不需要用太複雜的角度去解讀它。

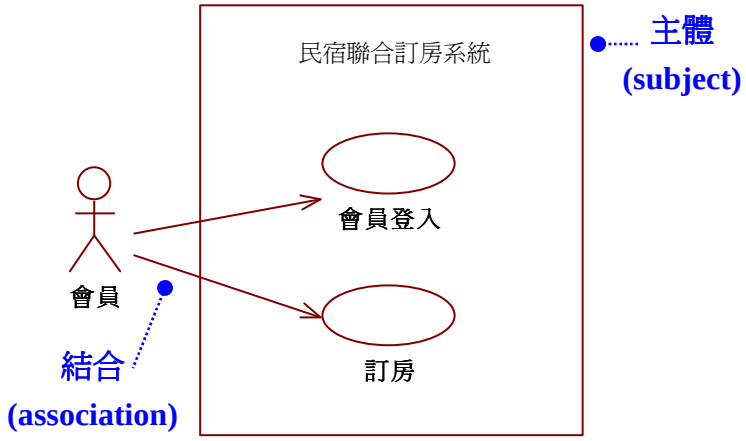

圖 2-2: 主體與結合

2.2.2 啓動者與支援者

事實上，並非只有使用者才算得上是參與者，只要是會與系統(主體)

互動的外部物件都算是參與者，無論是一般的人類使用者、其他系統、硬體設備、外部服務、外部資料庫都可能是參與者。

實務上，我們經常將參與者分爲兩大類：其一是用例的主要服務對象，也通常是用例的啓動者；另一類是扮演支援角色的參與者。以自動櫃員機(ATM)爲例，一般的存戶是啓動者，ATM 背後連線的銀行主機則是支援者，如圖 2-3 所示。

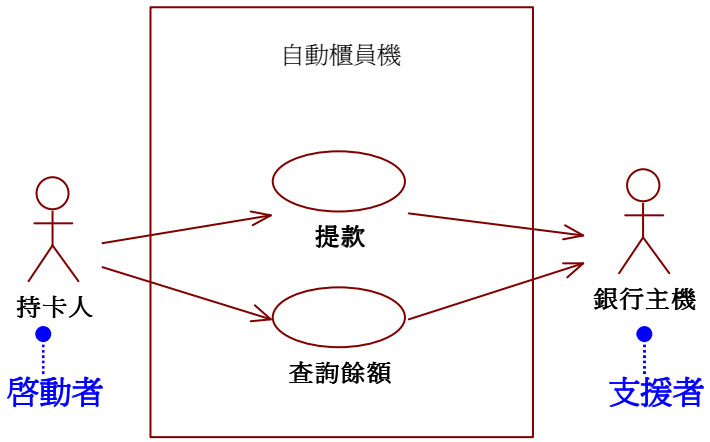

圖 2-3: 啓動者與支援者

再者，我會建議分析師可以善用結合關係的方向性，來指出啓動者或支援者。請看到圖 2-4，從會員指向訂房，代表會員是訂房用例的啓動者；然後，再從訂房指向民宿主人，代表民宿主人是訂房用例的支援者。而且，我還在訂房用例與民宿主人的結合關係上標出「發送電郵或簡訊」，用來提醒開發人員需要發送訂房通知給民宿主人。

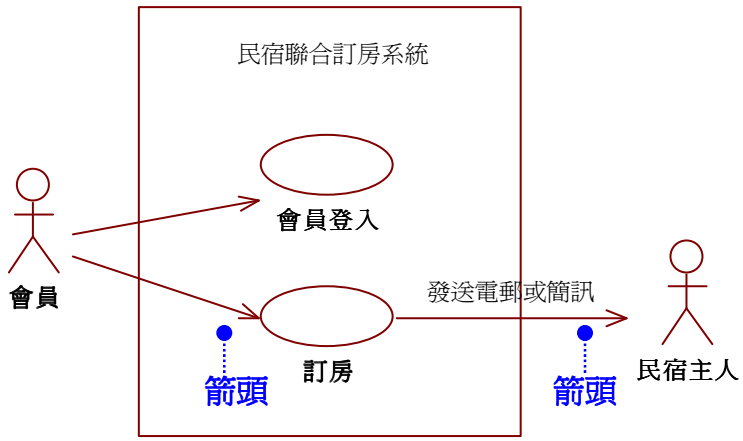

圖 2-4: 善用結合方向

2.2.3 時間代理人

用例一定需要啓動者嗎？有些用例是定時啓動的，設定的時間一到，就會自動執行用例。在這種情況下，建議分析師設置一個「時間代理人」(time agent)，它是個虛擬的參與者，用來指向定時啓動的用例。

別忘了，可以將啓動的時間標示在結合關係旁，這樣就更爲一目了然了，如圖 2-5 所示。訂房系統在月底時，會自動發送電子報給有訂閱的訪客。所有還未加入會員的匿名使用者，都是我們的訪客。當然，會員也可以訂閱電子報，但是我們要強調的重點在於，不具備會員身份的匿名使用者，也可以訂閱電子報。

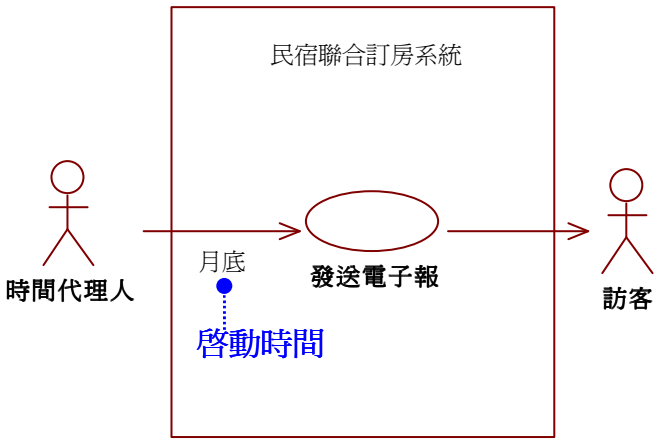

圖 2-5: 時間代理人

2.3 用例敘述

雖說用例圖有畫龍點睛的價值，但我們仍需要把整隻龍描述出來啊，所以除了用例圖之外，分析師還得使用文字描述用例的流程細節，這樣的文字說明，又稱爲「用例敘述」(use case description)。

話說，UML 是一套標準的圖形語言，其中只提出了十四款圖，沒有將用例敘述考慮在內，也當然沒有任何標準的用例敘述格式了。譬如，我在《寫給 SA 的 UML/MDA 實務手冊》和《寫給 SA 的 UML/UseCase 實務手冊》這兩本書中，所設計的用例敘述格式，就跟本書所使用的用例敘述格式不相同。

最簡單的用例敘述，至少會包含一條「主要流程」(basic course)，用來描述正常的使用過程。再者，有時會包含數條「替代流程」(alternative course)，用來描述錯誤的、例外的狀況。除此之外，用例敘述到底要記錄

哪些內容，其格式百花齊放，自由制定。

而且，主要流程的編寫風格，要採用散文式的整塊敘述，還是條列式的步驟敘述，或者是兩欄式的對話敘述，也都沒有標準作法。您要有興趣，從文章、書籍、網路上，都可以找到許多用例敘述格式來參考。

至於，本書採用表 2-2 的用例敘述格式，格式簡單說明如下：

1. 啓動者—遇到多個參與者都可以啓動用例時，用例圖面上放置一個最常見、或最重要的啓動者，但是在用例敘述中，可以將他們全部列出來。
2. 支援者—分析師有時比較難找到支援者，不過別擔心，等這份用例敘述到了設計師手上時，會再次釐清並填上支援者名稱。
3. 主要流程—建議採用條列式的敘述方式，列出參與者對系統說了或做了什麼？接著，系統又給了參與者什麼樣的回應？
4. 替代流程—剛開始寫用例敘述時，不用花太多時間在替代流程上頭，先把主要流程和物件找到，替代流程慢慢補上就可以了。
5. 企業規則—關於客戶指出必須遵守的重要規則、費用的運算公式等，都可以編號並集中管理。
6. 議題與其他—至於，有待解決的議題，以及其他備註說明，都可以放置於此。

用例	會員登入		
啓動者	會員	**支援者**	
主要流程			

1. 會員輸入電郵和密碼。 2. 系統確認會員身分之後，出現歡迎訊息。
替代流程 ☐ 資料不完整：客戶端提醒會員填入資料，直到資料完整才傳送到伺服端。 ☐ 驗證失敗：累積 5 次登入失敗，即鎖定，並出現請會員主動聯絡系統管理員的訊息。
企業規則 BR1：以會員電郵做爲會員代號。 BR2：會員累積 5 次登入失敗，即鎖定該會員帳號。只要登入成功，則失敗次數歸零。 BR3：一個人只能申請一個會員身份。
議題與其他 1. 由於，一個人只能申請一個會員身份，所以將類別圖中的個人與會員合併爲一。

表 2-2: 「會員登入」用例敘述

在撰寫表 2-2 的「會員登入」用例敘述之後，分析師可以立即將新發現的蛛絲馬跡回饋到類別圖中，同步更新了類別圖，如圖 2-6 所示。

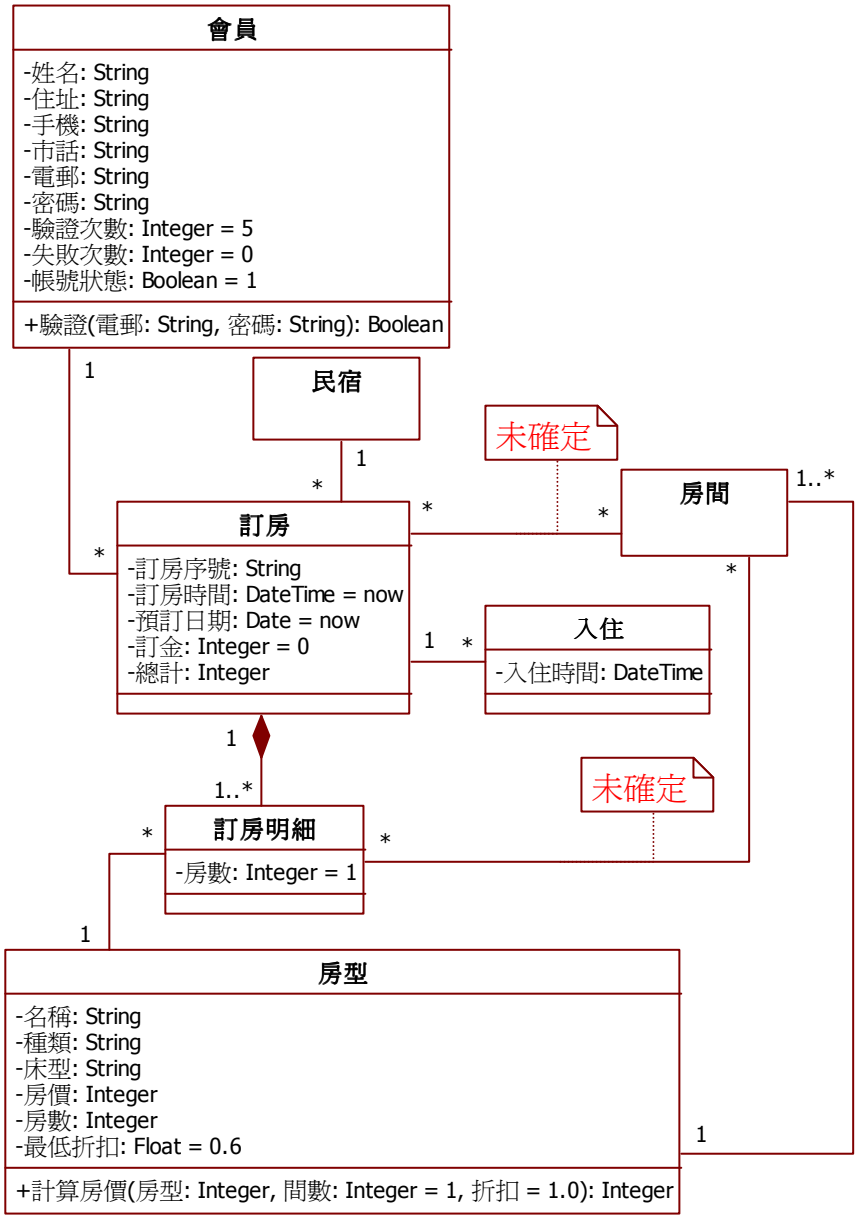
會員
-姓名: String
-住址: String
-手機: String
-市話: String
-電郵: String
-密碼: String
-驗證次數: Integer = 5
-失敗次數: Integer = 0
-帳號狀態: Boolean = 1
+驗證(電郵: String, 密碼: String): Boolean
民宿
未確定
房間
訂房
-訂房序號: String
-訂房時間: DateTime = now
-預訂日期: Date = now
-訂金: Integer = 0
-總計: Integer
入住
-入住時間: DateTime
未確定
訂房明細
-房數: Integer = 1
房型
-名稱: String
-種類: String
-床型: String
-房價: Integer
-房數: Integer
-最低折扣: Float = 0.6
+計算房價(房型: Integer, 間數: Integer = 1, 折扣 = 1.0): Integer

圖 2-6: 更新類別圖

2.4 民宿聯合訂房系統

在開發程序上，我會建議分析師用最短的時間，先產出用例圖，這樣才能交給專案經理做爲估算工時、派工的依據。之後，再來一邊寫用例敘述，一邊修訂類別圖。在《寫給 SA 的 UML/UseCase 實務手冊》書中，我有說明如何使用「用例點」(use case point)來估算工時，這邊就不再重述了。

接下來，我們要先來繪製用例圖，定出訂房系統大致的服務項目之後，再一一編寫用例敘述，並且同步修訂類別圖。

2.4.1 用例圖

通常，我們會先由參與者開始定出相關的用例，透過跟客戶的訪談，一下子就可以找到許多用例。實務上，分析師比較需要精進的技能是切割出顆粒度大小適度的用例，因爲用例顆粒度切得不好，會影響工時的估算，也同時影響了專案的成本。

由於，我們沒有真的去跟客戶訪談，所以就直接先抓些常見的用例。爲了降低圖面的複雜度，此處我們依照參與者的不同，分別列出相關的用例，如圖 2-7~9 所示。

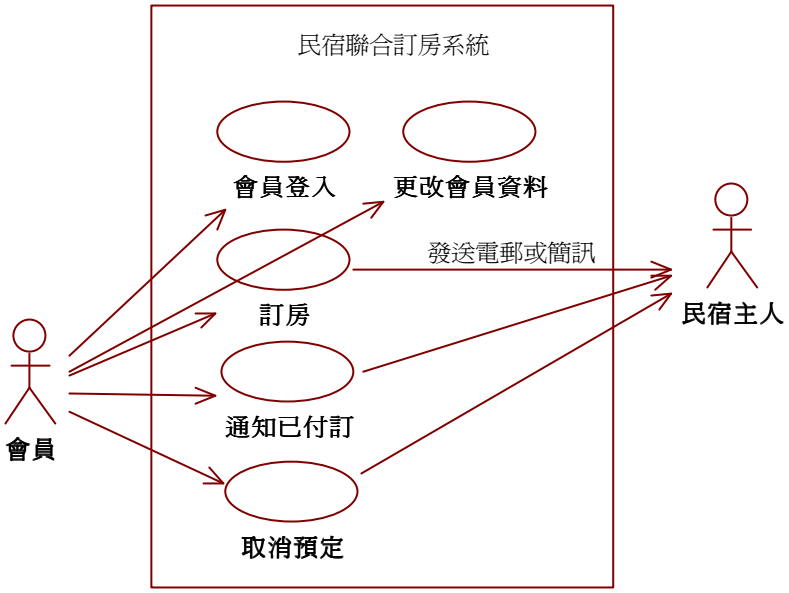

圖 2-7: 用例圖(會員)

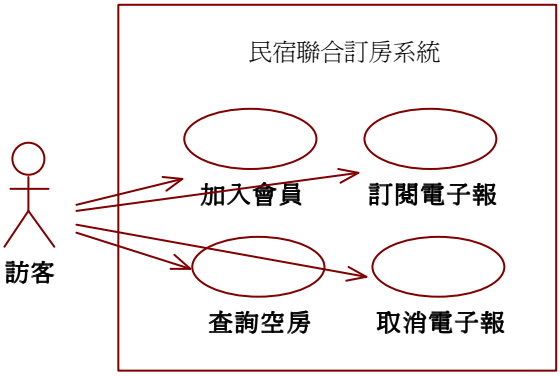

圖 2-8: 用例圖(訪客)

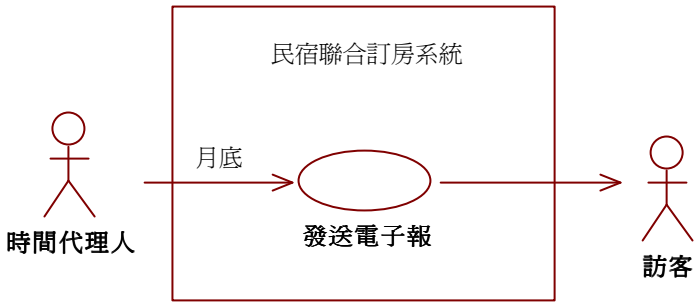

圖 2-9: 用例圖(時間代理人)

2.4.2 用例—會員登入

會員登入的用例敘述，此處並未更動，請您直接參考前述的表 2-2。至於，修訂後的類別圖，則如前述的圖 2-6 所示。

2.4.3 用例—訂房

訂房的用例敘述，記錄在表 2-3 中，相關的說明皆記錄在用例敘述中，不再另行解釋。此外，僅列出修訂過的類別圖局部，不再以全圖出現，如圖 2-10 所示。

用例	訂房		
啓動者	會員	**支援者**	民宿主人
主要流程			

1. 會員挑選一家民宿。
2. 系統秀出這家民宿所有的房型名稱、床型、空房數和房價。
3. 會員挑選預定的房型、房間數以及預訂日期。
4. 系統減少可預訂的空房數，並且新增一筆訂房交易。
5. 系統秀出交易代號、訂金與總價。
6. 系統提醒會員需要 48 小時內付訂金。
7. 系統發送訂房通知給民宿主人和會員。

替代流程

SR1：畫面上必須標記必須填寫的欄位，並且在客戶端先檢驗欄位，並且提醒使用者填寫完整資料，直到必填欄位完整之後，才會回送到伺服端。

企業規則

BR4：訂房交易序號的編碼規則爲「預訂 yyyyMMdd0001」，每日以流水號 0001 起始，每日流水號最大到 9999。

BR5：訂金=總價×0.1。

BR6：會員需在交易成立後，48 小時內付訂金。

議題與其他

1. 編定「系統規則」(System Rule,SR)，做爲整個訂房系統皆須遵守的規則。
2. 企業規則和系統規則將集中管理，用例敘述中僅片面記錄規則初次出現的時刻。

表 2-3: 「訂房」用例敘述

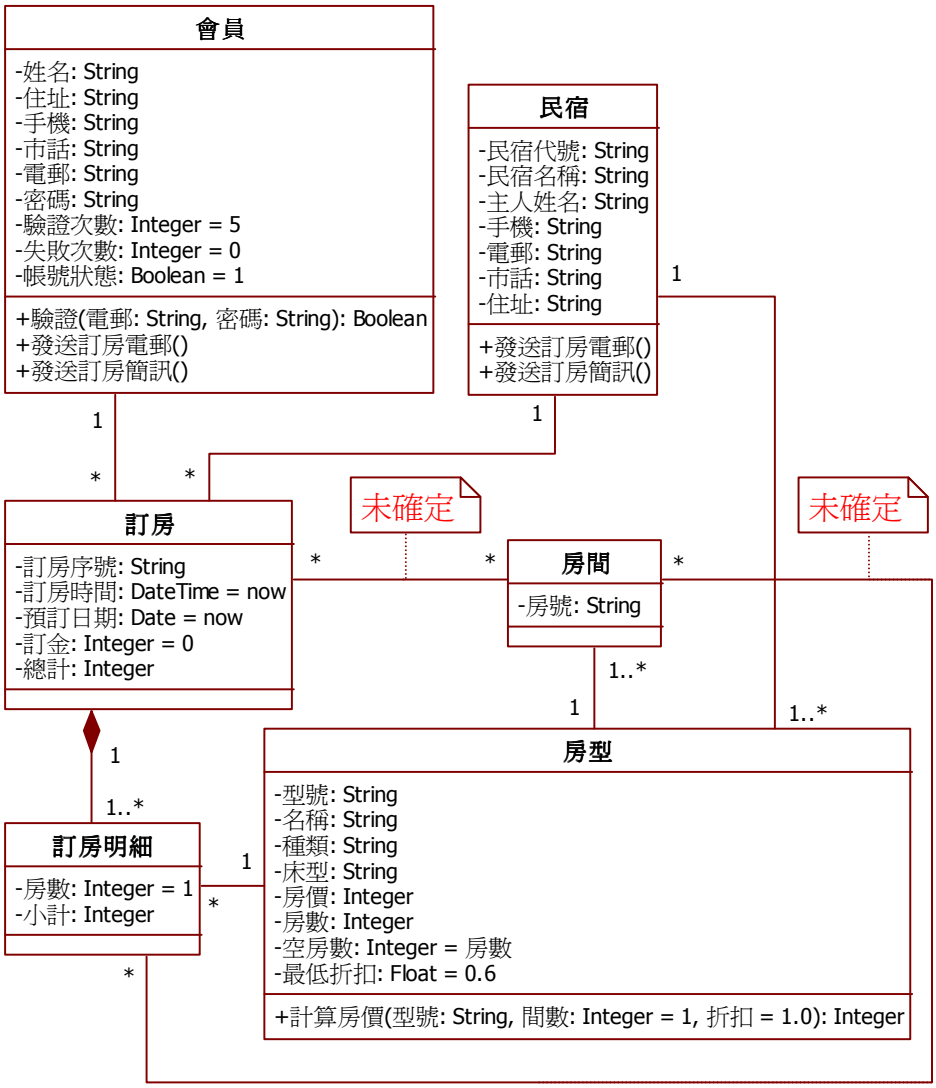

圖 2-10: 更新類別圖(訂房)

2.4.4 用例—通知已付訂

用例	通知已付訂		
啓動者	會員	**支援者**	民宿主人
主要流程 1. 會員選擇一筆未付訂的訂房交易。 2. 會員填入付訂金額、付訂帳號、付訂時間。 3. 系統記錄付訂資料。 4. 系統發送付訂通知電郵或簡訊給民宿主人和會員。			
議題與其他 1. 遺漏了「查詢訂房資料」和「查詢歷史訂單」這兩個用例。 2. 刪掉用例敘述中未使用的欄位。			

表 2-4: 「通知已付訂」用例敘述

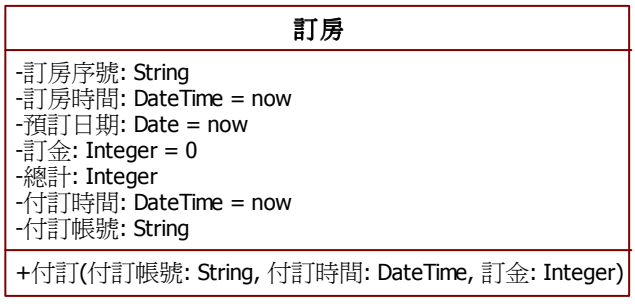

圖 2-11: 更新類別圖(通知已付訂)

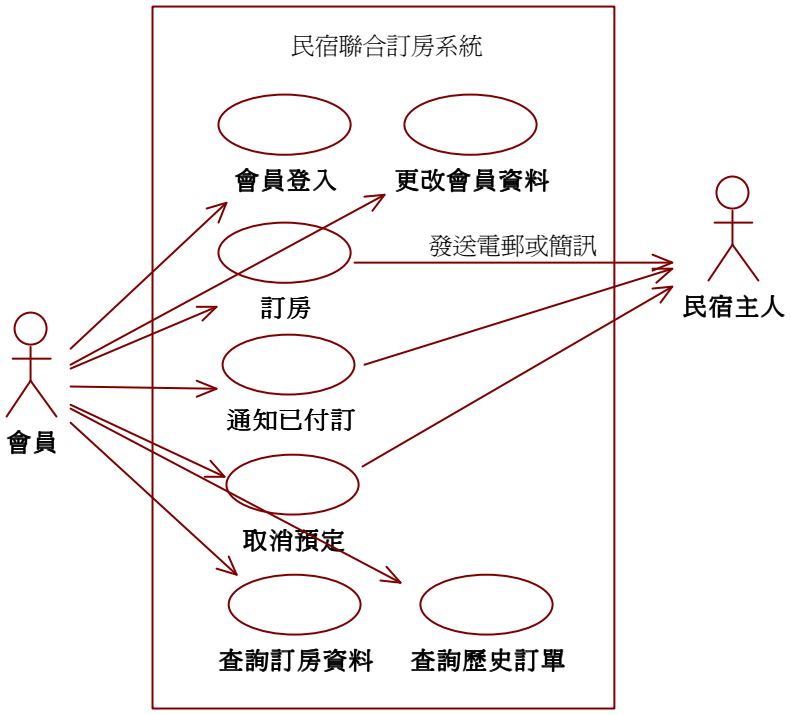

圖 2-12: 更新用例圖(會員)

2.4.5 定時不定量

請原諒我再嘮叨一次，上一章最後的齒輪圖，我一定要在此重秀一次，如圖 2-13 所示。同樣的，這一章要就此打住，因爲類別圖必須經過循序圖的測試。記得要掌握住定時不定量的做法，輪流轉動不同的齒輪，別想要一次完成所有的用例敘述、類別圖或循序圖。取而代之的是，先轉動一下類別圖，接著轉動一下用例敘述，然後再轉動一下循序圖，輪流動一動，才能用不同的角度檢驗彼此、帶動彼此的進展。

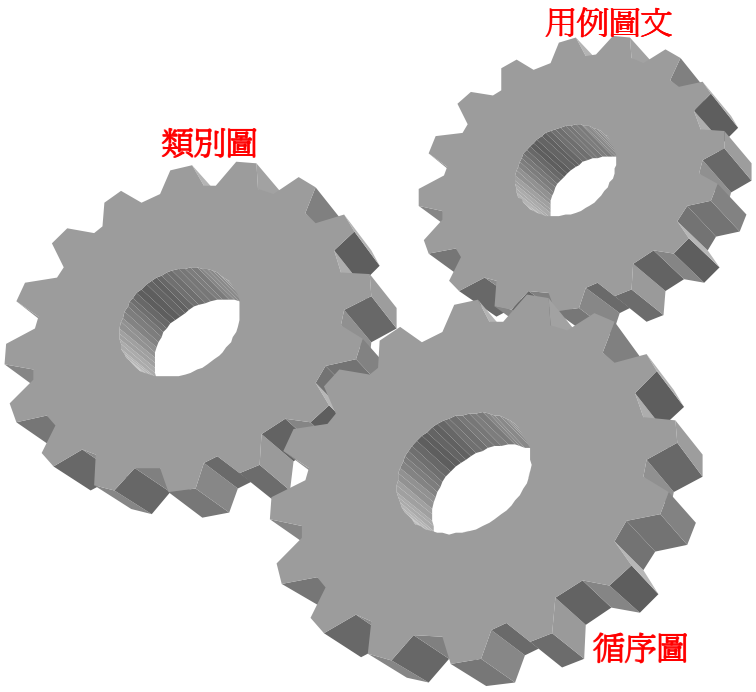

圖 2-13: 帶動彼此

最後，在本章結束之際，我們彙總一下目前進度。目前爲止，我們編寫了「會員登入」、「訂房」和「通知已付款」這三個用例敘述。相關的用例敘述、企業規則和系統規則，請您自行參照前述內容。再者，用例圖請參考圖 2-8~9 和圖 2-12，類別圖則如圖 2-14~15 所示。

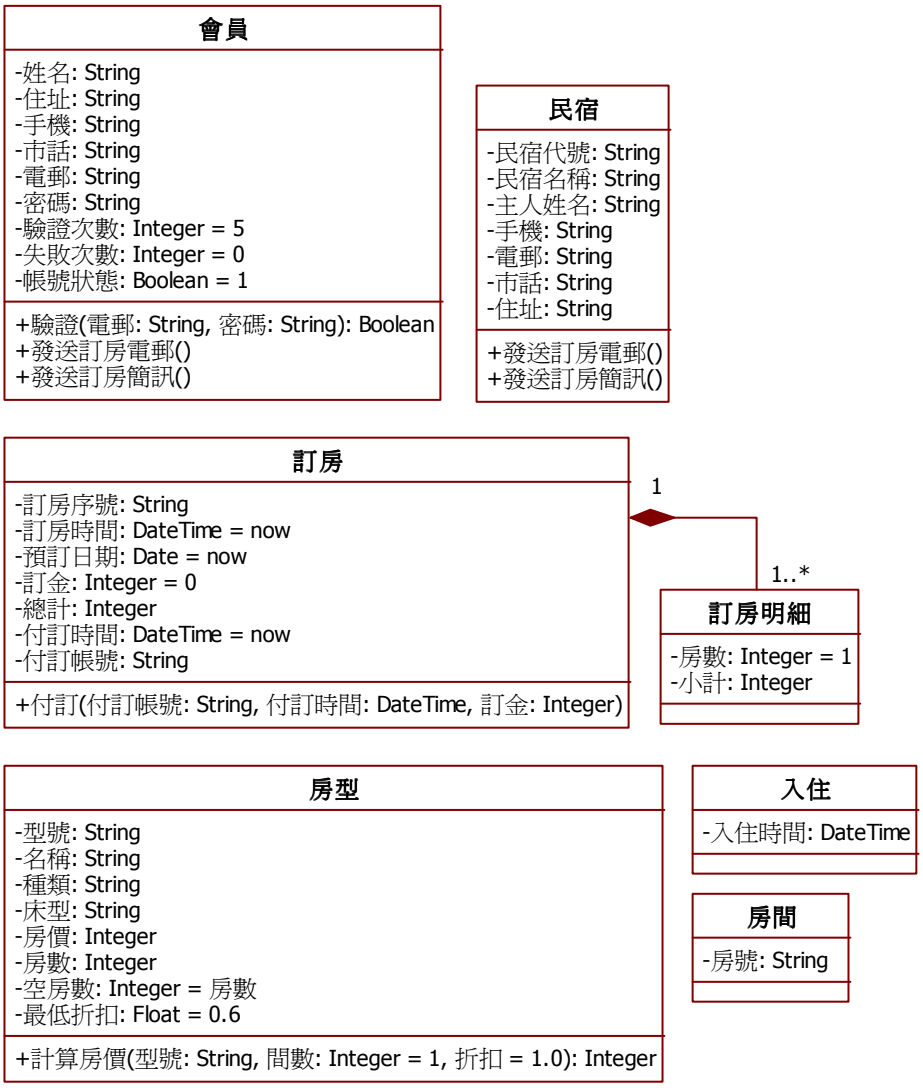
會員
-姓名: String
-住址: String
-手機: String
-市話: String
-電郵: String
-密碼: String
-驗證次數: Integer = 5
-失敗次數: Integer = 0
-帳號狀態: Boolean = 1
+驗證(電郵: String, 密碼: String): Boolean
+發送訂房電郵()
+發送訂房簡訊()
民宿
-民宿代號: String
-民宿名稱: String
-主人姓名: String
-手機: String
-電郵: String
-市話: String
-住址: String
+發送訂房電郵()
+發送訂房簡訊()
訂房
-訂房序號: String
-訂房時間: DateTime = now
-預訂日期: Date = now
-訂金: Integer = 0
-總計: Integer
-付訂時間: DateTime = now
-付訂帳號: String
+付訂(付訂帳號: String, 付訂時間: DateTime, 訂金: Integer)
1
1..*
訂房明細
-房數: Integer = 1
-小計: Integer
房型
-型號: String
-名稱: String
-種類: String
-床型: String
-房價: Integer
-房數: Integer
-空房數: Integer = 房數
-最低折扣: Float = 0.6
+計算房價(型號: String, 間數: Integer = 1, 折扣 = 1.0): Integer
入住
-入住時間: DateTime
房間
-房號: String

圖 2-14: 類別圖(類別)

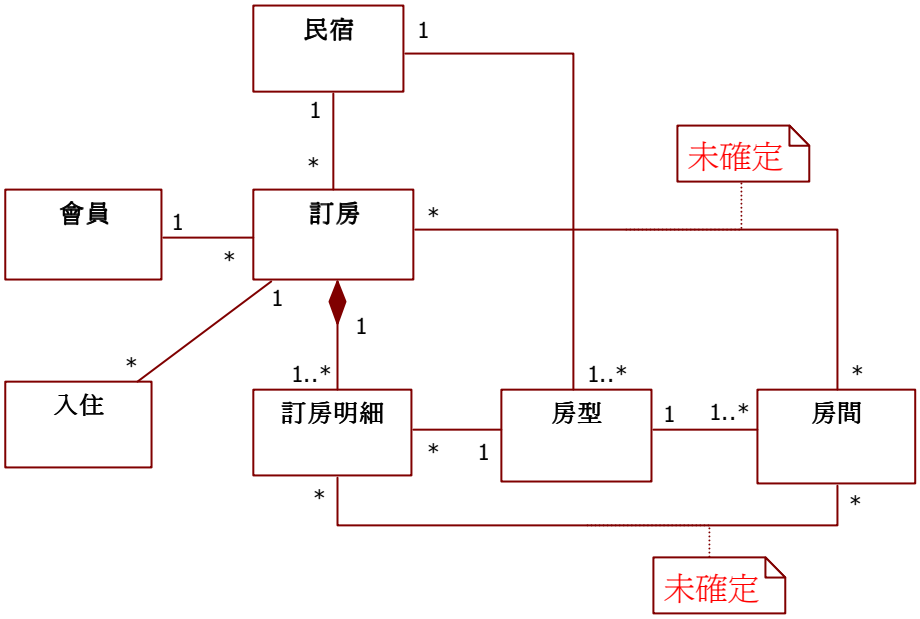

圖 2-15: 類別圖

3
(A3)循序圖

※※

3.1 循序圖

善用前面學到的知識，延伸比較三款 UML 圖，如表 3-1 所示。用例圖和循序圖(sequence diagram)都在表達動態行爲，只是前者表達系統外部物件(參與者)與系統這兩大物件之間的互動，而後者則重在表達系統內部一群小物件之間的互動。

類別圖	用例圖	循序圖
靜態結構 (系統內在結構)	動態行爲 (系統外在行爲)	動態行爲 (系統內在行爲)
類別	參與者、用例	物件(object)
結合	包含、擴充	訊息(message)
交易樣式	用例敘述	BCE 樣式
領域概念	業務流程	概念與流程的結合

表 3-1: 三款圖的比較

也就是說，循序圖結合了類別圖與用例圖兩方，表達了系統在與參與者互動執行某一個用例期間，系統內部的一群小物件的合作情況。因此，分析師可以使用雙叉法，同步進行類別圖與用例圖的分析，然後盡快透過循序圖來整合、調整三方，如圖 3-1 所示。

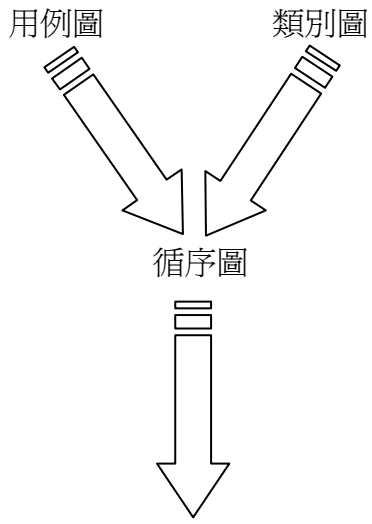

圖 3-1: 雙叉法

如同類別圖的情況，循序圖包含了一大堆元素，不過我們還是主張要聰明地選用。所以，接下來的小節中，分析師可以先學習必學的元素，接著再來看如何套用 BCE 樣式(Boundary-Control-Entity Patterns)，迅速繪製出整合領域概念(domain concept)和業務流程(business process)兩方的循序圖。

3.2 分析師必學元素

3.2.1 一群物件

循序圖是一張柵欄狀的圖形，展現出一群物件(object)的一小段存活期間，如圖 3-2 所示。而物件之間的互動訊息，會橫跨在存活線(lifeline)上，稍後我們會詳細說明「訊息」(message)。

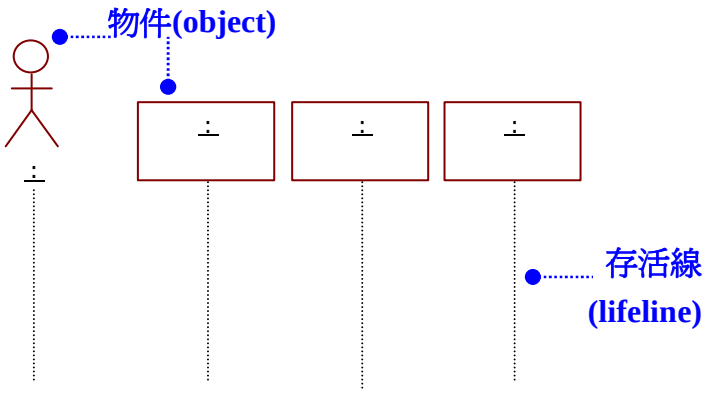

圖 3-2: 物件與存活線

既然，循序圖用在表達執行期間，系統內部一群物件之間的互動情況。因此，在實務上，我們經常使用循序圖來表達某一個用例的執行期間，系統內部的運作情況。至於，系統內部有哪些物件可用，理所當然地規範在類別圖中。請看到圖 3-3，分析師可以使用循序圖來整合用例與類別。

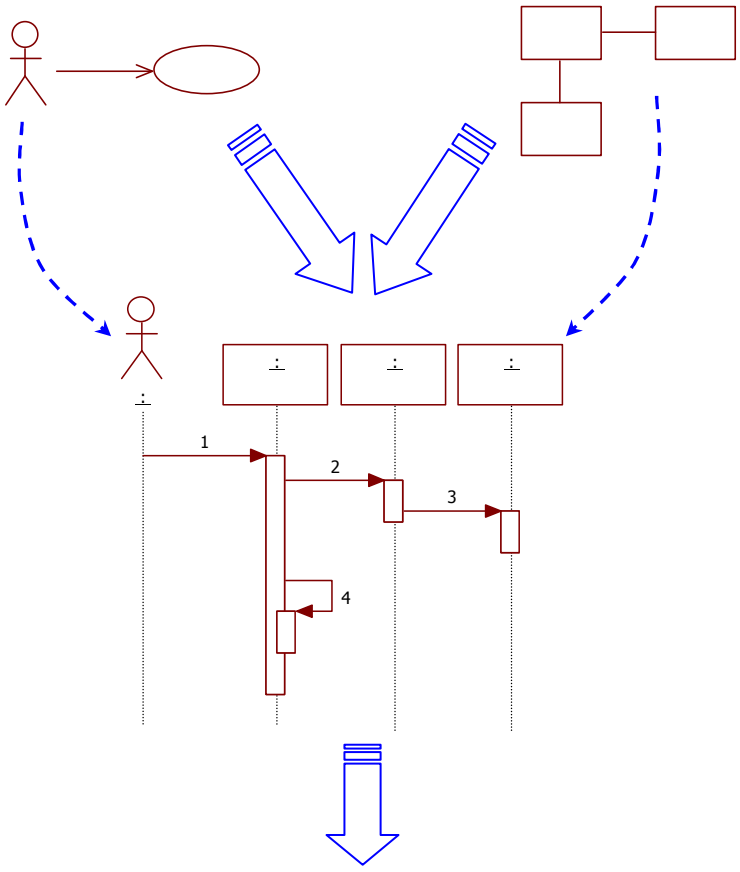

圖 3-3: 循序圖整合了用例與類別

3.2.2 訊息

一旦，分析師讓循序圖對應到用例之後，到底循序圖中的物件該做什麼事情，便可以參考用例敘述中的流程步驟了。而到了循序圖中，原先用例敘述中的流程步驟，將被拆解成物件之間的「訊息」(message)。換言之，物件之間透過互傳訊息來進行合作，彷如真實世界中，一群人之間的互動。

循序圖中的存活線，在概念上並無太多重要價值，僅用來表達這一群物件是存在的、活著的。不過，在循序圖的圖面上，有了一條條的垂直虛線剛好方便訊息橫跨其上，就從這一點來看，存活線就確實有其存在的價值了，如圖 3-4 所示。

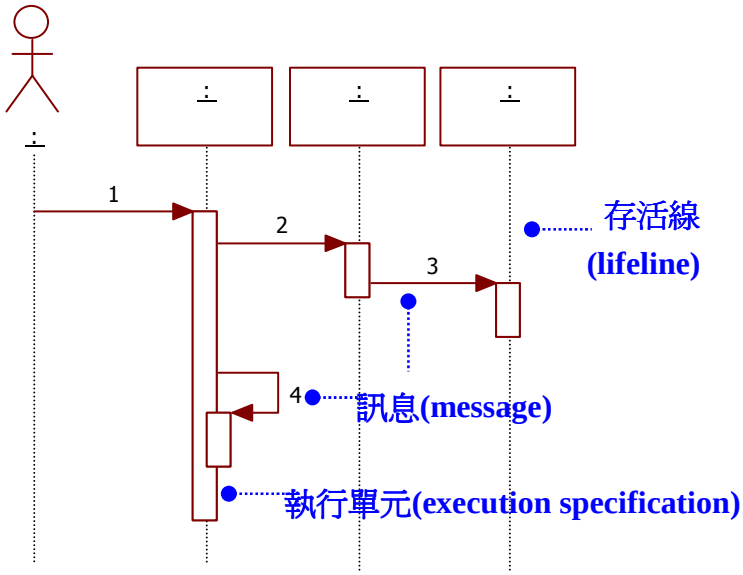

圖 3-4: 訊息與執行單元

再者，存活線上的長條矩形稱爲「執行單元」(execution specification)，用來代表物件收到訊息之後，會負責做一小段事情。物件接收到訊息之後，本來就會開始動作，所以如同存活線的情況，執行單元在概念上並無特殊價值。

但是，在循序圖面上，執行單元可以顯示在某一小段執行期間，物件可能發送訊息給其他物件，或者發送訊息給自己。同樣地，就從這一點來

看，執行單元也有其存在價值。

3.3 BCE 樣式

如果只是這樣，就要分析師畫出某一個用例背後的循序圖，正如同一開始要分析師畫出類別圖一樣困難重重。既然，我們可以套用交易樣式來降低繪製類別圖的難度，當然也可以套用 BCE 樣式來降低繪製循序圖的難度。

BCE 樣式(Boundary-Control-Entity Patterns)是用例技術的創辦人 Ivar Jacobson 大師所提出來的，跟用例技術同時誕生，只是命運大不相同。後來，用例技術被挑中成爲 UML 的一部分，BCE 樣式卻流浪在外。經過多年的演進，BCE 樣式在圖示、名稱上頭，都有一些小變動。不過，概念上沒大變動，圖示和名稱上也都還是能夠讓人一眼辨識出。

其實，BCE 樣式在概念上很淺白易懂，跟知名的 MVC 樣式(Model-View-Controller Pattern)觀念相似。簡單來說，在 BCE 樣式中，將物件分爲三類：邊界類別(boundary class)、控制類別(control class)和實體類別(entity class)，如圖 3-5 所示。

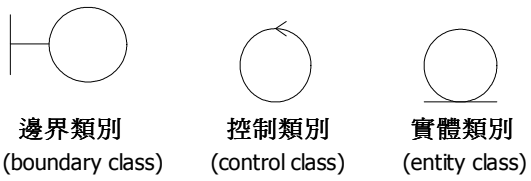

圖 3-5: 邊界、控制、實體類別

- 實體類別—從類別圖來看，前面我們抓出對應領域概念的類別，正屬於實體類別，主要用來保存問題領域中的重要資訊，封裝了跟資料結構和資料儲存有關的變動。
- 控制類別—控制類別對應著用例，用來控制用例執行期間的複雜運算或者業務邏輯(business logic)。所以，通常針對一個用例，就會對應產出一個控制類別。
- 邊界類別—邊界類別用來隔離系統內外，通常負責接收並回應系統內外的資訊。所以，參與者物件只能跟邊界物件互動，不能直接發送訊息給控制物件或實體物件。

因此，套用 BCE 樣式之後的循序圖，會像圖 3-6 的樣子。總之，套用 BCE 樣式時，要記得注意下列四項規約：

1. 針對每一個用例，可以對應產出一個控制類別。
2. 參與者物件只能跟邊界物件互動。
3. 實體物件不能發送訊息給邊界物件和控制物件。
4. 比較特別的是，如果只是單純對資料表進行增刪修查的話，可以不設置控制物件，讓邊界物件直接發送訊息給實體物件，改善整個循序圖的執行速度。

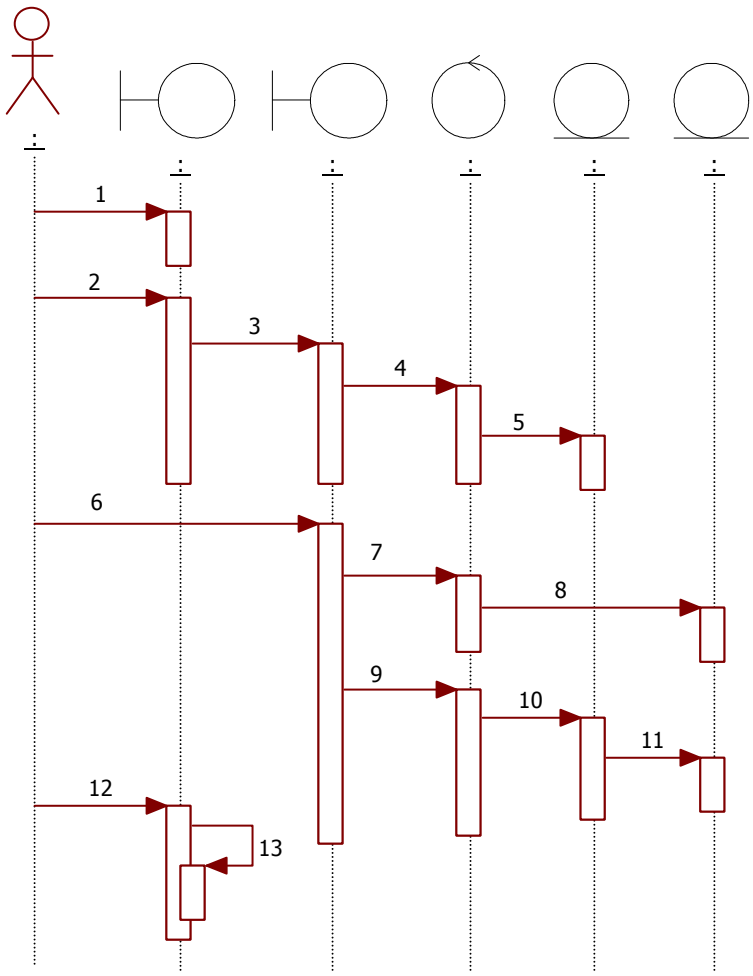

圖 3-6: 套用 BCE 樣式

好了，廢話不多說，我們就馬上套用 BCE 樣式，練習繪製訂房系統的循序圖。Let’s go！

3.4 民宿聯合訂房系統

通常，一個用例至少會有一張描述主要流程的循序圖，至於替代流程，如果太複雜的話，也會獨立畫一張循序圖，要是不複雜的話，直接混在主要流程中也無妨。

在接下來的練習中，爲了簡化設計，我們就只摘出用例的主要流程，其餘部份就先省略了。

3.4.1 用例—會員登入

針對表 3-2 的「會員登入」用例主要流程，一開始我們就直接先擺出屬於它的參與者物件和控制物件；還記得參與者物件不能直接發訊息給控制物件，而且如果需要跟資料庫要資料的話，應該會透過實體物件，所以我們先留了兩個空位放置邊界物件和實體物件，如圖 3-7 所示。

用例	會員登入		
啓動者	會員	**支援者**	
主要流程 1. 會員輸入電郵和密碼。 2. 系統確認會員身分之後，出現歡迎訊息。			

表 3-2: 「會員登入」的主要流程

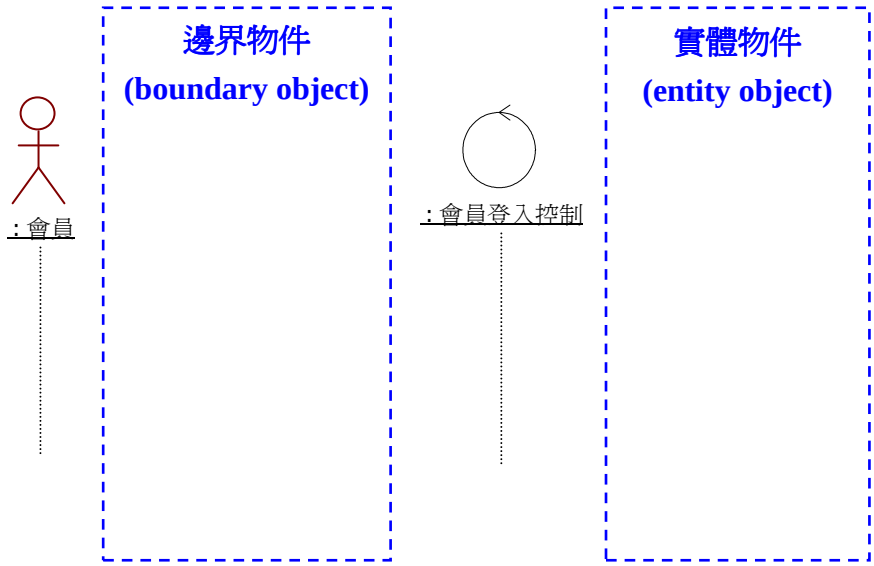

圖 3-7: 「會員登入」的參與者物件和控制物件

接著，我們可以一一分析主要流程的步驟，看看應該將工作分派給什麼物件，如下：

1. 會員輸入電郵和密碼—在參與者物件與控制物件之間，放置一個「登入畫面」邊界物件，用來接收並回應資料給參與者物件。
2. 系統確認會員身分—邊界物件會將參與者傳來的資料轉交給「會員登入控制」物件，請控制物件代爲處理。控制物件需要資料庫裡的會員資料，才能夠進行身分驗證，所以會將外界資料交給「會員」實體物件來處理。
3. 出現歡迎訊息—邊界物件發送訊息給自己，要求顯示出歡迎訊息。

經過上述的分析之後，我們補上了邊界物件和實體物件，並且加上了幾條重要的訊息，如圖 3-8 所示。

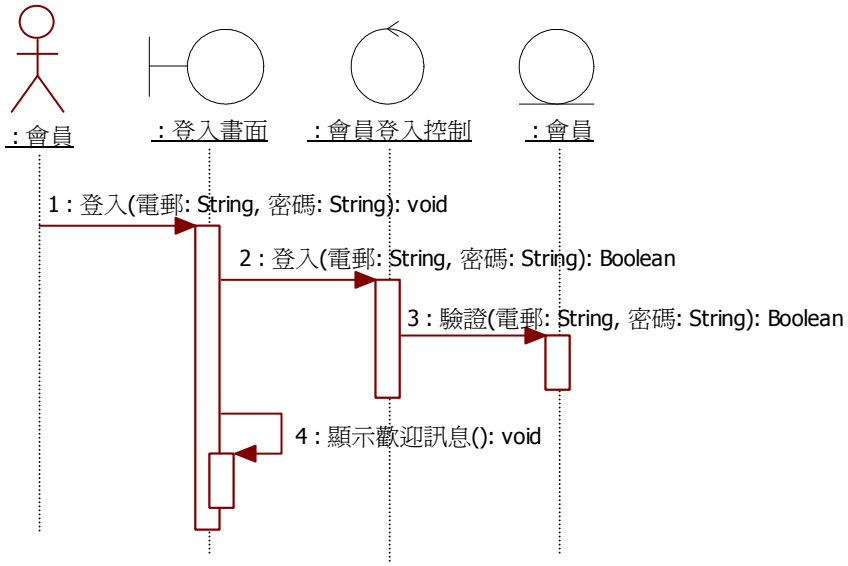

圖 3-8: 「會員登入」的循序圖

仔細看到圖 3-8 中 3 號訊息—驗證(電郵:String,密碼:String):Boolean，因爲先前在繪製類別圖時，我們已經爲「會員」類別定義這個操作了，所以這邊我們可以直接調用(call)。

最後，我們把跟「會員登入」用例有關的 BCE 類別，全列在圖 3-9 中了。別擔心矩形右上角的小裝飾，那只是方便我們辨識類別種類，多數的 UML 工具都有支援的。或者，您喜歡表達成圖 3-10 或者是正規的圖 3-11，都無妨，它們都是類別。

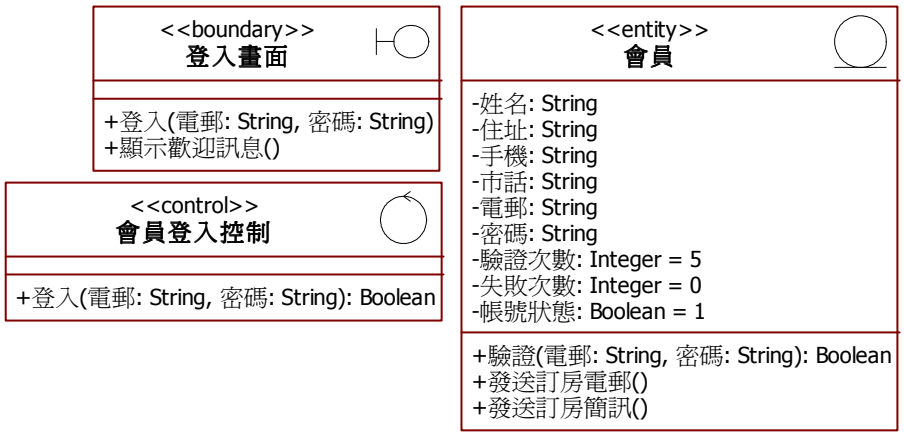

圖 3-9: 「會員登入」用例的 BCE 類別

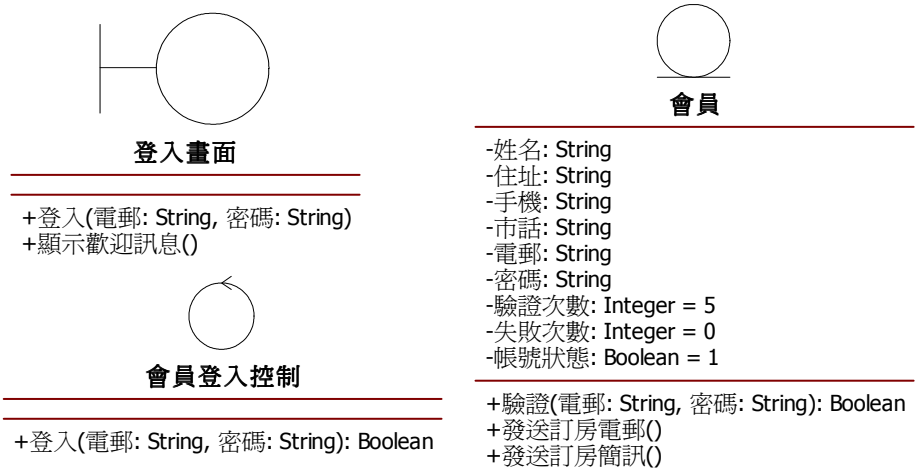

圖 3-10: 使用 BCE 類別圖示

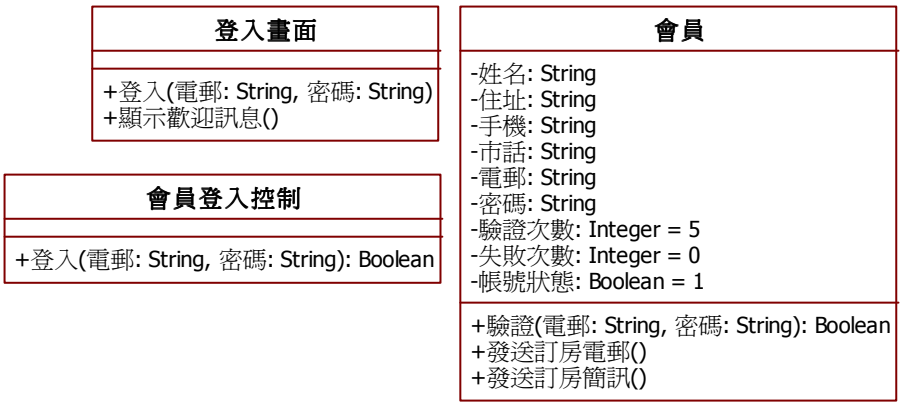

圖 3-11: 使用一般的類別圖示

此外，我們還可以趁機調整一下用例敘述，讓主要流程中的每一個步驟都更明確，或者發現遺漏了什麼步驟或說明，也可以趁這個機會同步修正類別圖和用例圖文，如表 3-3 所示。

用例	會員登入		
啓動者	會員	**支援者**	
主要流程 1. 會員輸入電郵和密碼。 2. 系統驗證會員身分。 3. 系統顯示歡迎訊息。			

表 3-3: 修改「會員登入」的主要流程

3.4.2 用例—訂房

上一個會員登入的用例很簡單，現在我們來練習「訂房」這個複雜一點的用例，先看到它的主要流程，如表 3-4 所示。

用例	訂房		
啓動者	會員	**支援者**	民宿主人
主要流程 1. 會員挑選一家民宿。 2. 系統秀出這家民宿所有的房型名稱、床型、空房數和房價。 3. 會員挑選預定的房型、房間數以及預訂日期。 4. 系統減少可預訂的空房數，並且新增一筆訂房交易。 5. 系統秀出交易代號、訂金與總價。 6. 系統提醒會員需要 48 小時內付訂金。 7. 系統發送訂房通知給民宿主人和會員。			

表 3-4: 「訂房」的主要流程

同樣的，第一步我們先把這個用例的參與者物件和控制物件，先擺到循序圖中，如圖 3-12 所示。特別注意到，訂房用例有兩個參與者物件，我們很確定會員參與者物件一定會存在，但是不太確定民宿主人參與者物件是否需要出現在循序圖中，不過我們還是先把所有的參與者物件都擺上，稍後再看是否要做修改。

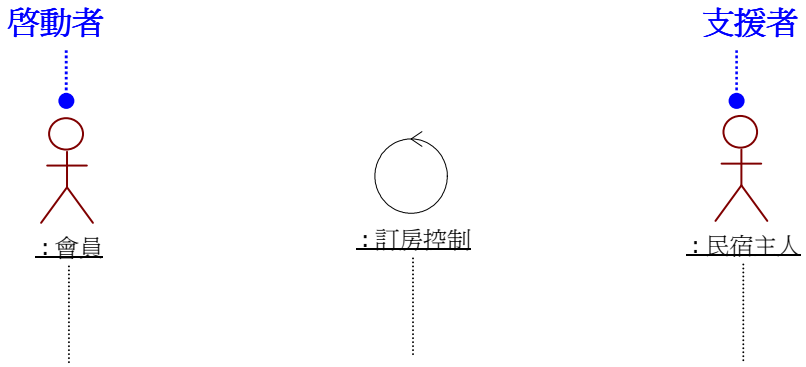

圖 3-12: 「訂房」用例有兩個參與者

先不忙著分析主要流程，我們再來推敲看看，會用到哪些實體物件？訂房有兩個很重要的步驟：第一要先確認有空房，第二步才是新增訂房交易資料。所以，看起來，我們可能會用到房型、訂房和訂房明細。然後，也會需要一個訂房畫面，來讓會員做些預訂房間的動作，如圖 3-13 所示。

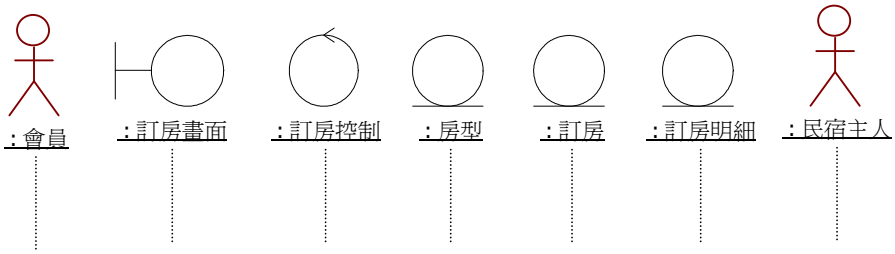

圖 3-13: 加上邊界物件和實體物件

好了，現在我們可以開始來分析主要流程了，如下：

1. 會員挑選一家民宿—這個動作有點玄機，顯然系統一開始必須先提供民宿清單給會員挑選，所以我們可以先放個「民宿清單畫面」邊界物件。
2. 系統秀出這家民宿所有的房型名稱、床型、空房數和房價—然後，經由控制物件，再到「民宿」實體物件，請它告知名下所有的房型資料。

前面沒細想到這兩個步驟，所以這邊我們先就這兩個步驟畫出局部的循序圖，如圖 3-14 所示。

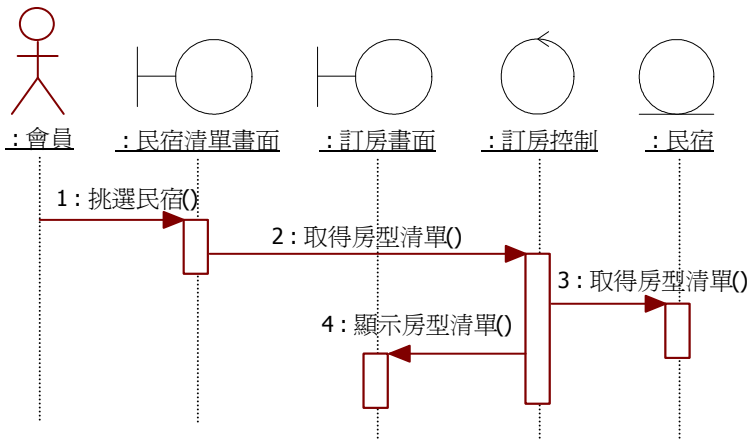

圖 3-14: 查詢房型

3. 會員挑選預定的房型、房間數以及預訂日期—由於，訂房畫面中已經列出房型資料了，所以會員就可以從中挑選預定的房型了。
4. 新增—系統顯示出訂房總價。
5. 修改—系統新增一筆訂房交易。
6. 修改—系統減少可預訂的空房數。

以上步驟，先畫出局部的循序圖，如圖 3-15 所示。

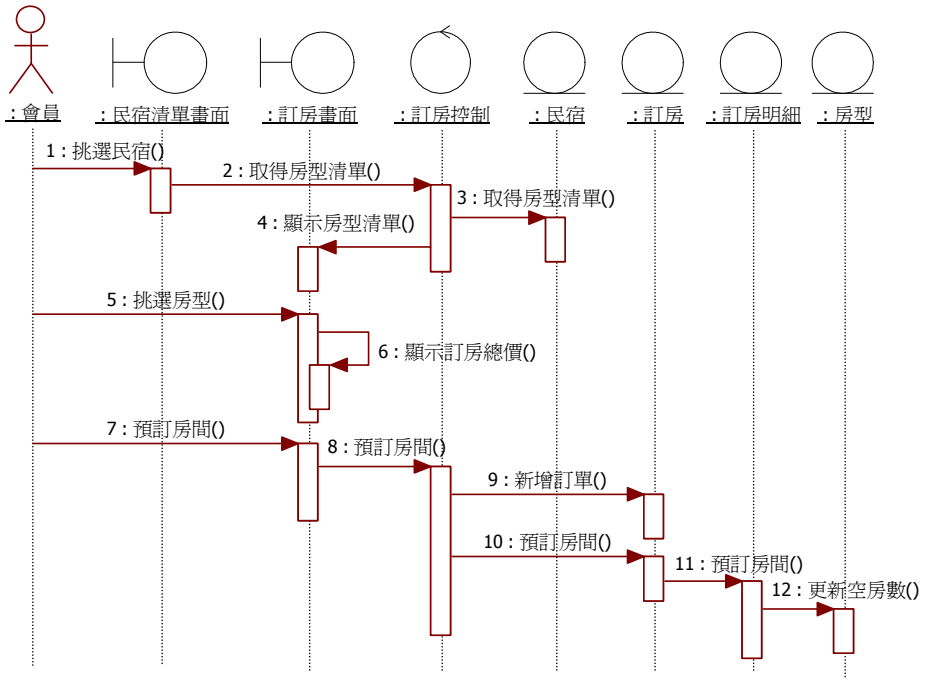

圖 3-15: 挑選並預訂房間

7. 修改－系統發送訂房通知給民宿主人。
8. 修改－系統發送訂房通知給會員。
9. 系統秀出交易代號、訂金與總價。
10. 系統提醒會員需要 48 小時內付訂金。

最後，我們決定刪去民宿主人參與者，加入上述步驟，得到圖 3-16 的訂房循序圖。其實，遇到真正需要提供介面給支援者使用，或者系統真的需要調用支援者所提供的介面的情況下，似乎在循序圖或用例圖中出現該支援者會比較有價值。

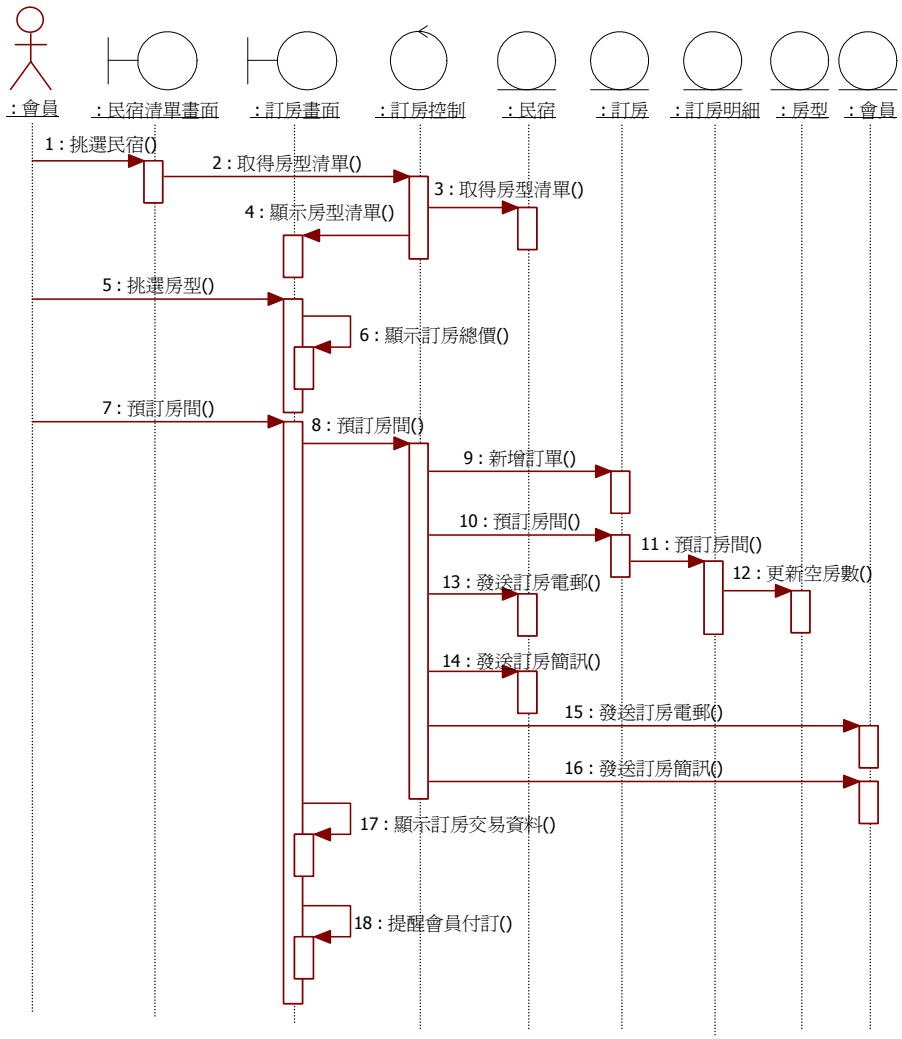

圖 3-16: 刪掉民宿主人參與者

而且，爲了讓整個專案在表達上可以更爲一致，所以我們乾脆刪掉了用例圖文中的民宿主人，同時更新了用例敘述，如圖 3-17 與表 3-4 所示。

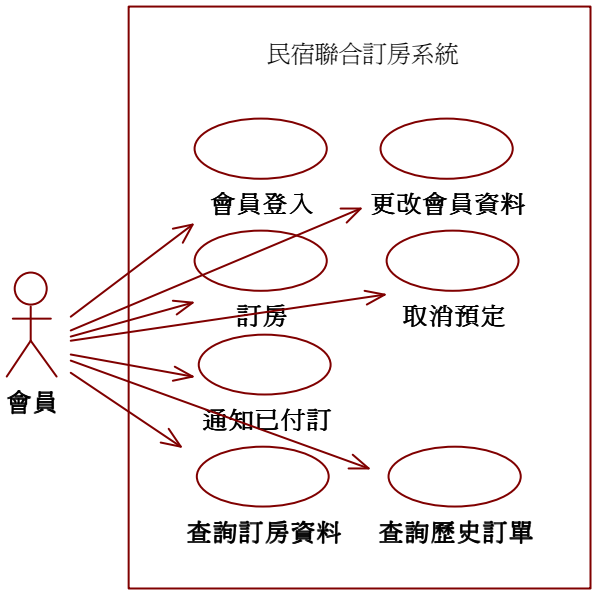

圖 3-17: 刪去了民宿主人

用例	訂房		
啓動者	會員	**支援者**	
主要流程 1. 會員挑選一家民宿。 2. 系統秀出這家民宿所有的房型名稱、床型、空房數和房價。 3. 會員挑選預定的房型、房間數以及預訂日期。 4. 系統顯示出訂房總價。			

5. 系統新增一筆訂房交易。
6. 系統減少可預訂的空房數。
7. 系統發送訂房通知給民宿主人。
8. 系統發送訂房通知給會員。
9. 系統秀出交易代號、訂金與總價。
10. 系統提醒會員需要 48 小時內付訂金。

表 3-4: 「訂房」的主要流程

看起來一切都沒問題之後，分析師可以學習自己用手指和腦袋跑一下整張循序圖，測試一下。我試著跑完這張循序圖之後，發現一個很大的問題：房型物件怎麼會知道某一天的空房數呢？空房數應該會跟著時間變動，而不是一個固定的值。我們可以找出類別圖，確認一下，如圖 3-18 所示。

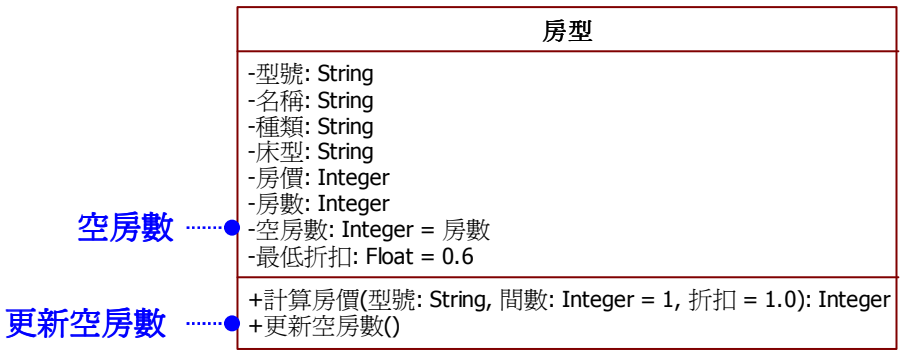

圖 3-18: 房型類別

所以，我們試著修改了房型類別，更新成圖 3-19 的樣子，如下：

1. 新增了一個「預訂紀錄」類別，並且跟房型之間有組合關係。也就是說，當某一個房型物件刪掉時，它底下的所有預訂紀錄物件都會一併被刪掉。
2. 刪去房型類別中的「空房數」屬性，把原先的「更新空房數」操作，改成「更新預訂紀錄(預訂日期:Date, 預訂間數:Integer):Boolean」操作。

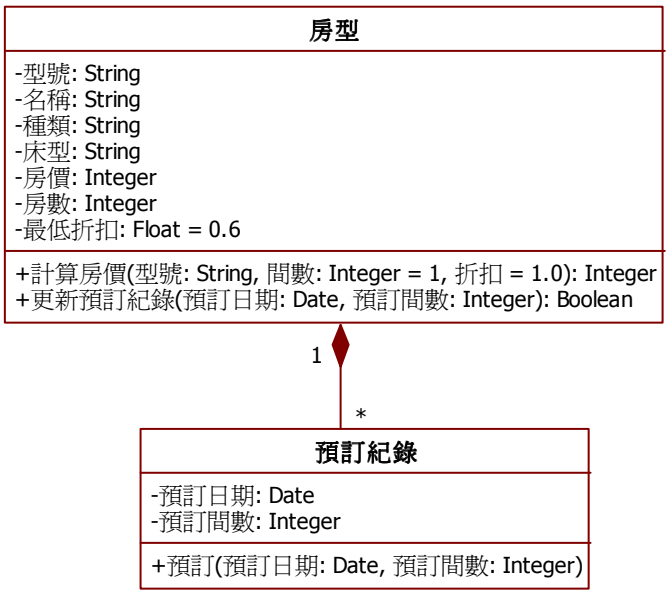

圖 3-19: 新增「預訂紀錄」類別

再者，循序圖也要同步更新，如圖 3-20 所示，多增加了「預訂紀錄」物件，並且讓房型物件在執行第 12 號訊息期間，發送第 13 號訊息給預訂紀錄物件，調用它的預訂操作。

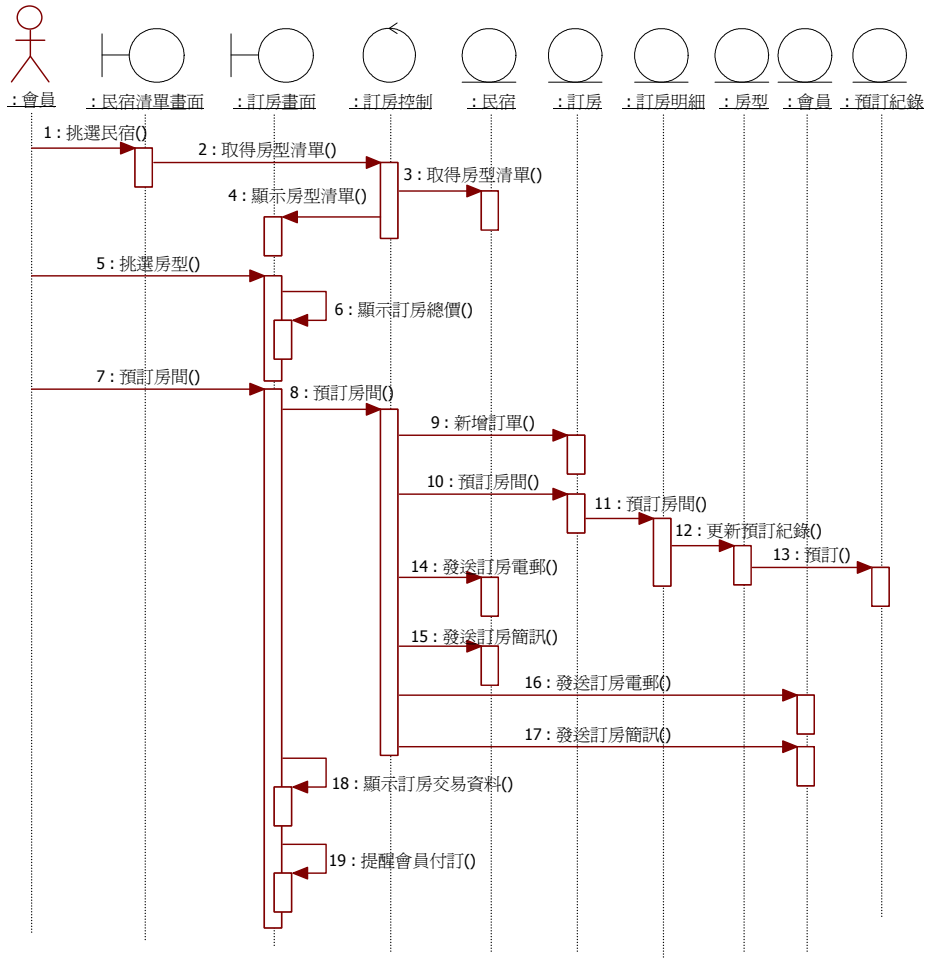

圖 3-20：修改循序圖

最後，請您再看到圖 3-21~22，我們把訂房用例涉及的實體類別、邊界類別和控制類別，一併更新了。

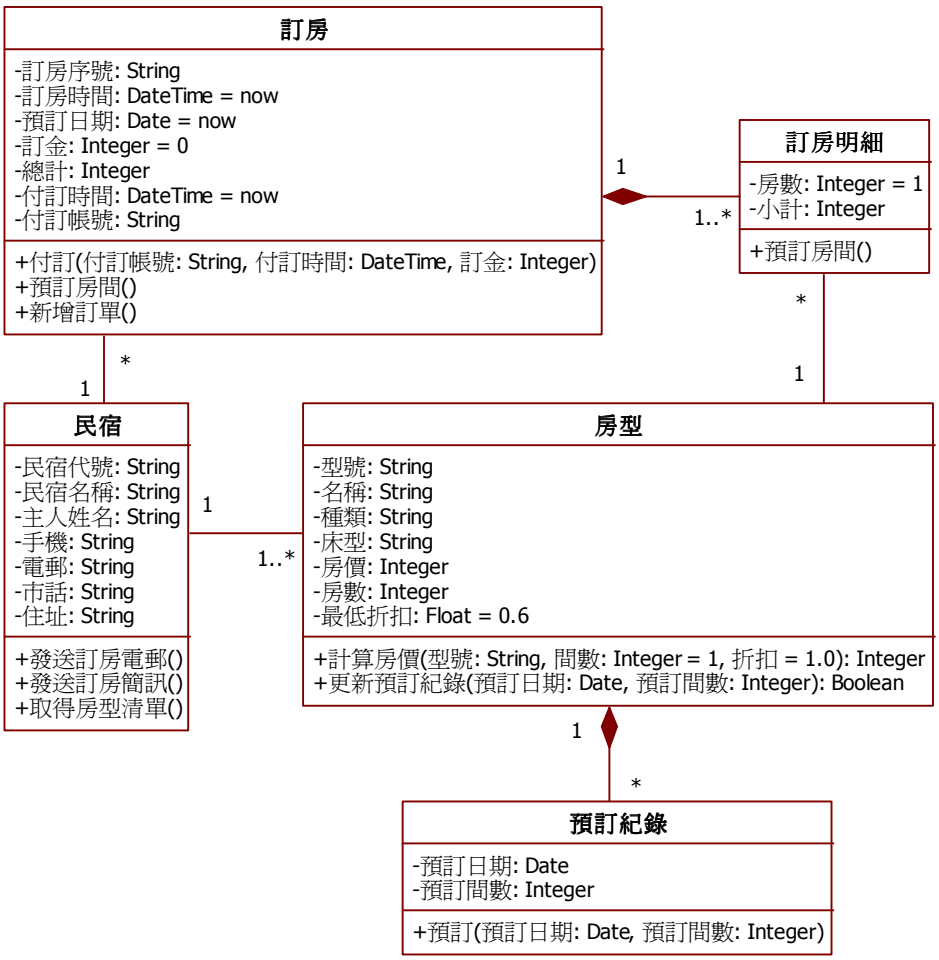

圖 3-21: 「訂房」用例的實體類別

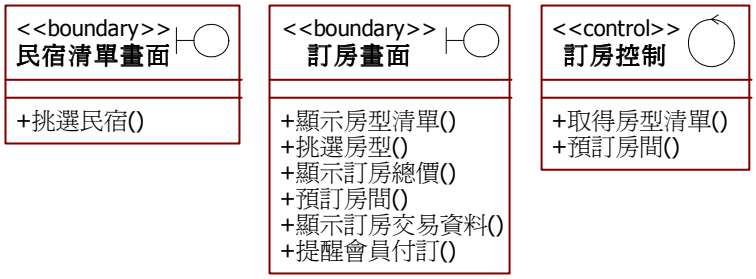

圖 3-22: 「訂房」用例的邊界類別和控制類別

3.4.3 用例—查詢民宿資料

爲什麼我們之前一直強調要盡快進入下一個分析設計階段？那是因爲，從不同的角度、不同的產出，會引發我們發現之前未注意到、或者遺漏的部分。

比方說，前面在分析「訂房」用例的細節時，我們還發現了，我們可以提供一般的訪客查詢民宿資料、查詢房型資料的服務啊，不一定只有會員才能享用到這些服務啊，所以我們又新增了兩個之前沒想到的用例，分別爲：查詢民宿資料和查詢房型資料，更新了跟訪客有關的用例圖，如圖 3-23 所示。

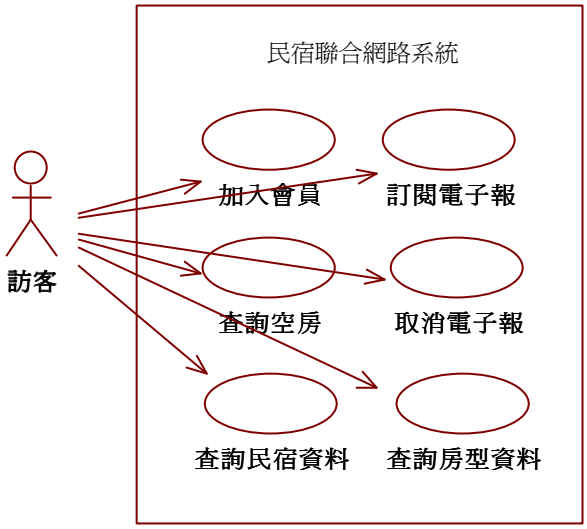

圖 3-23: 更新用例圖(訪客)

接著，我們可以先寫「查詢民宿資料」的用例敘述，並且先找出相關的物件，如表 3-5 與圖 3-24 所示。在圖 3-24 中的循序圖物件，我們打算重用前面用過的「民宿清單畫面」邊界物件，然後再新增另一個「民宿畫面」邊界物件，當然別忘了擺上「民宿」實體物件。再者，由於這個用例只單純去資料庫查詢資料，沒有什麼複雜的控制流程，所以我們也就不爲它設計專屬的控制物件了。

用例	查詢民宿資料		
啓動者	訪客	**支援者**	
主要流程 1. 訪客依照民宿地點、名稱，搜尋符合條件的民宿。			

2. 系統顯示民宿清單，包含民宿名稱、地點、房間數、房間價位。 3. 訪客從中點選某一家民宿，查看民宿資料。 4. 系統顯示民宿資料，除了上述第 2 步驟的資料外，還額外包含民宿網址、簡介、特色、景觀照片。

表 3-5: 「查詢民宿資料」的主要流程

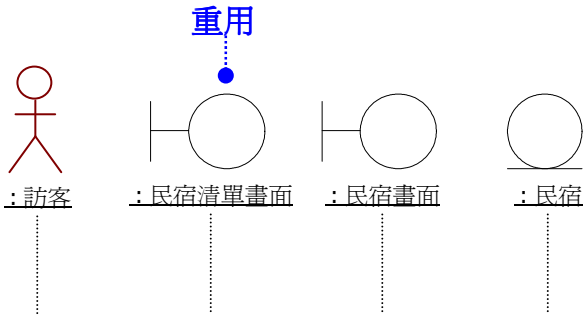

圖 3-24: 「查詢民宿資料」的相關物件

由於，這個用例很簡單，所以我們就直接畫出圖 3-25 的循序圖，並且同步更新民宿類別，如圖 3-26 所示。

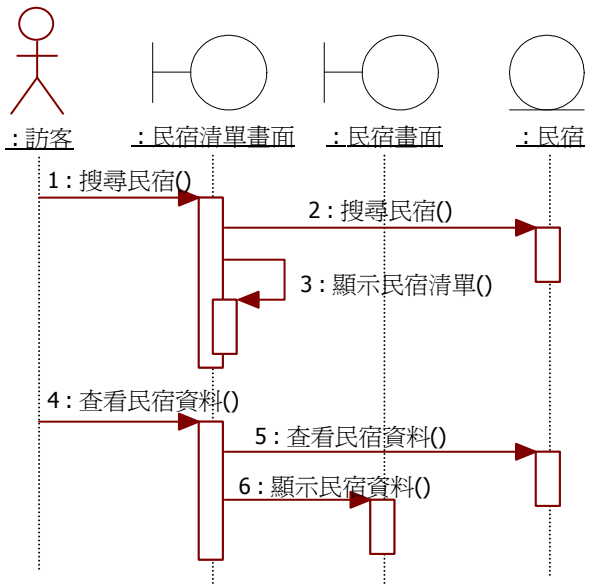

圖 3-25: 「查詢民宿資料」的相關物件

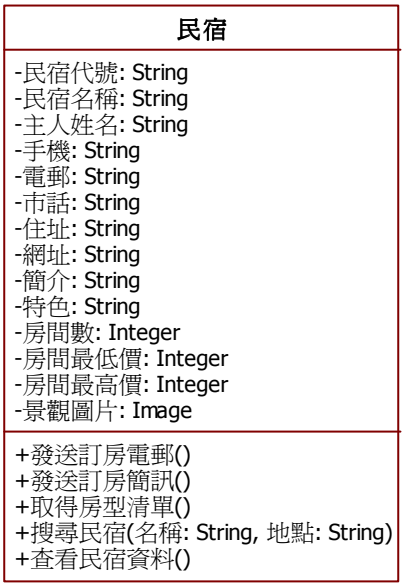

圖 3-26: 修改過的民宿類別

不過，更新完民宿類別之後，我們發現想要多放幾張景觀圖，而且想加上圖片說明，以及圖片的替代文字，所以將原先的圖 3-26 改成此處的圖 3-27。

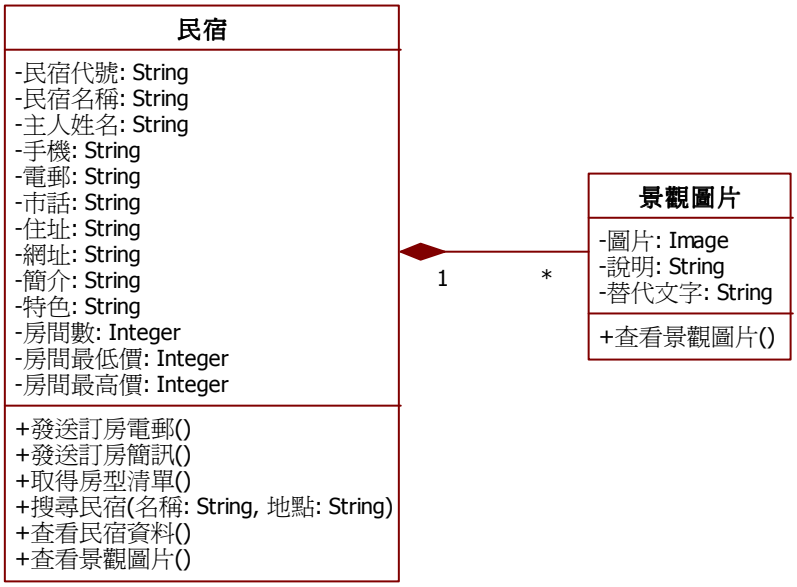

圖 3-27: 增加「景觀圖片」類別

最後，我們再回頭修改一下循序圖，形成圖 3-28 的循序圖，多了查看景觀圖片的訊息。順便列出「查詢民宿資料」的 BCE 類別，如圖 3-29 所示，我們把民宿和景觀圖片的屬性和操作隱藏起來了。

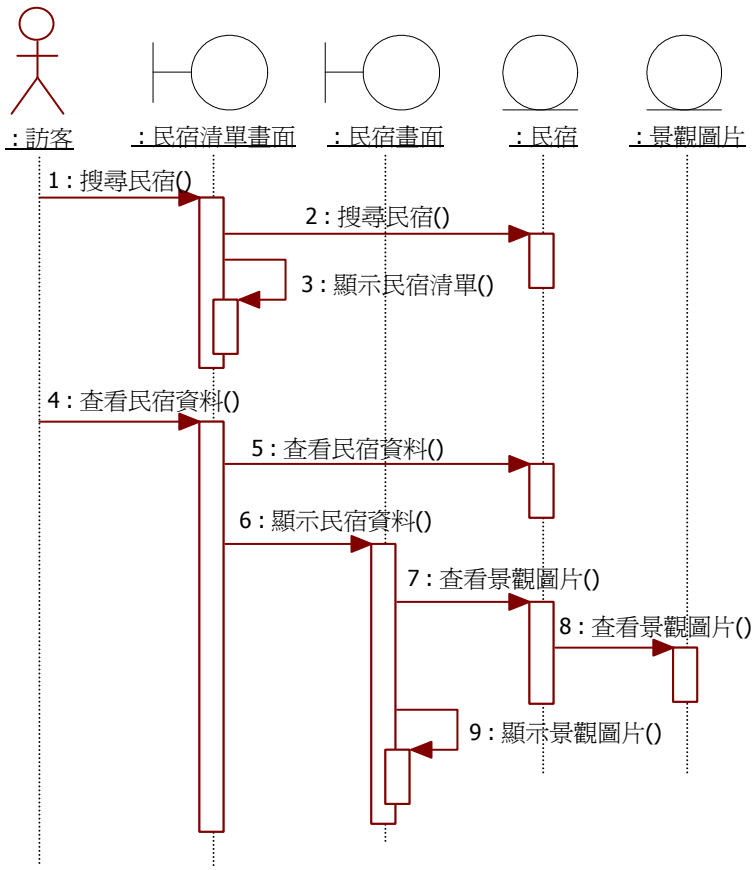

圖 3-28: 增加「景觀圖片」物件

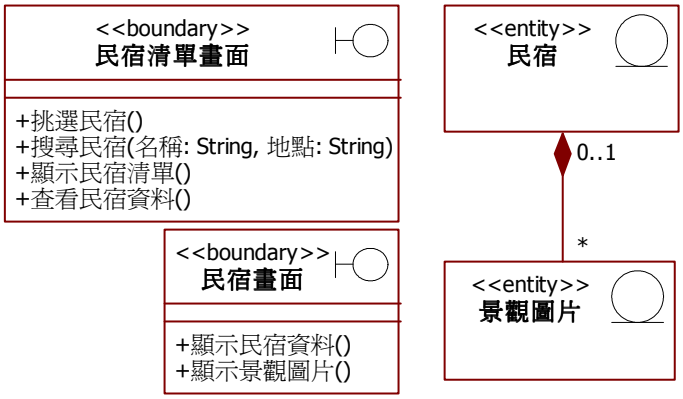

圖 3-29: 「查詢民宿資料」用例的 BCE 類別

3.4.4 用例—查詢房型資料

您會發現分析過前面的「查詢民宿資料」用例後，再來分析「查詢房型資料」用例，更快了，因爲有許多雷同之處，我們就直接複製前述的用例敘述來改，如表 3-6 所示。

用例	查詢房型資料		
啓動者	訪客	**支援者**	
主要流程 1. 訪客依照床型、房價，搜尋符合條件的房型。 2. 系統顯示房型清單，包含房型名稱、床型、房間數、房價。 3. 訪客從中點選某一個房型，查看房型資料。 4. 系統顯示房型資料，除了上述第 2 步驟的資料外，還額外包含房			

間設備、簡介、特色、景觀照片。

表 3-6: 「查詢房型資料」的主要流程

至於，「查詢房型資料」的循序圖，我們同樣直接模仿「查詢民宿資料」的循序圖，如圖 3-30 所示。

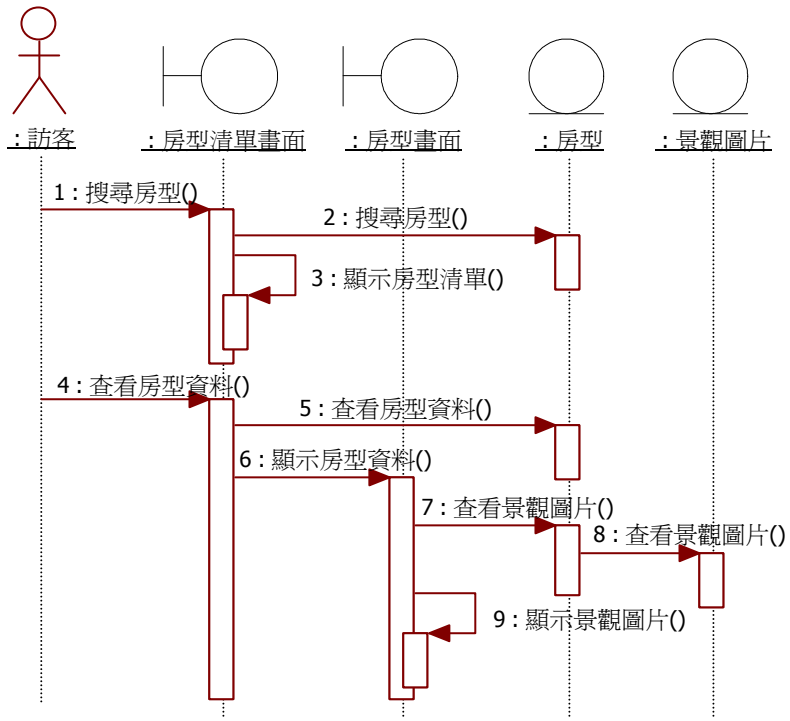

圖 3-30: 「查詢民宿資料」用例的循序圖

不過，類別圖要修改一下，景觀圖片類別中多了一個「場所」的屬性，

我想用它來記錄是民宿的圖片，還是房型的圖片。再者，原先民宿端的個體數目是「1」，現在改成「0..1」了，因爲某一張圖片可能是房型的圖片，不是民宿的圖片，如圖 3-31 所示。最後，歸納列出 BCE 圖，如圖 3-32 所示。

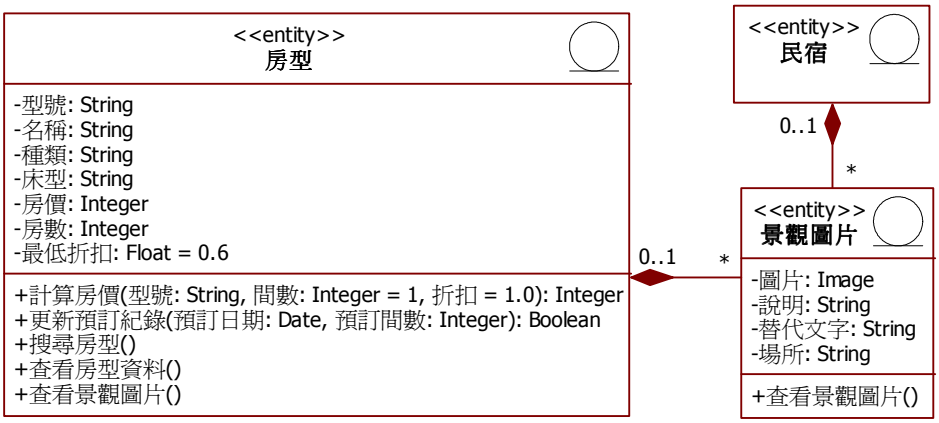

圖 3-31: 修改類別圖

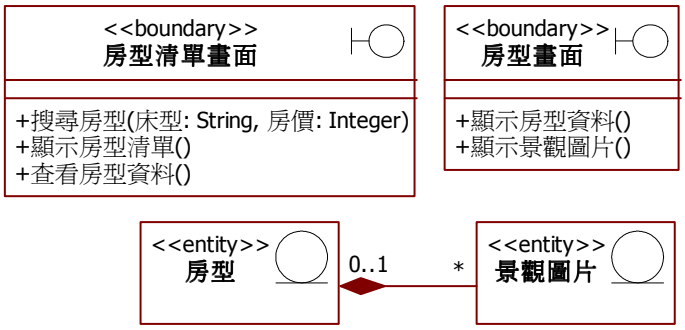

圖 3-32: BCE 類別圖

3.4.5 用例—通知已付訂

最後，我們來看「通知已付訂」用例的主要流程，特別注意到我們已經刪掉它的支援者民宿主人了，跟它的用例圖同步，如表 3-7 所示。

用例	通知已付訂		
啓動者	會員	**支援者**	
主要流程 1. 會員選擇一筆未付訂的訂房交易。 2. 會員填入付訂金額、付訂帳號、付訂時間。 3. 系統記錄付訂資料。 4. 系統發送付訂通知電郵或簡訊給民宿主人和會員。			

表 3-7: 「通知已付訂」的主要流程

首先，分析一下主要流程，大致會用到訂房、民宿和會員實體物件，然後加上一個控制物件、一個邊界物件，如圖 3-33 所示。而且，我們同時列出相關的實體類別，查看一下有哪些屬性和操作可以使用，如圖 3-34 所示。

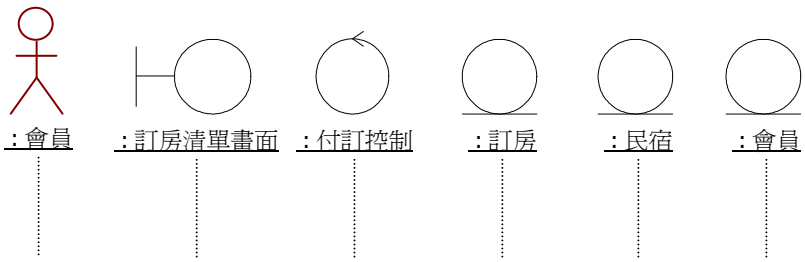

圖 3-33:「通知已付訂」用例的物件

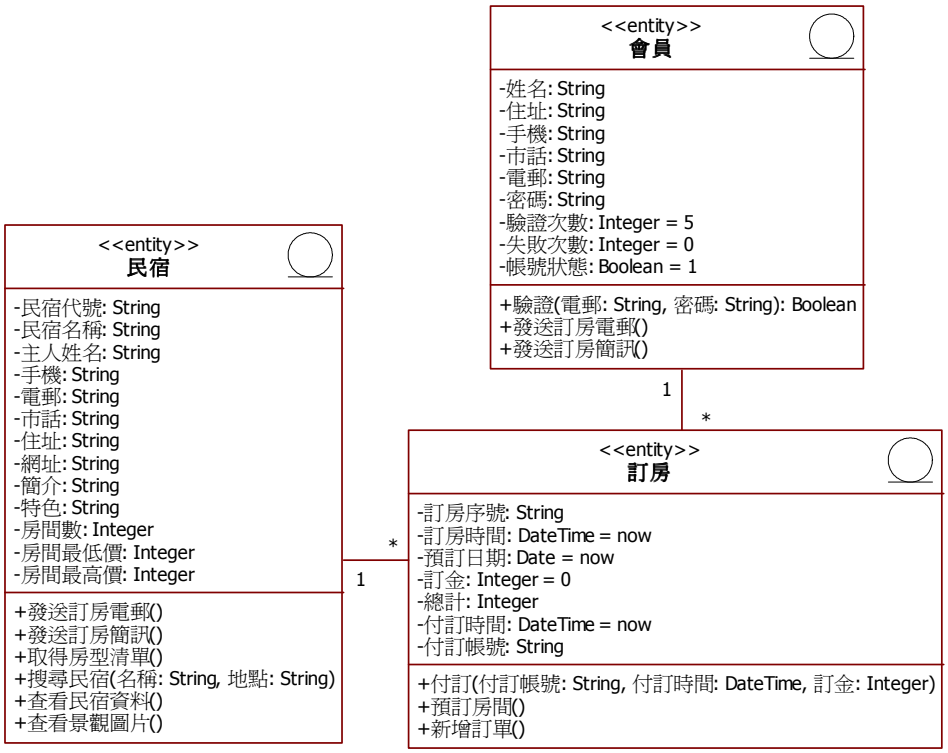

圖 3-34: 實體類別

這個用例不難，所以我們就直接來繪製循序圖了，如圖 3-35 所示。

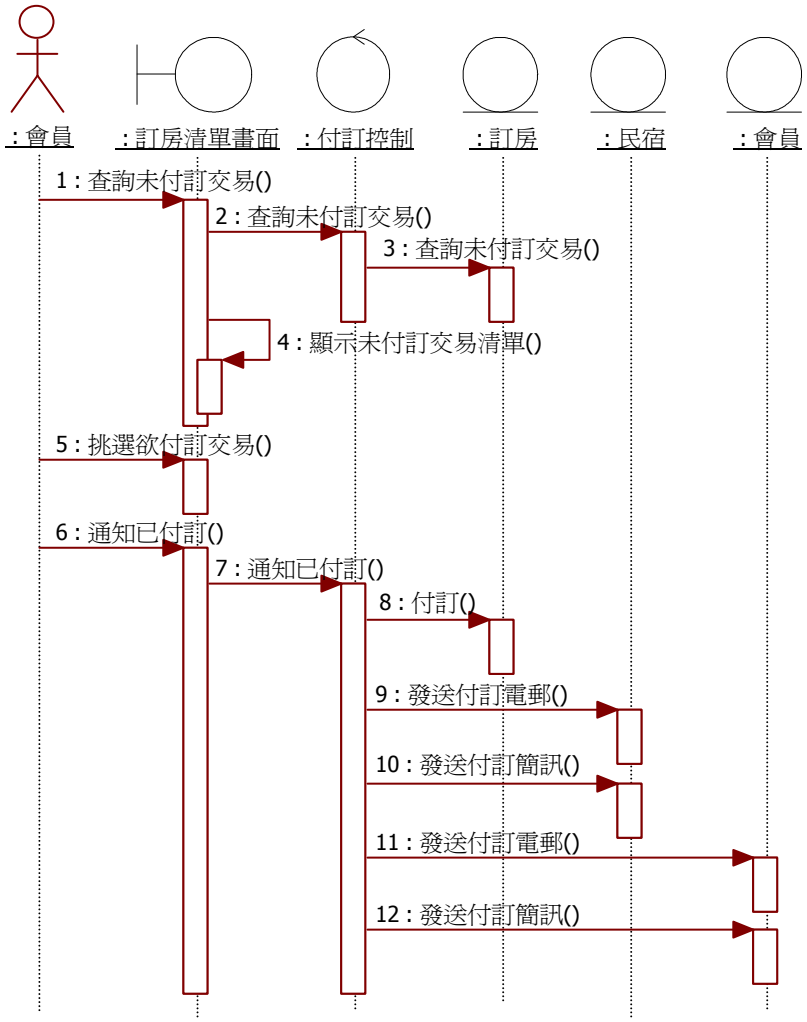

圖 3-35: 「通知已付訂」用例的循序圖

最後，回過頭來小小更新一下用例敘述，如表 3-7 所示。同時，也列出相關的 BCE 類別，如圖 3-36~37 所示。

用例	通知已付訂		
啓動者	會員	**支援者**	
主要流程 1. 會員選擇一筆未付訂的訂房交易。 2. 會員填入付訂金額、付訂帳號、付訂時間。 3. 系統記錄付訂資料。 4. 系統發送付訂通知電郵或簡訊給民宿主人。 5. 系統發送付訂通知電郵或簡訊給會員。			

表 3-7: 「通知已付訂」的主要流程

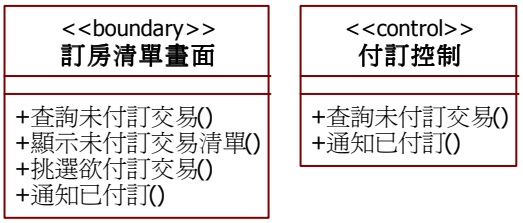

圖 3-36: 「通知已付訂」用例的邊界類別與控制類別

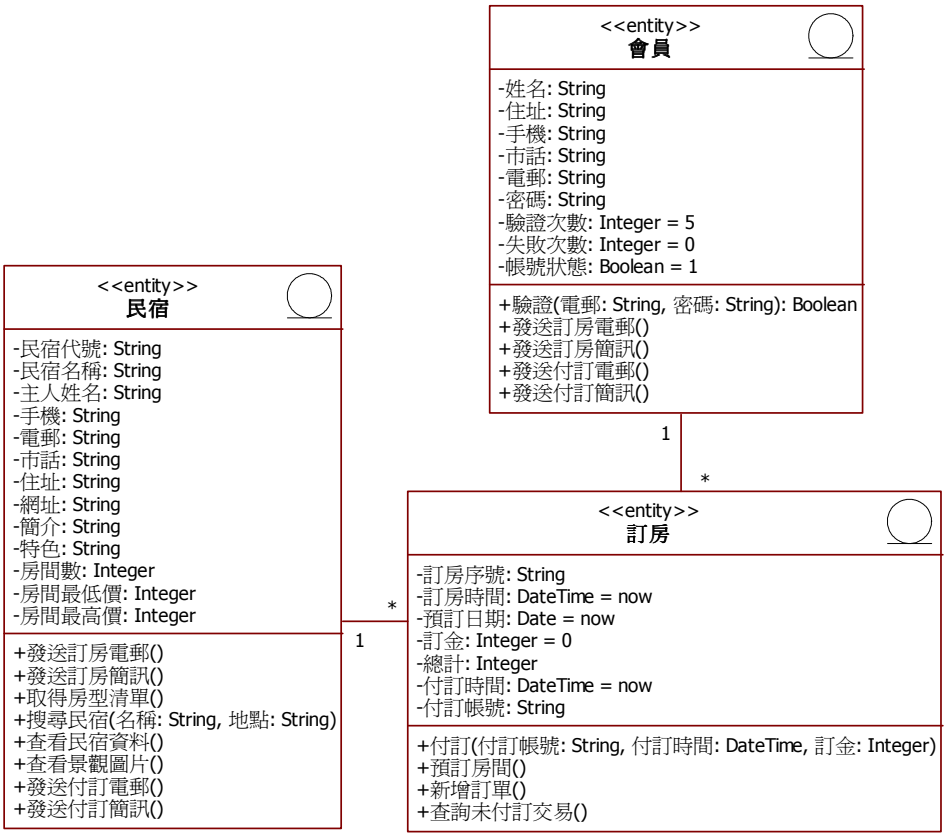

圖 3-37: 「通知已付訂」用例的實體類別

3.5 繪製僞畫面

其實，到目前爲止，關於分析階段的循序圖已經告一段落，本小節要提到「僞畫面」(mockup screen)的主題，倒是與 UML 無關。

實務上，有些分析師習慣繪製僞畫面，雖然這個部分已經跳脫了 UML 範圍，也不是本書要談的主題。不過，我個人倒是非常鼓勵分析師

使用僞畫面跟客戶溝通需求，僞畫面主要可以做爲下列三項用途：

1. 具象溝通—分析師可以透過具體的畫面跟客戶溝通需求。更甚者，如果客戶願意的話，也可以請客戶自行產出僞畫面。
2. 測試畫面—僞畫面可以做爲功能測試畫面。日後，請開發人員依照僞畫面編寫實際的測試畫面，並且定期將測試畫面丟到網站上，邀請客戶一塊參與測試工作。
3. 操作說明—此外，跟畫面有關的操作說明，剛好也可以配著僞畫面一併說明。最好，僞畫面工具可以爲我們自動產出相關文件，做爲開發過程中交付給客戶的操作說明。

如果要將僞畫面跟 UML 結合的話，可以針對代表畫面的邊界類別繪製僞畫面。邊界物件有很多種，畫面物件是最常見的一種，而我們的訂房系統中用到的邊界物件，到目前爲止，都還是畫面物件。

此處，我們要特別區分「畫面雛型」(UI Prototype)與「僞畫面」(mockup screen)的不同。一般所謂的畫面雛型，是指有功能、真的可以執行的畫面，但是功能比真正上線的系統陽春，所以才會稱爲「雛型」(prototype)。

但是，此處的僞畫面，是指沒有功能、無法真正執行的畫面，只是「僞」(mockup)的畫面。正因爲僞畫面是無法執行的假畫面，所以不需要使用功能強大的開發工具來繪製僞畫面，甚至使用一般可以繪製簡單圖示的文字編輯器就可以製作僞畫面了。

雖然，UML 並沒有適合表達僞畫面的圖款，所幸網路上可以找到許多相關的僞畫面工具，您可以一一下載來試用看看。接下來，我們會簡單介紹三款付費或免費開源(open source)的軟體，建議您下載這些軟體工具來試用看看。

3.5.1 MockupScreens

MockupScreen(http://www.mockupscreens.com)是一套付費的僞畫面工具，您可以在它的官方網站下載 30 天試用版。我個人相當喜歡 MockupScreens，甚至購買了它的個人版，喜歡的原因有下列幾項：

1. 操作簡潔—這個原因純屬個人習慣，我比較喜歡簡潔的操作和畫面，這樣的軟體學習曲線較低。
2. 按鍵換頁—雖然僞畫面是假的畫面，可是 MockupScreens 可以讓我們在畫面元件背後設定跳到其他畫面。比方說，我們可以在按鍵背後設定跳到另一頁，這樣在展示僞畫面時，按下按鍵就可以換畫面，模擬執行時的換頁情況。
3. 操作說明—針對每一個畫面元素，可以加上說明文字，形成簡單的操作流程說明。
4. 特殊標誌—MockupScreens 提供一組像小圓貼紙的標記，方便貼在畫面的任何地方，還可以針對標記附上說明文字。
5. 產生文件—針對上述一切的僞畫面、文字說明、特殊標記，都可以匯出成 pdf 文件檔。如果，匯出成 html 檔案的話，雙擊有設定換頁的畫面元素，還可以在瀏覽器中展示出換頁的效果。

請您看到圖 3-38，這是 MockupScreens 的主要畫面，左手邊放置畫面元件，中間爲畫面繪製區，右手邊則是所有畫面的列表。MockupScreens 的功能和畫面簡潔，十分建議您嘗試著說服客戶，或許客戶會願意自己動手使用這套軟體來繪製系統的畫面，以便提高開發團隊與客戶雙方的溝通品質與效率。

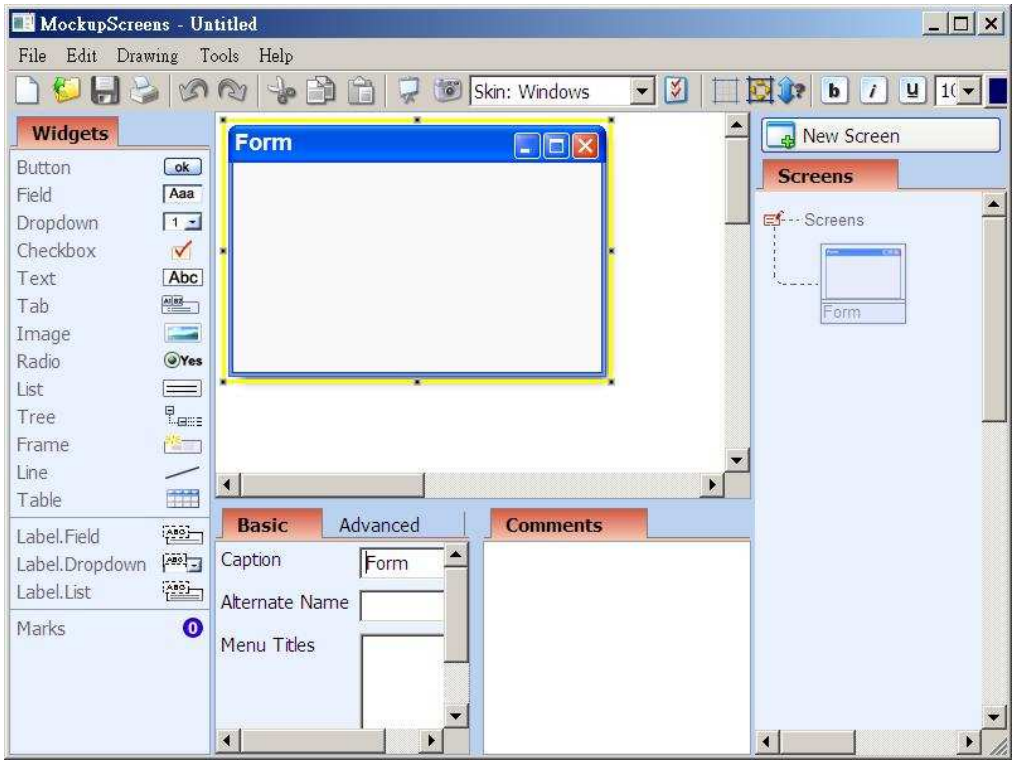

圖 3-38: MockupScreens 的主要畫面

我購買 MockupScreens 4.23 個人版，如圖 3-39 所示。使用時，發現一些小問題，希望後續改版時，作者可以將這些小問題解決掉。

圖 3-39: MockupScreens 4.23 版

此處，特別將 MockupScreens 的問題記錄下來，若您的專案有考慮選購 MockupScreens 的話，還請您多試用一陣子再決定。記錄如下：

1. 匯出 Pdf 檔—MockupScreens 有提供匯出 Pdf 檔的功能，可是有時成功，有時會匯出失敗，或者是匯出結果有亂碼。我的噗友也反應過有這種狀況，所以懷疑是 MockupScreens 本身的 bug。
2. 系統不穩定—MockupScreens 執行一段時間之後，會越來越慢，我甚至遇到過當掉的情況。
3. 標記由上而下排序—MockupScreens 會自動爲標記排序，不過是由上而下依序。如果想改變序號的話，只能改變畫面元件放置的位置，無法手動改變序號。

可惜 MockupScreens 有上述問題，希望作者可以盡快解決，否則的話，我會建議您的專案還是購買等級好一些的僞畫面工具，或許會比較穩定，也比較妥當。

其實，我認爲這樣的軟體，簡單穩定就好，不需要功能太複雜或龐大，最好功能簡單到可以讓客戶動手操作。如果，您是軟體公司的員工，我更建議您可以跟公司談談，看看能不能公司自己寫個類似 MockupScreens 的軟體，相信一定可以大大提升專案開發的品質與效能。軟體公司專門在幫客戶寫系統，也寫個有助於開發專案的小軟體給自己用，應該是相當值得投資的。

3.5.2 Balsamiq Mockups

MockupScreens 最美中不足的是它的畫面元素太少，所以另一套有著可愛畫面元件的 Balsamiq Mockups(http://www.balsamiq.com)，也是您可以試用看看的僞畫面工具。您可以到它的官方網站，立即線上試用，如圖 3-40 所示。

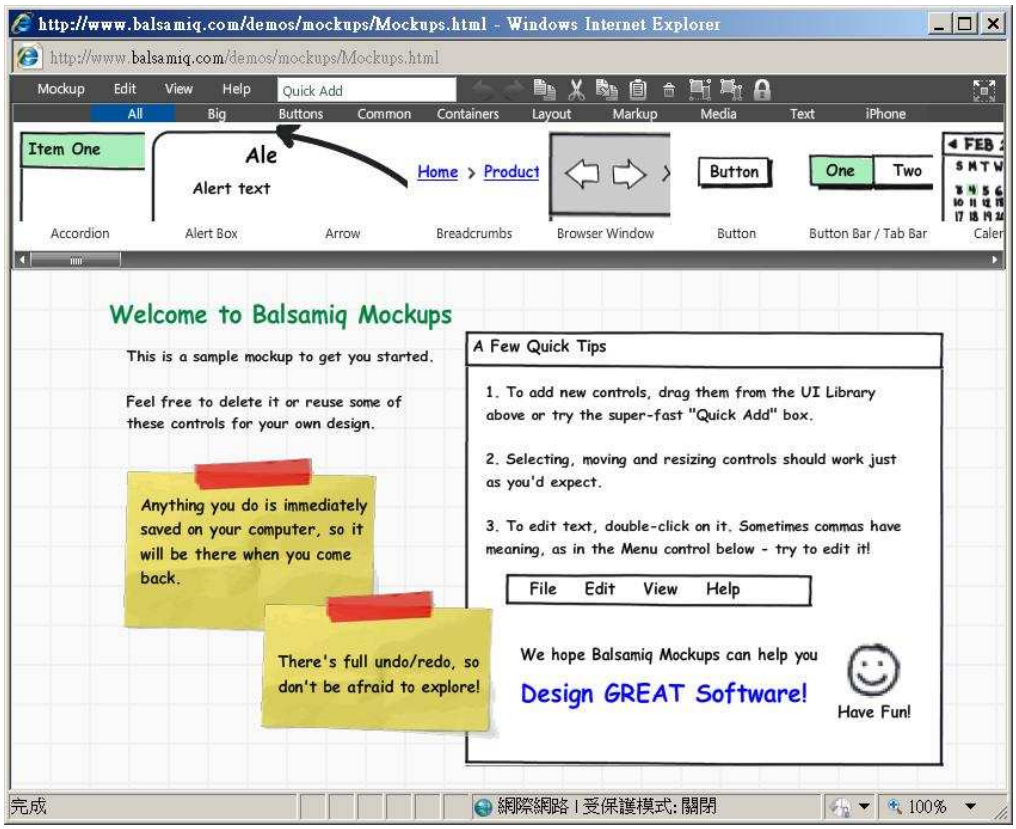

圖 3-40: Balsamiq Mockups 的線上試用

特別的是，Balsamiq Mockups 的行銷策略相當成功，也相當大方，它授權給特定人士一個免費的授權。您可以連到它的官方網站，看到網頁(http://www.balsamiq.com/products/mockups/desktop)中有提到如何取得免費授權，如圖 3-41 所示。

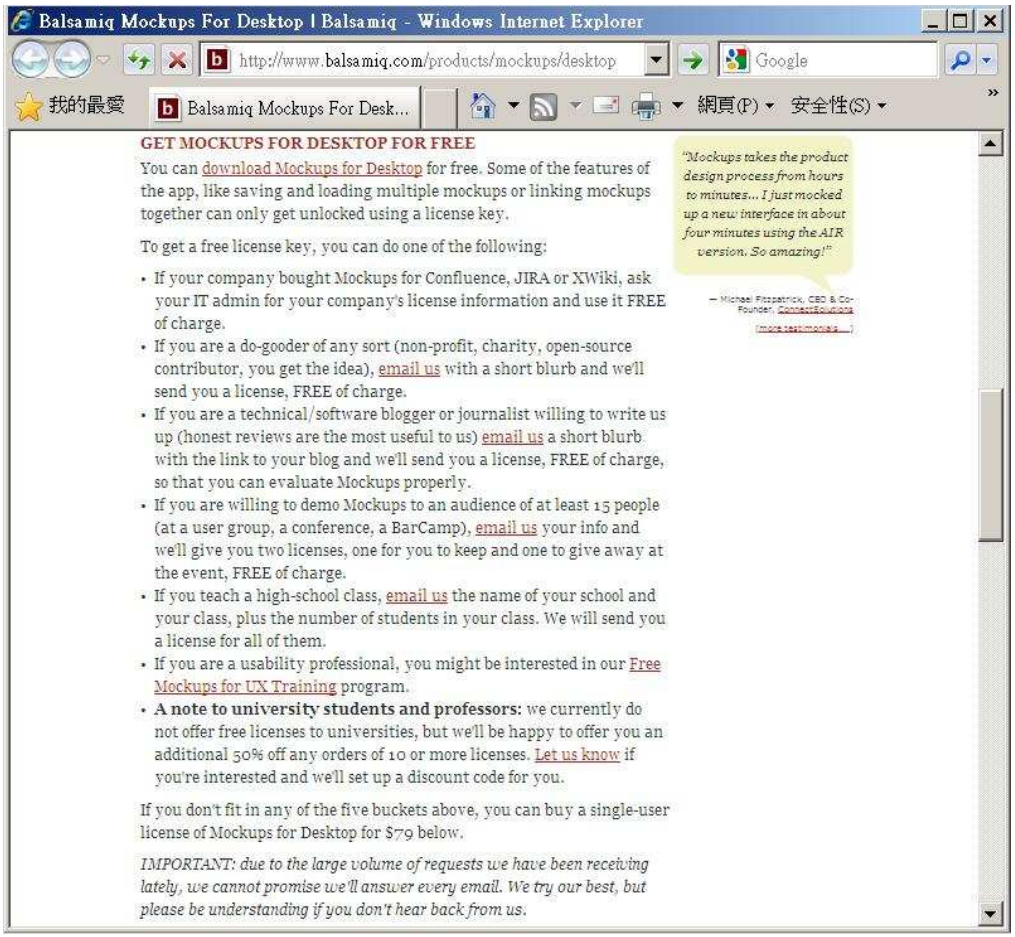

圖 3-41: 免費授權

Balsamiq Mockups 有列出下列七個條件，大抵上，只要您有協助推廣 Balsamiq Mockups，其實都可以取得免費授權。我也有寫信過去 Balsamiq Mockups，跟他們要到一個免費的授權，建議您也寫信去要個免費授權玩玩看，如表 3-8 所示。

Get Mockups For Desktop For Free

You can download Mockups for Desktop for free. Some of the features of the app, like saving and loading multiple mockups or linking mockups together can only get unlocked using a license key.

To get a free license key, you can do one of the following:

- If your company bought Mockups for Confluence, JIRA or XWiki, ask your IT admin for your company's license information and use it FREE of charge.
- If you are a do-gooder of any sort (non-profit, charity, open-source contributor, you get the idea), email us with a short blurb and we'll send you a license, FREE of charge.
- If you are a technical/software blogger or journalist willing to write us up (honest reviews are the most useful to us) email us a short blurb with the link to your blog and we'll send you a license, FREE of charge, so that you can evaluate Mockups properly.
- If you are willing to demo Mockups to an audience of at least 15 people (at a user group, a conference, a BarCamp), email us your info and we'll give you two licenses, one for you to keep and one to give away at the event, FREE of charge.
- If you teach a high-school class, email us the name of your school and your class, plus the number of students in your class. We will send you a license for all of them.
- If you are a usability professional, you might be interested in our Free Mockups for UX Training program.
- A note to university students and professors: we currently do not offer

free licenses to universities, but we'll be happy to offer you an additional 50% off any orders of 10 or more licenses. Let us know if you're interested and we'll set up a discount code for you.

表 3-8: 免費授權

雖然，Balsamiq Mockups 有很多可愛的畫面元件可以選用，不過卻少了一些好用的功能，像是：可以針對畫面元件寫說明、可以貼上特殊標記、可以自動產出文件等等的功能。

3.5.3 Pencil

最後，我們要介紹的 Pencil(http://www.evolus.vn/en-US/Products/Pencil.aspx)，它是一款完全免費的開源軟體。Pencil 有兩個版本，一個是可以單機執行的版本，另外一個是 Firefox 的插件。您要是有安裝 Firefox 的話，可以直接搜尋 Pencil，安裝插件，如圖 3-42 所示。

圖 3-42: Pencil 做成 Firefox 的插件

Pencil 功能簡單，可以讓我們繪製僞畫面，然後存成圖檔。Pencil 的執行畫面如圖 3-43 所示，您不妨下載來玩玩看，不一定要安裝 Firefox，也可以直接下載它的單機版來用。

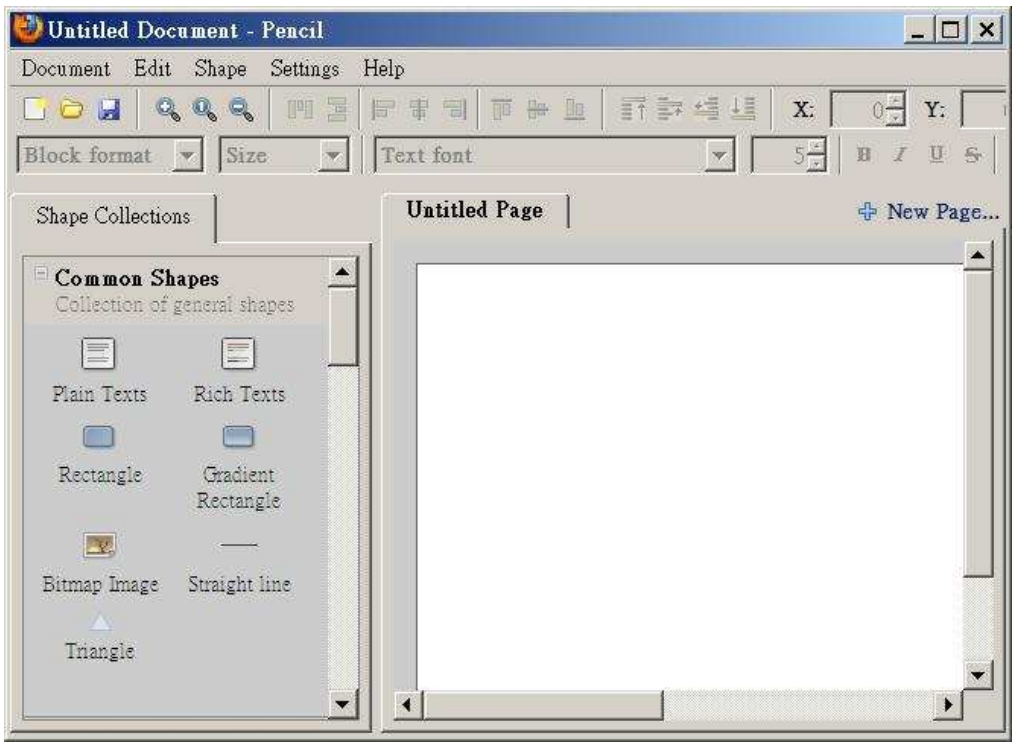

圖 3-43: Pencil 的主畫面

4

(D1)類別圖

※※

4.1 從分析到設計

首先，我們先來簡單歸納一下，我們在分析階段產出的文件，如下：

1. 類別圖—類別圖在描述系統內部的靜態結構，以領域概念爲參考對象。如果，套用 BCE 樣式的話，原先的類別圖會是實體類別圖，而在循序圖產出後，會額外產出邊界類別圖和控制類別圖。
2. 用例圖文—用例圖文在描述系統的外部行爲，也就是在描述參與者與系統互動，以便獲取服務的使用過程。
3. 循序圖—循序圖在描述系統的內部行爲，針對每一個用例，至少會有一張描述主要流程的循序圖。在套用 BCE 樣式之後，循序圖內部的一群物件，將由邊界物件、控制物件和實體物件所組成。換言之，循序圖的一群物件必須來自於類別圖，而物件之間的互動過程，則來自於用例敘述。

分析階段跟設計階段最大的差別在於，分析階段所關注的重點在領域概念、業務流程等等，並未考慮並涉及到實作平台。所以，到了設計階段，不再需要花費太多時間在業務概念上頭，取而代之的是，必須把心力放在實作平台上頭，爲承接分析階段而來類別圖、用例圖文、循序圖加上實作平台或者是開發人員的觀點，產出可以交付給程序員的設計文件。

因此，在本書的開發流程規劃中，我們會讓設計師直接承接分析師的產出文件，進行下述的加工：

1. 類別圖—分析師所產出的類別圖通常跟實作平台有些落差，所以設計師要補上一些實作平台的概念，讓設計出來的類別圖可以真

正交付給程序員實作。

2. 用例圖文—之前我們沒有教分析師，用例之間的包含關係和擴充關係，讓用例圖文保持單純化，以便將焦點聚焦在業務流程上頭。此處，我們會教設計師放進開發人員的觀點，使用包含關係和擴充關係，節錄出可以共用的部分，並且讓用例圖文更爲細緻化。

3. 循序圖—在分析階段的循序圖並沒有太重視訊息上頭的參數，在設計階段，每張循序圖都要拿出來再跑一次，加上所需的參數。由於，有些分析師已經太久沒摸過程式碼了，所以產出的循序圖偏離實作狀況太大，所以需要設計師來補上這一塊，否則程序員是很難直接參考分析文件編寫程式碼的。

好了，接下來，我們要再來多談一些類別圖中的元素，這些元素可能對分析師意義不大，但是對設計師而言，會是非常實用的概念。

4.2 設計師必學元素

4.2.1 依賴關係

之前，我們學到了類別之間的結合關係、或組合關係，它們都是一種需要長期保存在資料庫中的靜態關係。相較之下，「依賴關係」(dependency relationship)是一種暫時的、動態的關係，它不需要被長期保存，可以在使用的瞬間建立，不用了就回收。

因此，當兩物件之間可以互傳訊息時，意味著兩物件之間存在有，需要長期保管的靜態關係，或者是暫時性的動態關係。比方說，在圖 4-1

中，邊界物件與實體物件之間可以透過動態的依賴關係互動，用完就丟，不需要將這個動態關係保存在資料庫中。而房型和景觀圖片兩者之間由於存有組合關係，所以它們可以藉由靜態關係互動。

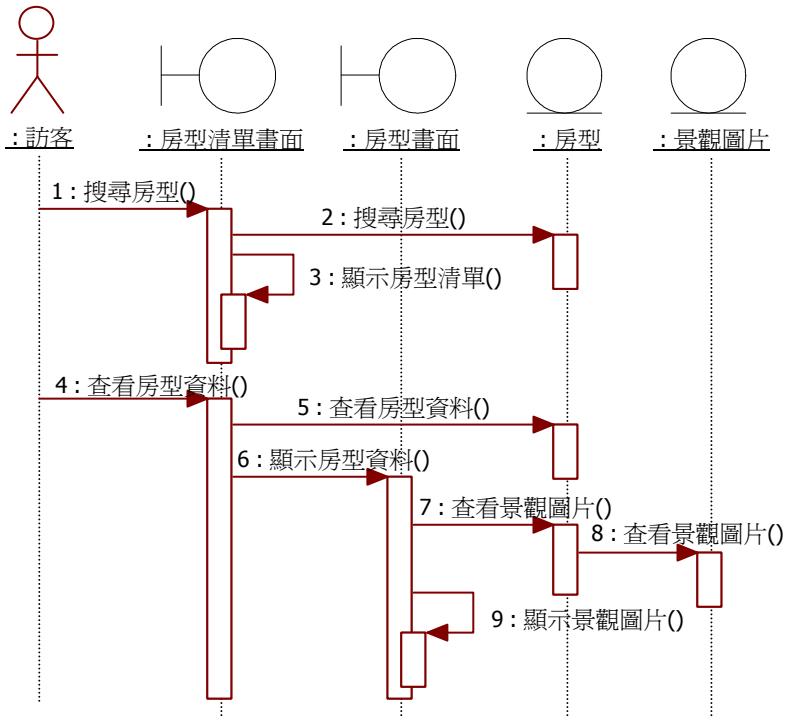

圖 4-1: 「查詢民宿資料」用例的循序圖

所以，回過頭來看，我們可以在類別圖中，同時繪製出動態的依賴關係，以及靜態的結合關係或組合關係，如圖 4-2 所示。依賴關係的圖示是帶箭頭虛線，由扮演依賴角色的「客戶端」(client)指向扮演被依賴角色的「支援端」(supplier)。

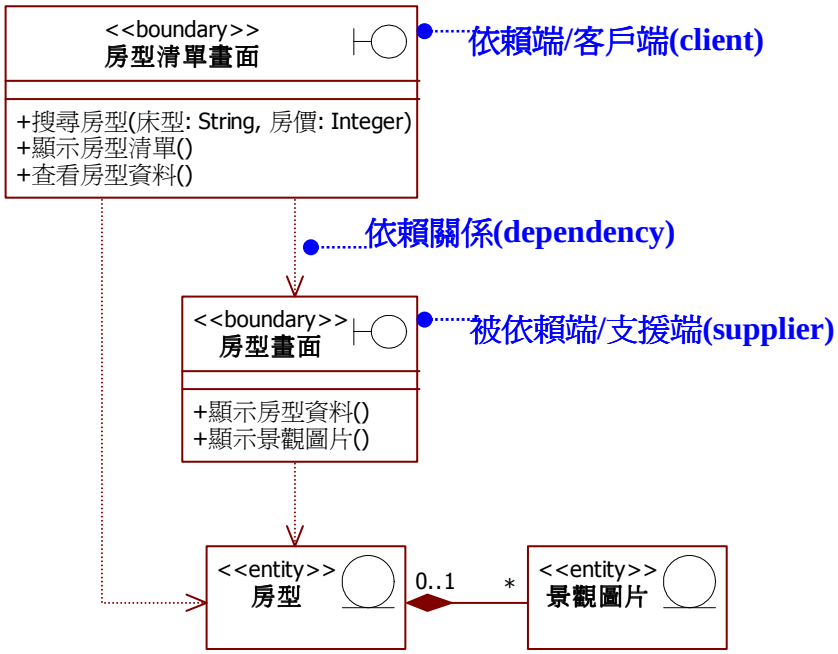

圖 4-2: 依賴關係

特別注意到，既然說是「依賴」則意味著連動性，支援端只要一變動，很可能客戶端會受到連動。所以，在建構依賴關係時，被依賴的支援端愈穩定愈好，像是在 BCE 樣式的概念中，您會發現邊界物件、控制物件比較不穩定，所以我們不讓穩定的實體物件依賴它們，避免因此而導致整個結構的不穩定。

總之，如果兩物件之間已經存有靜態關係時，可以優先使用靜態關係互動，而且記得要將靜態關係保存在資料庫中。如果兩物件之間沒有靜態關係，也可以建立暫時性的依賴關係，以便進行互動，而且用完即丟，不

需要費心保存在資料庫中。

4.2.2 一般化關係

「一般化關係」(generalization relationship)是一種分類關係；針對同一種概念的事物，區分成：一般性的(general)和特定性的(specific)，然後再建構起兩類別之間的一般性關係。

比方說，在訂房系統中，我們可以將景觀圖片分爲兩大類：民宿圖片和房型圖片。相較於原先的景觀圖片，民宿圖片和房型圖片兩者都屬於較爲特定的圖片，因此我們可以透過一般化關係建構出三者的關係，如圖 4-3 所示。

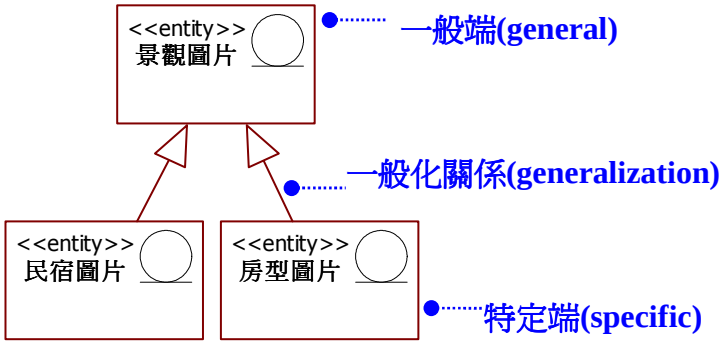

圖 4-3: 一般化關係

實務上，我們常稱一般端的類別爲「父類別」(superclass)，特定端的類別爲「子類別」(subclass)。而且，在一般化的圖示上，也可以多個子類別共用大三角形端點，如圖 4-4 所示。

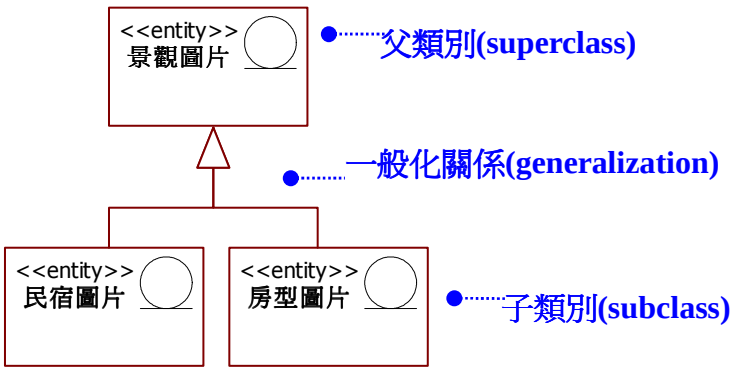

圖 4-4: 父類別與子類別

在一般化關係中，有個很重要的特色是，子物件可以替代父物件；這是因爲子類別繼承了父類別的特徵，具體來說，子類別可以透過一般化關係繼承父類別的屬性、操作與靜態關係。還記得，前面我們說過靜態關係是指結合關係，而組合關係則是結合關係的一種。

所以，請看到圖 4-5 的例子，雖然民宿圖片和房型圖片並未宣告任何屬性、操作和關係，但其實它們已經擁有景觀圖片已經定義好的屬性、操作和關係了，這就好比小孩繼承了父母留下來的財產一樣。

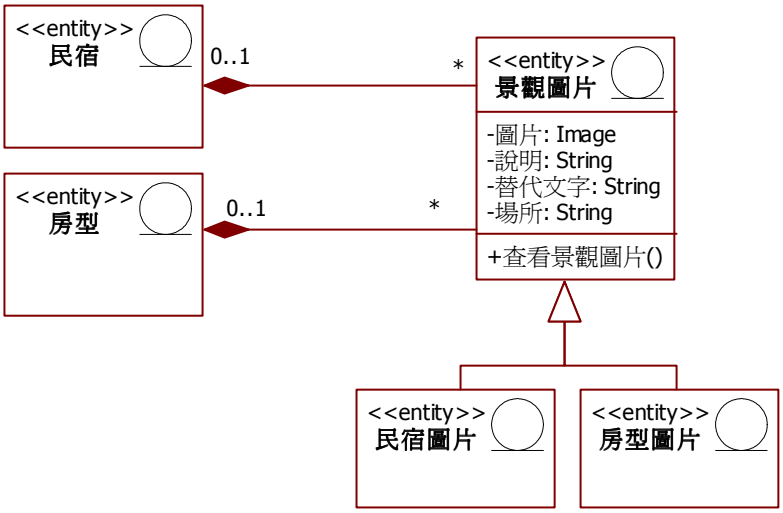

圖 4-5: 繼承

不過，在圖 4-5 的例子中，如果我們不想繼承關係，可以改成圖 4-6 的設計，特別注意原先圖 4-5 中 0..1 的個體數目，到了圖 4-6 中則改成了 1，這樣才是正確的個體數目。

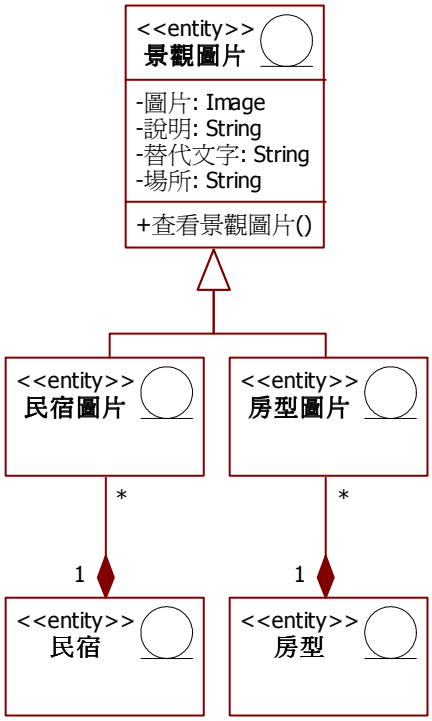

圖 4-6: 個體數目 0..1 改成 1

再者，繼承而來的屬性、操作和關係，也可以在子類別處「重新定義」(redefine)，不一定要全盤接受。譬如，在景觀圖片中的「場所」屬性並沒有預設值，繼承之後，我們在它的子類別處加上了預設值，重新定義了場所，如圖 4-7 所示。

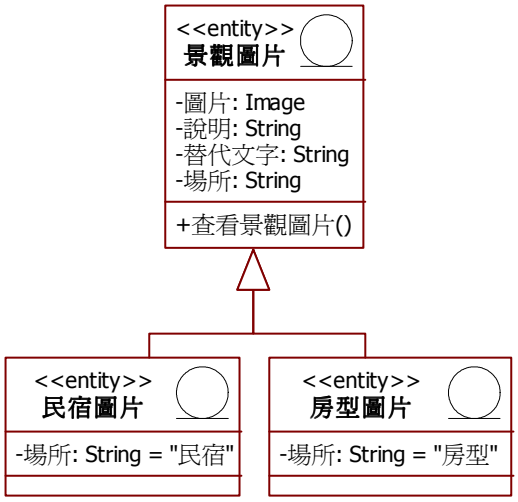

圖 4-7: 重新定義「場所」

當然，子類別除了繼承父類別外，也可以額外增加屬於自己專用的屬性、操作或關係。假設，在民宿圖片類別中，我們需要多記錄一項圖片的尺寸，但是在房型圖片類別中不需要這項屬性，如圖 4-8 所示。

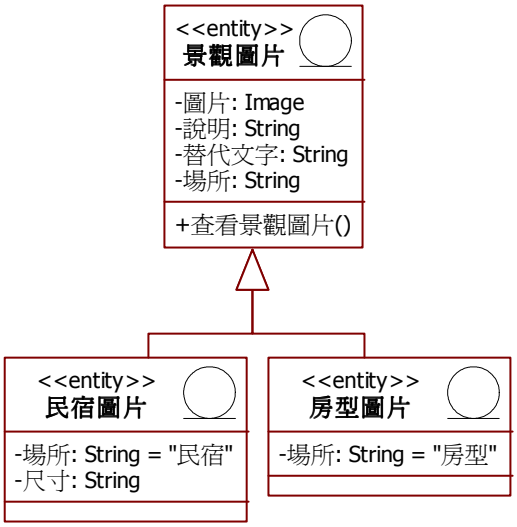

圖 4-8: 子類別增加專有的屬性

4.2.3 保護等級

雖然，透過一般化關係，子類別可以繼承父類別已經定義好的屬性、操作和關係。不過，要是父類別將這些成員宣告成私有等級(private)的話，子類別就無法繼承來使用了。特別是屬性，一般爲了封裝性，都會建議設置成私有等級能見度。

在這種情況下，UML 設置了另一種「保護等級」(protected)的能見度，能見程度介於私有等級與公開等級之間，專門配合一般化關係使用。在使用上，父類別可以將私有等級的成員改成保護等級，這樣一來，子類別便能夠直接使用保護等級的成員。請看到圖 4-9，保護等級的符號是井字號(#)。

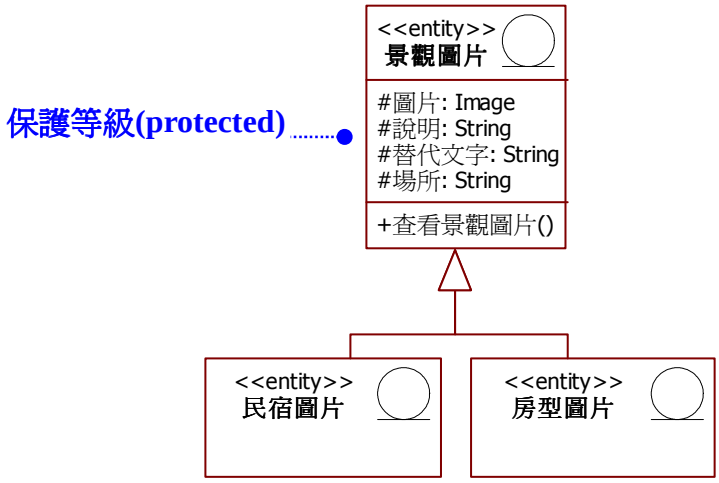

圖 4-9: 保護等級的能見度

4.2.4 抽象類別

在一般化架構中，有時候，父類別本身並不完整，所以無法誕生物件，這種無法產子的類別稱爲「抽象類別」(abstract class)。抽象類別同樣使用矩形圖示，只不過它的類別名稱採用斜體字，如圖 4-10 所示。

圖 4-10: 抽象類別

通常，我們之所以會將類別設置成抽象類別，主要是因爲類別中含有

不具實作內容或者不完整的「抽象操作」(abstract operation)，所以才會將它設置成抽象類別。請看到圖 4-11，抽象操作的表示法如同一般的操作，只不過操作名稱使用斜體字，就跟抽象類別的表示法相似。

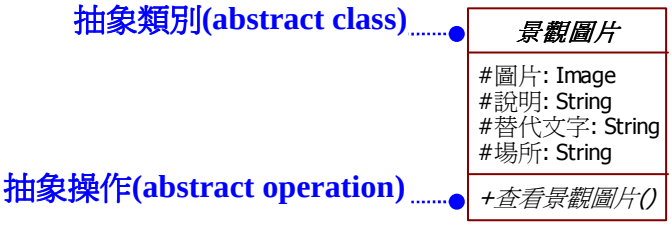

圖 4-11: 抽象操作

一旦，抽象類別配合一般化關係後，子類別可以針對抽象操作提供完整的實作，這樣一來，不僅有子物件可用，也可以誕生子物件來替代父物件，像在圖 4-12 中，民宿圖片和房型圖片都提供了具體的「查看景觀圖片」操作。

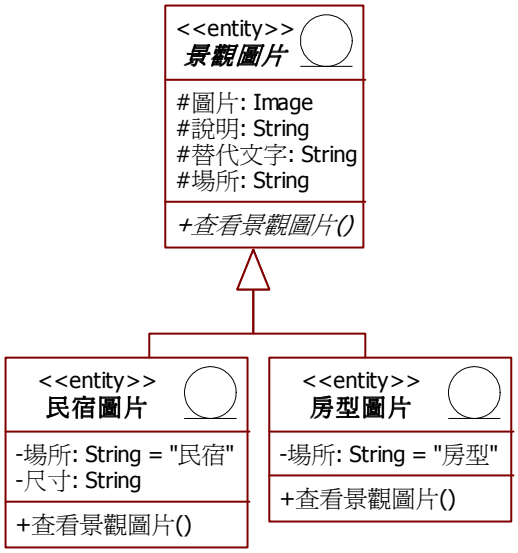

圖 4-12: 子類別提供了具體操作

4.2.5 類別層級

之前，我們所看到的屬性和操作都屬於「物件層級」(object-scoped, instance-scoped)的，代表該類別所誕生的每一個物件都擁有一份物件層級的屬性，而且外界也只能透過物件調用物件層級的操作。

相對的，有另一種稱爲「類別層級」(class-scoped, classifier-scoped)的屬性與操作，代表整個類別共用一份屬性，而且外界只要透過類別就可以調用操作，不需要先誕生物件。

譬如，在訂房系統中，會員類別內部的「驗證」操作，要是設爲類別層級的操作，或許會比設爲物件層級的操作，更爲恰當。因爲，當我們請會員進行驗證時，其實根本就還沒正確找出某一個會員物件，所以可能不是請某一個會員物件來進行驗證，而是應該找會員類別先進行驗證，通過

驗證之後，才找出正確的會員物件才對。

所以，請您看到圖 4-13，類別層級的屬性與操作名稱有底線，之前我們看到沒有底線的屬性或操作都屬於物件層級的。還有，應該由所有的會員物件共同維護一份會員總數量即可，所以這個屬性也適合設成類別層級的屬性。

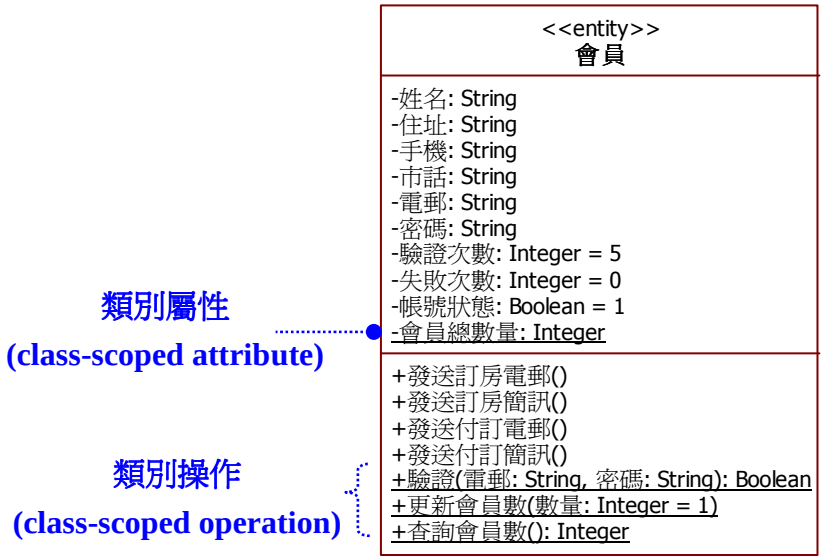

圖 4-13: 類別層級的屬性與操作

此外，其他像是誕生物件、刪除物件的操作，其實都適合設成類別層級，因爲在物件誕生之前，該物件根本不存在，怎麼請這個物件執行誕生自己的動作呢！至於，刪除物件也同樣交給所屬類別來執行，或許會比請物件自殺，要合適多了。

4.2.6 公用類別

除了抽象類別因爲不完整，所以不能誕生物件外，此處的「公用類別」(utility class)則是因爲它所擁有的屬性、操作都是「類別層級」的，所以不需要誕生物件，就可以直接使用公用類別所提供的屬性和操作。

顧名思義，公用類別存在的目的，就是像工具一樣，給所有類別、物件使用的。雖然，公用類別不誕生物件，但是它是完整的，跟抽象類別不同。所以，公用類別的圖示是一般的矩形，名稱底下沒有底線，不過類別名稱上頭會標示<<utility>>，用來區辨它是個公用類別。

再者，公用類別裡頭所有的屬性和操作，都必須是類別層級的，不能設置物件層級的屬性及操作。請您看到圖 4-14，我們爲訂房系統設計一個「自動產碼」的公用類別，讓其他類別、物件都可以使用它來自動產生序號。

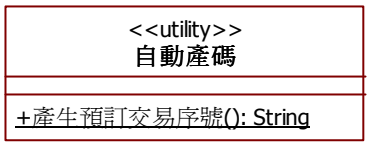

圖 4-14: 公用類別

4.2.7 列舉型別

前面，我們都在談類別或物件，最後我們來看個特殊的資料型別(data type)概念—「列舉型別」(enumeration)。列舉型別也是採用矩形圖示，不過名稱上頭多了<<enumeration>>的字眼，而且除了屬性和操作外，矩形

內部最底部多了一格放置「列舉值」(enumeration literal)，如圖 4-15 所示。

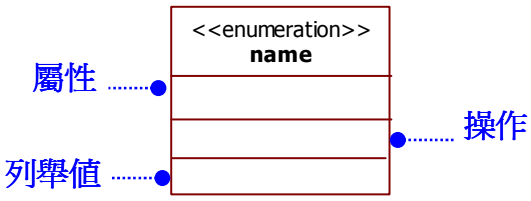

圖 4-15: 列舉型別

不過，由於我們不太爲列舉型別設置屬性和操作，所以大部分時候，列舉型別的屬性格和操作格都會被隱藏起來，僅出現列舉型別名稱和列舉值。請您看到圖 4-16，我們設計了一個名爲「床型種類」的列舉型別，然後把床型的資料型別設爲床型種類，限制床型只能是其中一種列舉值。

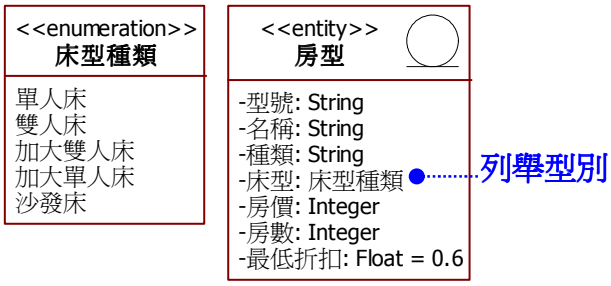

圖 4-16: 「床型種類」列舉型別

4.3 從物件導向到關聯式資料庫

在 UML 的世界中，系統以物件的方式在運作，當然也包含以物件的方式儲存在資料庫中。可是，實作的世界中，並不總是如此完美，雖然物件導向資料庫(Object-Oriented Database,OODB)已經發展多年，但關聯式資料庫卻是目前的主流技術。

所以，從物件導向到關聯式資料庫，一直是個鴻溝，卻也是許多年來微軟等大廠一直努力投入的主題。所幸，近年來，關於「物件關聯映射」(Object-Relational Mapping, ORM)技術有十足的進展，像是 Java 陣營發展出來的 Hibernate、或是微軟自家發展出來的「實體框架」(Entity Framework)，在在填補了物件導向與關聯式資料庫之間的鴻溝。因此，很幸運地，我們可以比過去的前輩們，更專心在物件導向技術中，無須太過擔心資料庫端的轉換。

本書假設後端採用關聯式資料庫，而且我們也不願意花時間繪製實體關聯圖(ERD)的情況下，設計師可以選擇在類別中加入關聯式資料庫的「主鍵」(Primary Key, PK)和「外鍵」(Foreign Key, FK)的概念，讓實體類別圖可以一圖兩用，同時落實到程式碼和關聯式資料庫。

特別注意到，在物件導向的世界中，物件之間透過連結就可以連到對方，因此類別之間有靜態關係，也就不需要額外加入鍵值，特別是外鍵的概念。所以，請您看到圖 4-17 中，每一個類別中的屬性都是自己的屬性，並沒有抓別的類別的屬性過來當外鍵。

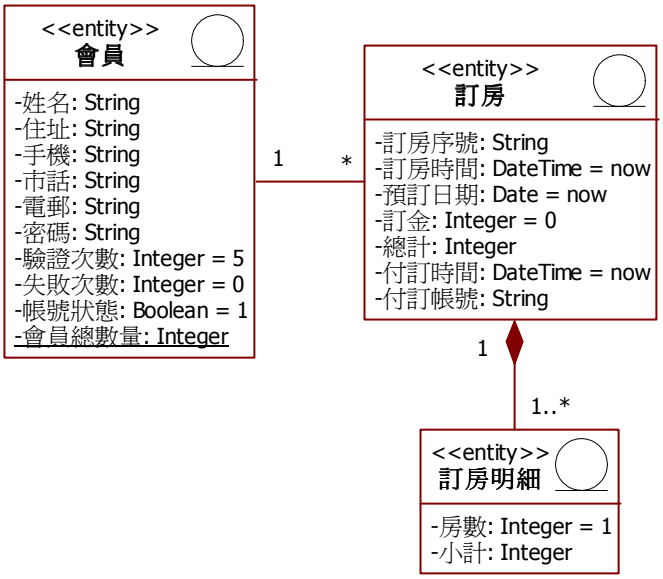

圖 4-17: 靜態關係

但是，在關聯式資料庫中，資料表中必須設置鍵值，所以設計師得在類別中設置主鍵，並且增加做爲外鍵的屬性。請看到圖 4-18，我們可以在屬性前面加上<<PK>>代表主鍵，以及加上<<FK>>代表外鍵。原本在訂房類別中，並沒有「電郵」這個屬性，現在把會員的電郵抓過來當外鍵。同樣的，本來在訂房明細中也沒有「訂房序號」，現在也得把訂房類別中的訂房序號抓過來當外鍵。

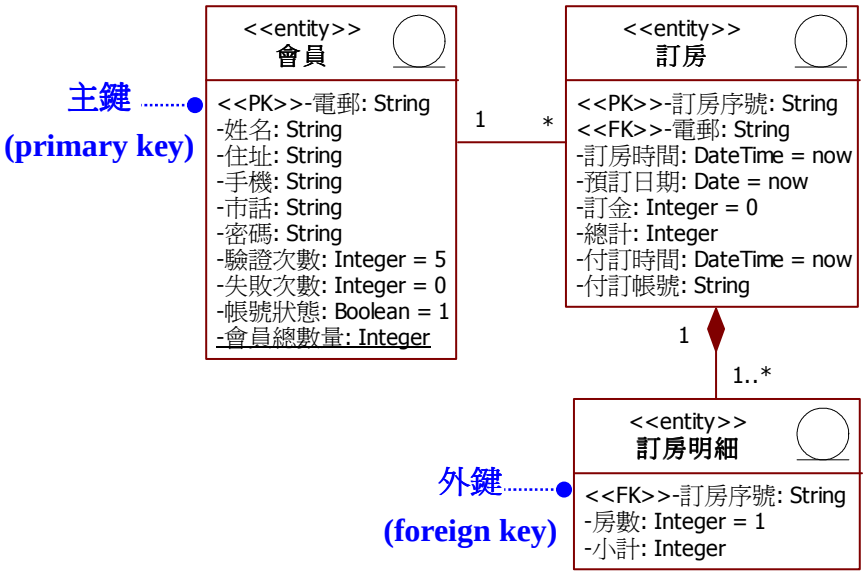

圖 4-18: 主鍵與外鍵

4.4 民宿聯合訂房系統

既然，在分析階段都已經產出一份份的用例了，接下來在設計階段，設計師還是以用例爲一個個的工作單元，檢驗每一個用例中所涉及的 BCE 類別，爲這些類別進行加工。

重點加工的部分有下述：

1. 把中文的類別名稱、屬性名稱、操作名稱都變成英文。如果您在分析階段就已經採用英文的話，可以先在不確定的英文名稱前頭加上雙斜線(//)，在設計階段確認之後，才把雙斜線去掉。特別是操作名稱，通常在分析階段都比較不確定。當然，如果在分析

階段就已經非常確認的話，也可以不加雙斜線，直接就用英文名稱。

2. 針對每一個用例，建立 BCE 類別圖，並且加上 BCE 類別之間的依賴關係。

3. 針對前述提到的依賴關係、一般化關係、保護等級能見度、抽象類別、類別層級屬性與操作、公用類別和列舉型別，都要找一找修正一下實體類別圖。

4. 針對更新後的實體類別圖，加上適當的主鍵和外鍵。

至於，操作中的輸入參數和回覆參數，我們到了設計循序圖的時候，才來檢驗並加工。還有，邊界類別和控制類別，也是稍後再來加工，此處設計師可以先專注在重要的實體類別圖上頭。

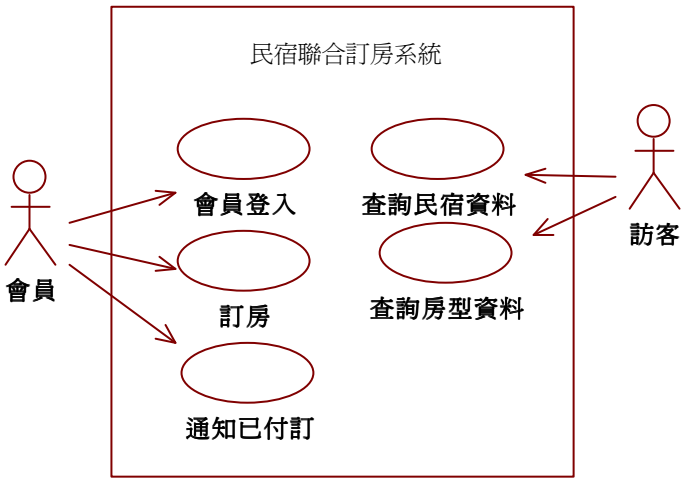

圖 4-19: 有分析的用例

目前為止，我們已經分析過的用例，一共有：會員登入、查詢民宿資料、查詢房型資料、通知已付訂、訂房，如圖 4-19 所示，接下來我們就逐個來加工吧！

4.4.1 用例—會員登入

在會員登入用例中，主要用到「會員」實體類別，如圖 4-20 所示。

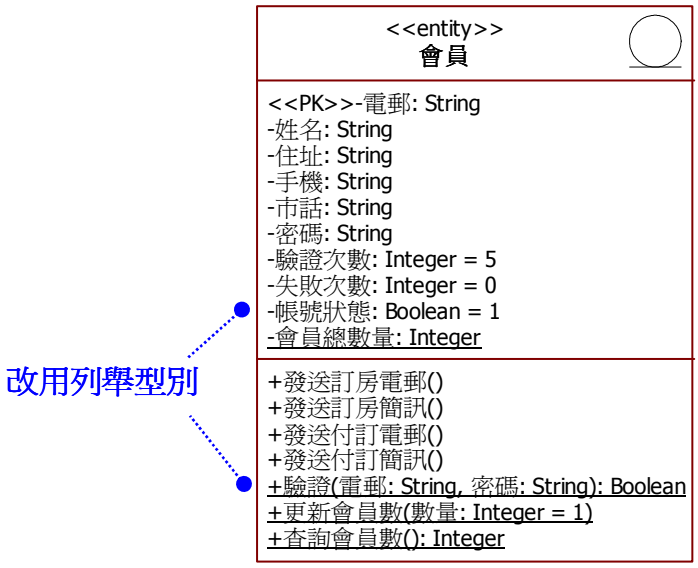

圖 4-20: 會員

除了把會員的類別名稱、屬性、操作全改成英文名稱外，還把原先

「帳號狀態」屬性和「驗證」操作的回覆參數的資料型別改成列舉型別，如圖 4-21 所示。

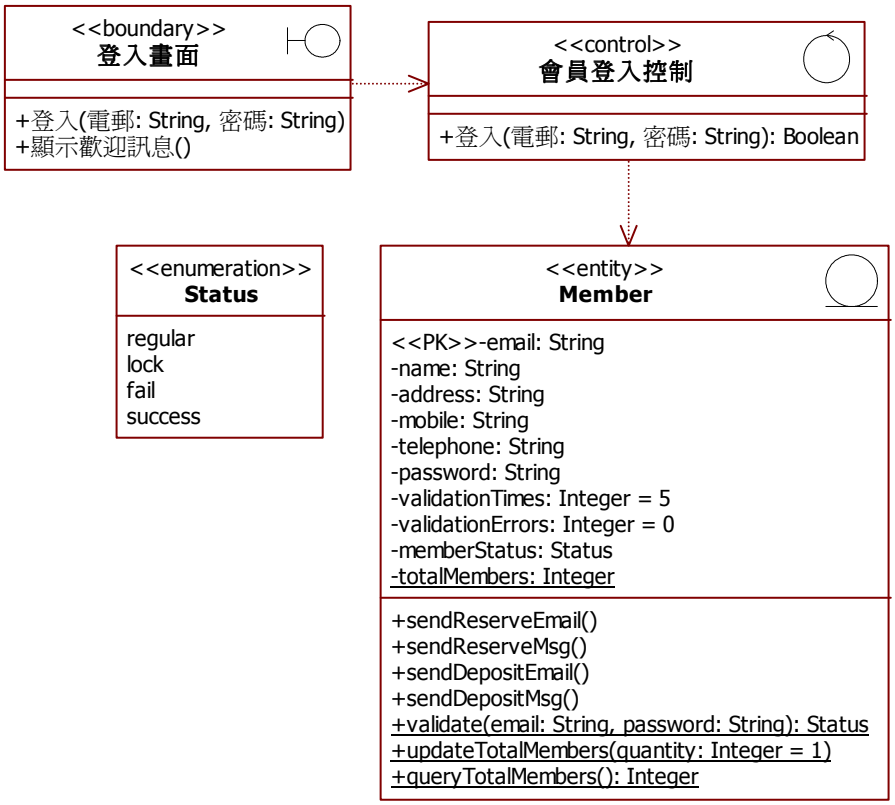

圖 4-21: 「會員登入」的 BCE 類別圖

至於，主鍵的部份，之前舉例時已經加工過了，這邊就不再說明了；外鍵的部份，等出現跟其他類別之間的靜態關係，再來修改。

4.4.2 用例—查詢民宿資料

「查詢民宿資料」用例很簡單，連控制物件都沒設置，僅做資料庫查詢的工作，它的 BCE 類別依賴關係如圖 4-22 所示。

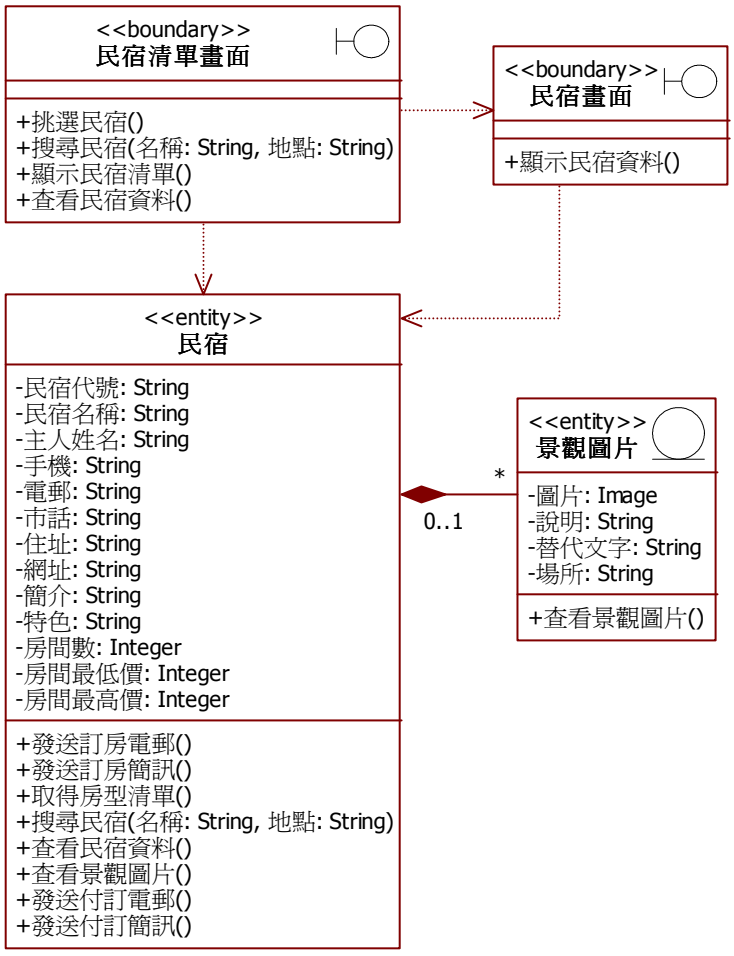

圖 4-22: 「查詢民宿資料」的 BCE 類別圖

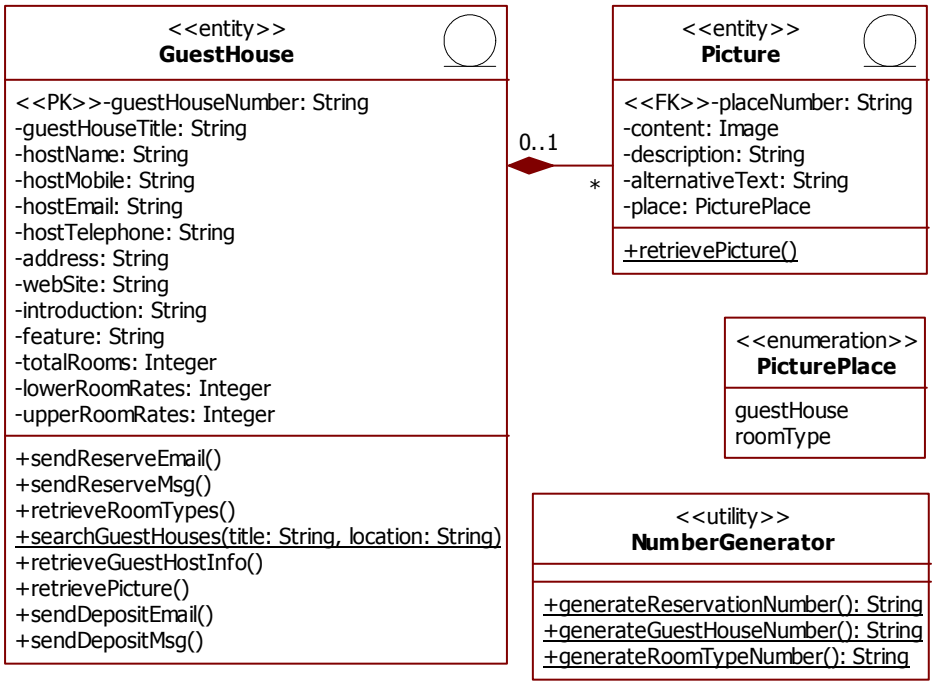

圖 4-23: 「查詢民宿資料」的實體類別

在圖 4-23 中，有幾個重點，如下：

1. 在景觀圖片的部分，我們並沒有使用前面談到的一般化架構，而是讓民宿和房型都連到這個景觀圖片。所以，景觀圖片的外鍵會有兩個來源一個是來自民宿，另一個來自房型。因此，另外取了「場所代號」(placeNumber)做爲景觀圖片的外鍵名稱。
2. 再者，我們爲景觀圖片的「場所」(place)屬性，設計了一個 PicturePlace 列舉型別，其列舉值有民宿和房型兩個。

3. 至於，民宿代號也一併透過「自動產碼」(NumberGenerator)公用類別自動產出。

4.4.3 用例—查詢房型資料

請先看到圖 4-24 關於「查詢房型資料」的 BCE 類別圖，其中的「景觀圖片」(Picture)實體類別前面已經加工過了，所以這邊就僅秀出實體類別的圖示，把它的屬性和操作隱藏起來了。

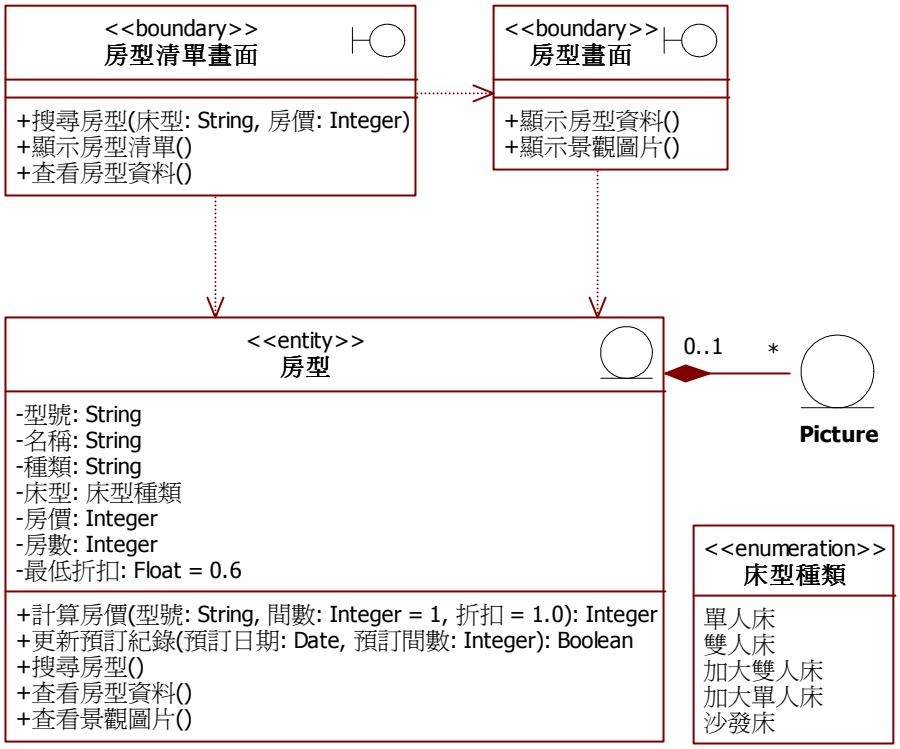

圖 4-24: 「查詢房型資料」的 BCE 類別圖

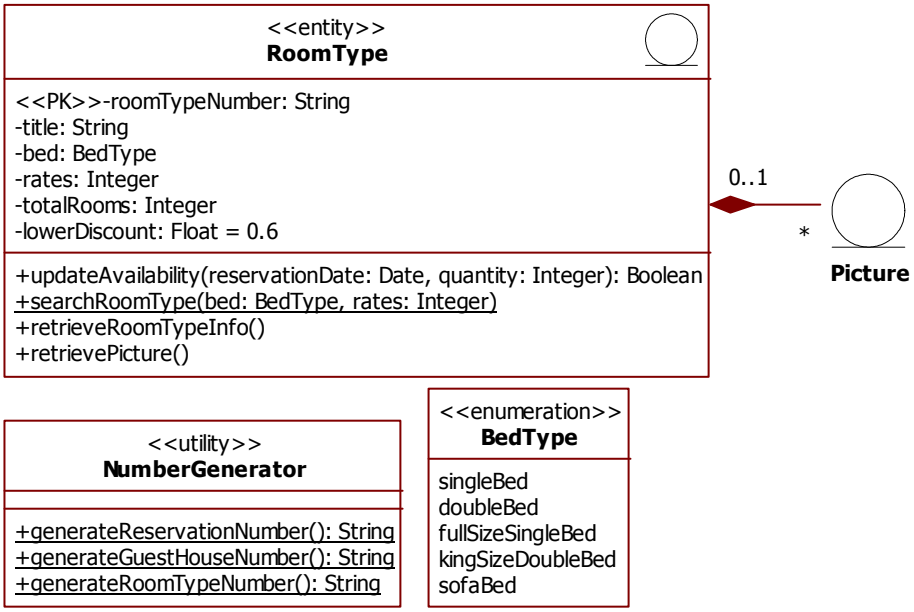

圖 4-25: 「查詢房型資料」的實體類別

接著，請看到加工後的圖 4-25，幾項重點如下：

1. 房型類別中的「計算房價」操作，一直都還沒用到，所以我們就先把它刪掉了，日後有需要再補上。
2. 由於，訂房系統中可能會用到房間(Room)類別，所以這邊的「房型」就翻譯成「RoomType」了。
3. 還有，房型類別已經有「床型」(bed)屬性了，所以似乎不再需要用到「種類」屬性，所以我們也把這個屬性刪了。

4. 再者，房型名稱是指民宿主人幫房型取的名稱，像是尊貴總統房、紫色浪漫屋之類的房型暱稱。

5. 自動產碼(NumberGenerator)公用類別增加了一個「產生房型代號」(generateRoomTypeNumber)的操作，所以我們又將這個類別更新了一次。

4.4.4 用例—通知已付訂

接著，再來看「通知已付訂」用例的 BCE 類別，只剩訂房類別還沒加工過，如圖 4-26 所示。

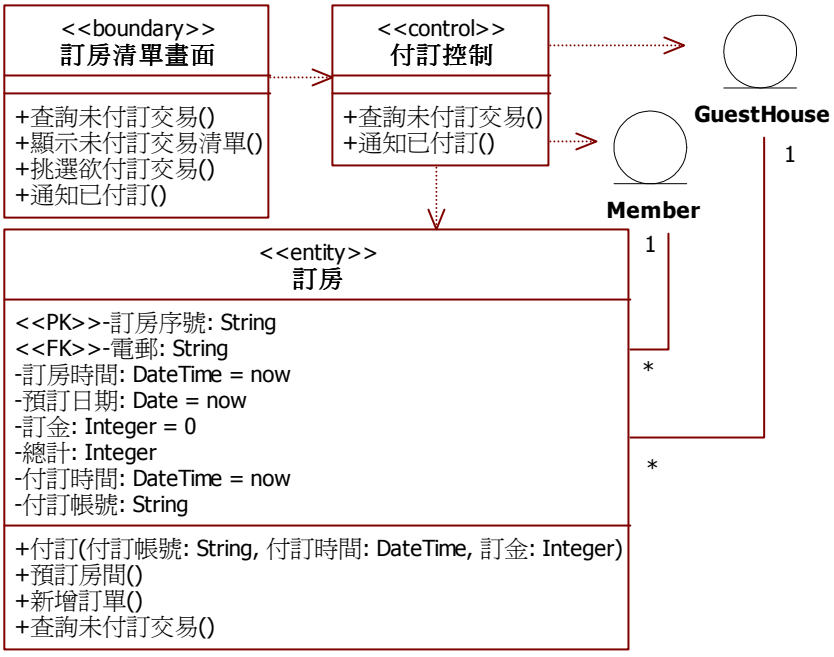

圖 4-26: 「通知已付訂」的 BCE 類別圖

訂房除了還要加設一個「民宿代號」(guestHouseNumber)屬性做爲外鍵，其餘就沒什麼特別需要注意之處了，加工結果如圖 4-27 所示。

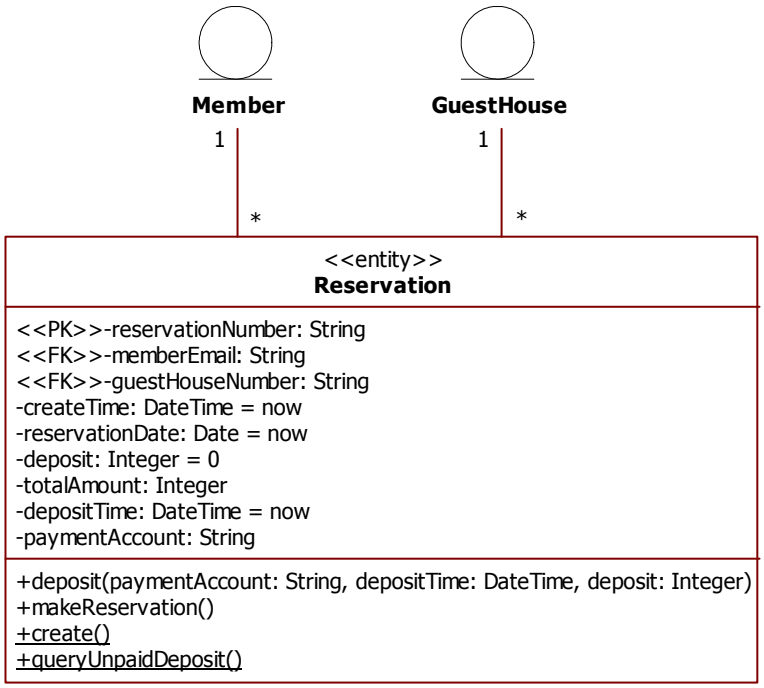

圖 4-27: 「通知已付訂」的實體類別

4.4.5 用例—訂房

「訂房」用例挺複雜，涉及到多個物件，所以我們先把它的 BCE 類別之間的依賴關係繪製出來，如圖 4-28 所示。

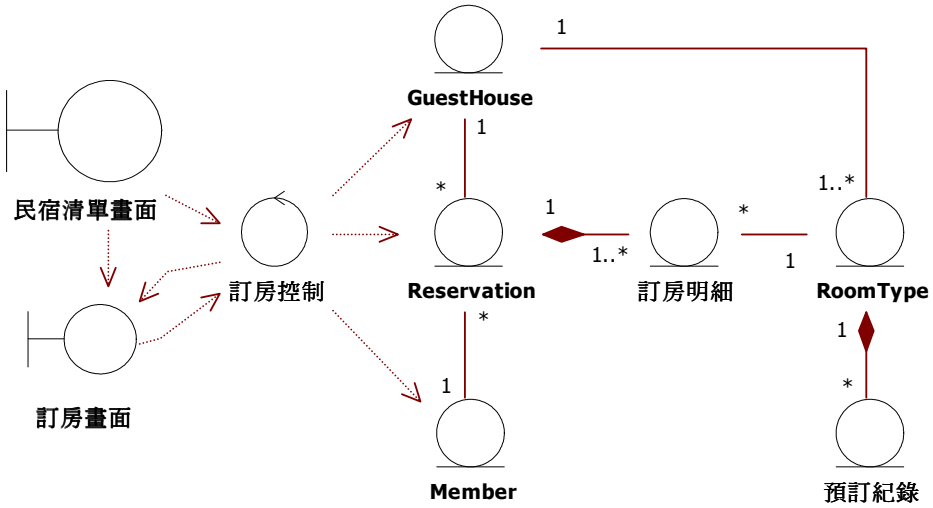

圖 4-28: 「訂房」的 BCE 類別圖

不過，我們特別把訂房用例留在最後處理，所以好幾個實體類別都已經加工過了，只剩下「訂房明細」和「預訂紀錄」實體類別需要加工，如圖 4-29 所示。

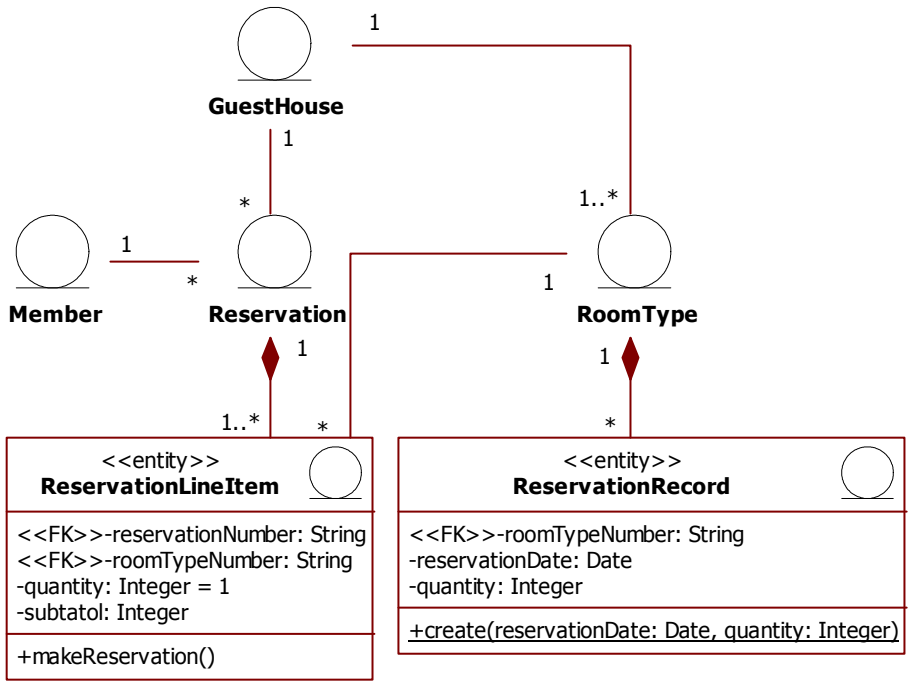

圖 4-29: 「訂房」的實體類別

4.4.6 類別圖

還記得，我們在第二章的最後，彙總了一張類別圖，我把它重複貼到圖 4-30 中。在後續發展的用例中，都暫時沒用到「入住」和「房間」這兩個實體類別，所以我想先刪掉它們，讓類別圖單純些，如果我們分析新的用例有使用到新的實體類別，再來擴充實體類別圖好了。

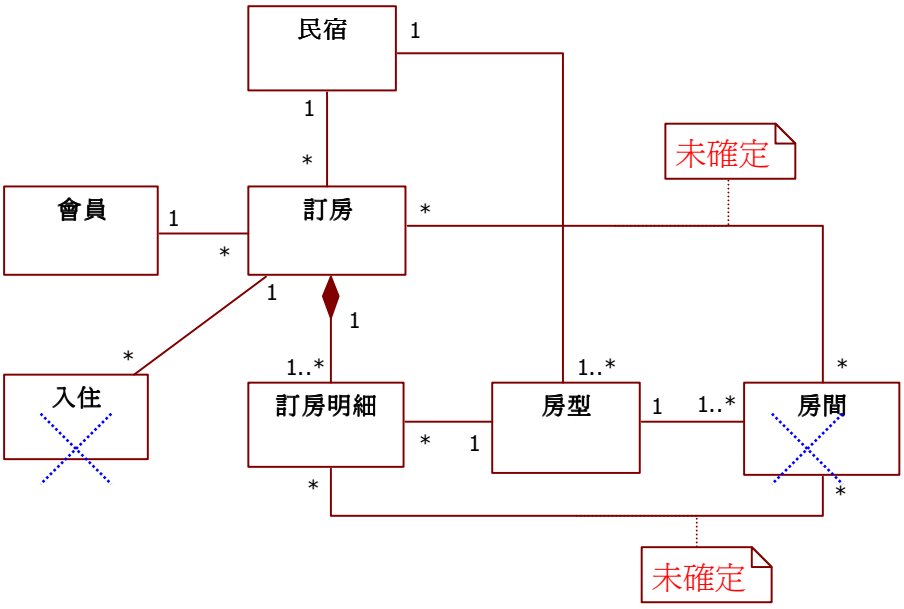

圖 4-30: 類別圖

然後呢，很久沒出現的齒輪圖，我還是要再囉嗦一次，如圖 4-31 所示。前面刪去入住和房間實體類別，是個活生生、血淋淋的驗證，還好當時我們並未在類別圖花過多的時間，並沒有選擇單角度的想要極度探究類別圖，很快的進入下一個用例圖文、循序圖的階段，所以白工才花的少，否則現在要刪掉的可能更多呢！

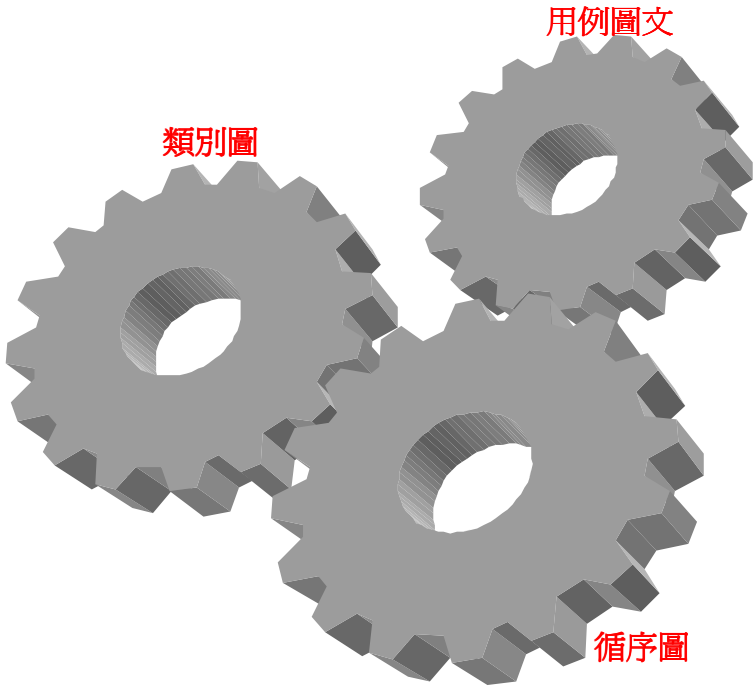

圖 4-31: 不要只動其中一塊

所以，最後我們再巡覽一次實體類別圖，整體修訂一下個體數目，而且發現房型(RoomType)遺漏了一個外鍵，所以增加一個名爲「民宿代號」(guestHouseNumber)屬性做爲外鍵，就沒再做其他的變動了，如圖 4-32~37 所示。

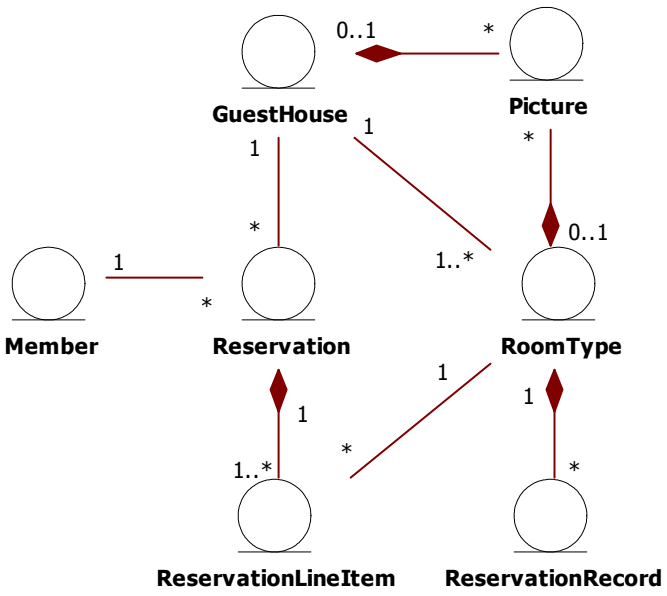

圖 4-32: 實體類別圖

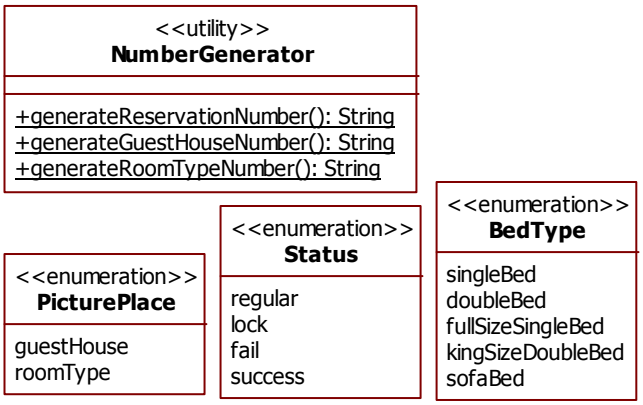

圖 4-33: 公用類別與列舉型別

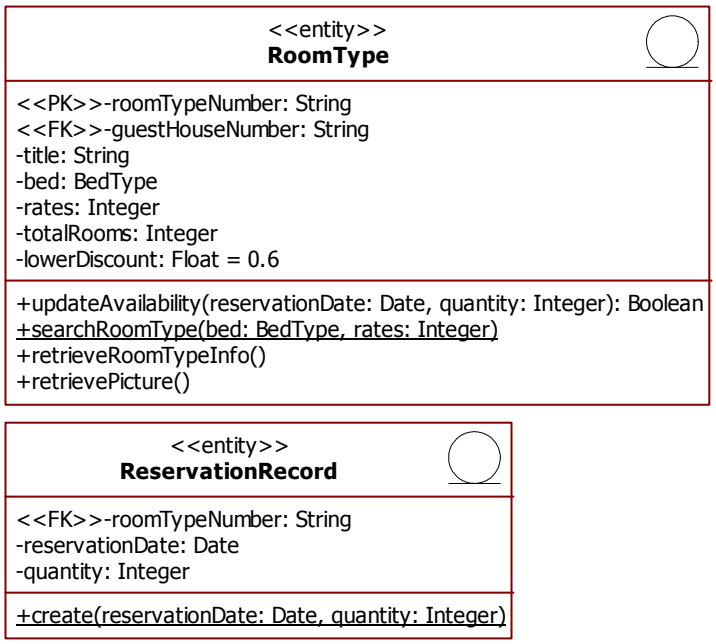

圖 4-34: RoomType 與 ReservationRecord 實體類別

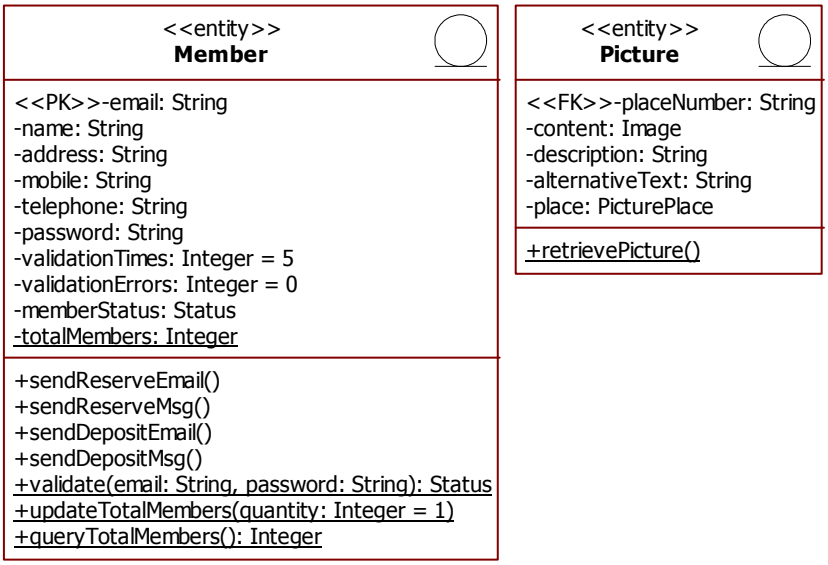

圖 4-35: Member 與 Picture 實體類別

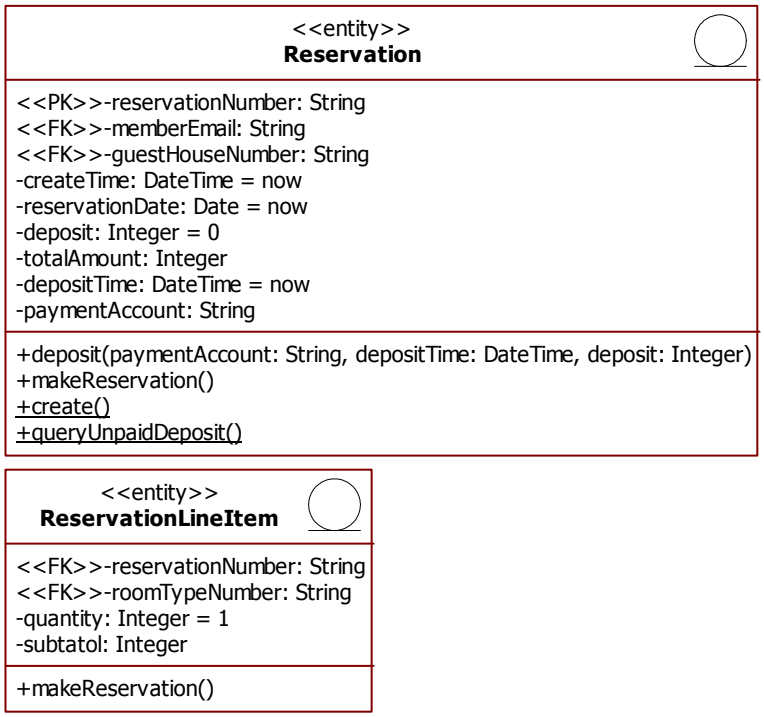

圖 4-36: Reservation 與 ReservationLineItem 實體類別

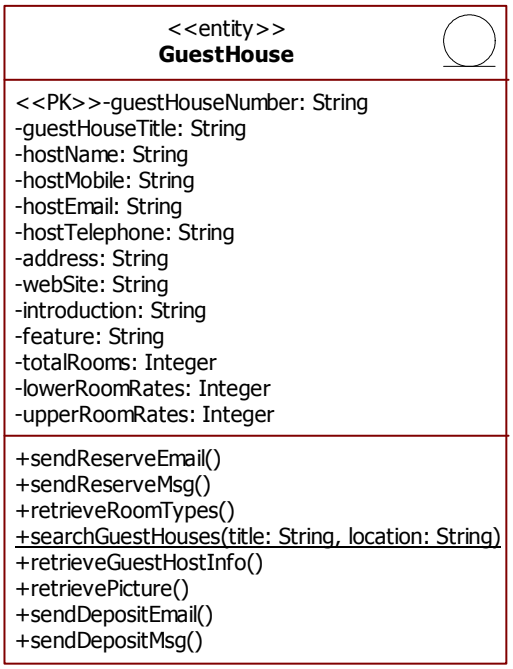

圖 4-37: GuestHouse 實體類別

5
(D2)用例圖文

※※

5.1 使用者觀點與開發人員觀點

用例跨在使用者和分析師之間，同時我們又期望分析師產出的用例，也可以跨在分析師與設計師之間，甚至也可以成爲設計師與程序員之間的溝通橋樑。

所以，分析師在產出用例的過程中，只好少放一些開發人員的觀點，這樣產出的用例才能夠與使用者溝通。不過，一旦等到用例進入設計階段後，設計師便得加工放入更多開發人員觀點，讓這份用例設計可以同時有助於程序員後續的編碼工作。

回顧前面分析師學到的用例概念中，著重在定義出大小適度的用例，找出用例的參與者，描述參與者與系統之間的互動過程。到此爲止，可以說都未涉入開發人員的觀點；唯一有涉入開發人員觀點的部份是，注意用例除了一般的啓動者外，是否有其他連線的、扮演支援角色的系統參與者。

總之，設計師現在需要加入更多的開發人員觀點，對用例圖文做一些加工，當然也同時趁這個機會調整一下用例圖文，讓它可以更貼近「真實」一些；分析師產出的用例圖文最令人詬病之處，就是太抽象、難以落實，這個部份必須靠設計師來拉近現實。

接下來，我們會談到一般化關係(generalization relationship)、包含關係(include relationship)和擴充關係(extend relationship)，教設計師如何在用例圖文中加入更多開發人員的觀點。Let's Go！

5.2 設計師必學元素

5.2.1 一般化關係

還記得在上一章類別圖中，我們提到設計師必學元素，其中談到類別之間的一般化關係(generalization relationship)。其實，在用例圖中，我們也可以在參與者之間、或者用例之間，建立一般化關係，讓子元素繼承父元素已經定義好的一切。請看到圖 5-1，透過一般化關係，會員也可以使用「查詢民宿資料」、「查詢房型資料」這兩個用例。

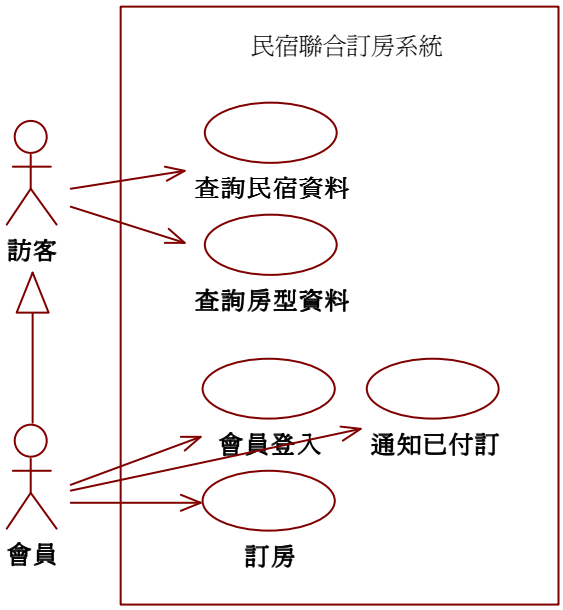

圖 5-1: 參與者之間的一般化關係

其實，本來訪客可以使用的用例，會員都可以使用。所以，此處我們可以對訪客和會員做更明確定義，如下：

- 訪客—所有會瀏覽民宿聯合訂房系統的使用者，都是訪客。

■ 會員—訪客登入系統，同時也經過系統驗證通過者，才是會員。

此外，用例之間的一般化關係，其實比參與者之間的一般化關係更有價值。爲什麼呢？還記得我們在循序圖中，有爲用例設置專屬的控制物件，所以用例之間設置了一般化關係的話，對應其背後的控制物件之間也可能設置一般化關係。

比方說，訂房系統提供了一個「轉帳付訂」的用例，可以讓會員透過網路 ATM 執行線上付訂，如圖 5-2 所示。

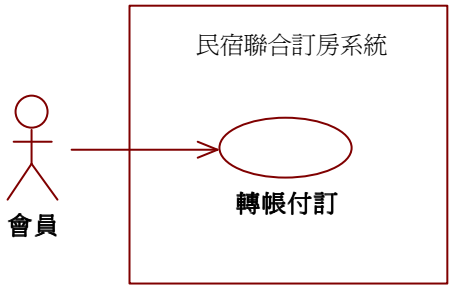

圖 5-2: 轉帳付訂

隨後，分析師發現還需要一個「刷卡付訂」的用例，所以又多了一個用例，如圖 5-3 所示。

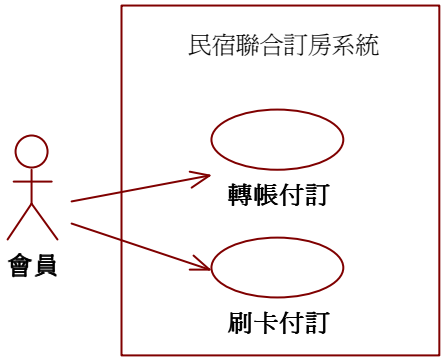

圖 5-3: 刷卡付訂

最後，分析師也寫好了轉帳付訂和刷卡付訂這兩個用例敘述，一併交給設計師。設計師發現這兩個用例敘述，一大段的主要流程都相同，而且這兩個用例又都是線上付訂的概念，所以設計師這時可以爲它們設置一般化關係，把兩個用例相同的流程移到「線上付訂」父用例中，如圖 5-4 所示。

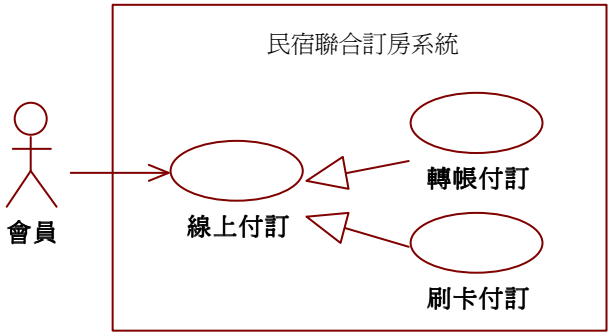

圖 5-4: 線上付訂

而且，用例背後的用例控制類別之間，也可以設置一般化關係，讓子控制類別可以直接繼承父控制類別的一切，如圖 5-5 所示。

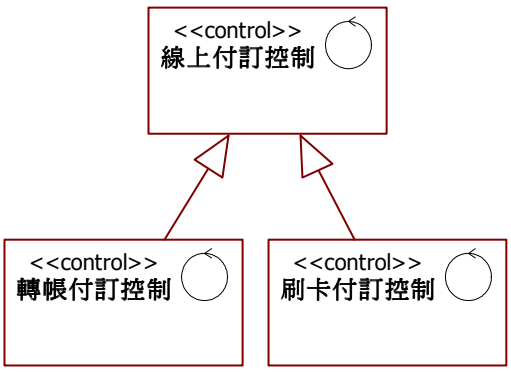

圖 5-5: 控制類別

5.2.2 抽象用例

當然，如果設計師也可以設置「抽象用例」(abstract use case)，表達這個用例的流程不完整，無法實際誕生用例物件。假設在我們的訂房系統中，就只提供兩種付訂方式，這時就可以將線上付訂改成抽象用例，如圖 5-6 所示。

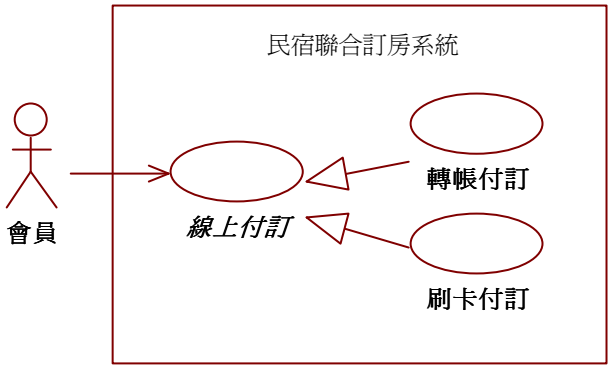

圖 5-6: 抽象用例

同理，用例背後有專屬的控制類別，也可以對應成抽象的控制類別，如圖 5-7 所示。

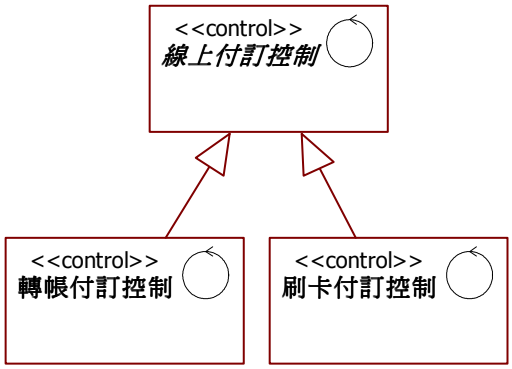

圖 5-7: 抽象控制類別

5.2.3 包含關係

用例之間除了一般化關係外，不同的用例還可以透過「包含關係」(include relationship)來共用(share)相同的用例片段。包含關係的圖示是一條帶箭頭虛線，虛線旁標示<<include>>字眼，由原先的用例指向被抽離出來共用的小用例片段。

比方說，在訂房系統中，訂房和通知已付訂都會發送電郵和簡訊給民宿主人和會員，這一小段流程其實可以抽離出來共用，如圖 5-8 所示。

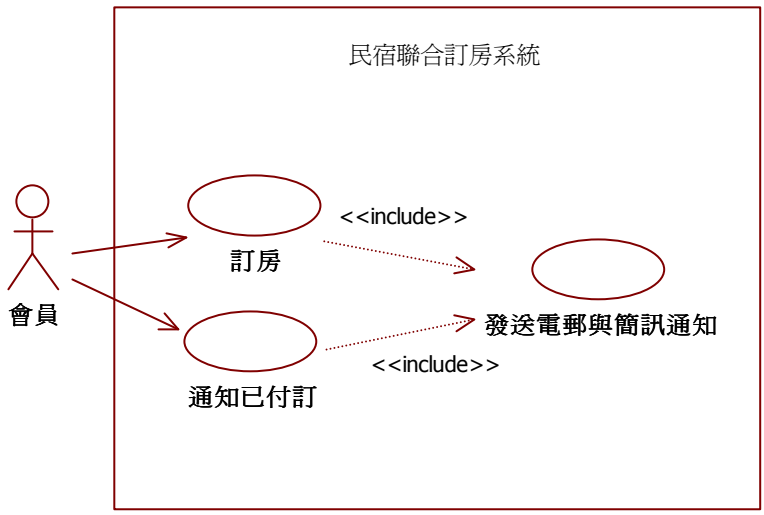

圖 5-8:發送電郵與簡訊通知

還有，另一個「取消預定」用例其實也會需要發送電郵與簡訊通知，所以這時就可以快速的共用已經設計好的用例了，如圖 5-9 所示。

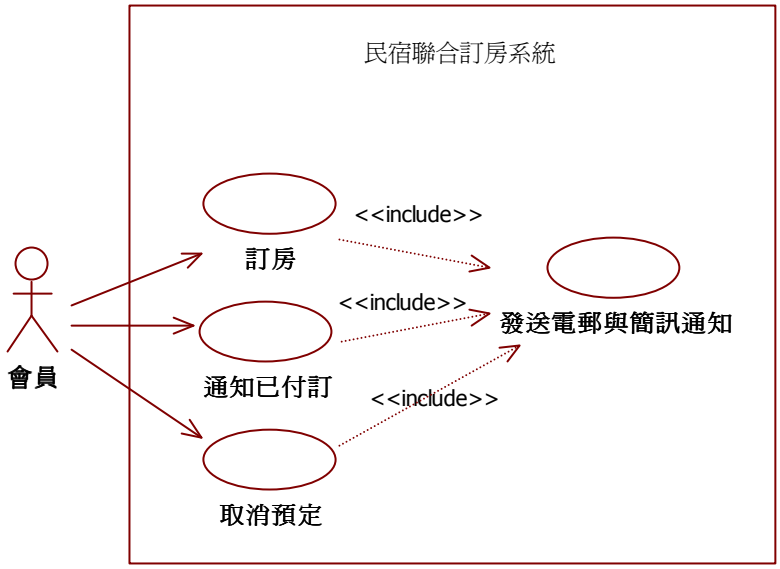

圖 5-9: 包含關係

至於，相對應的用例控制類別上，也可以透過依賴關係來調用，如圖 5-10 所示。請再看到圖 5-11，這是通知已付訂用例的循序圖雛型，付訂控制物件可以發訊息給電郵簡訊通知控制物件，進行用例之間的串接。

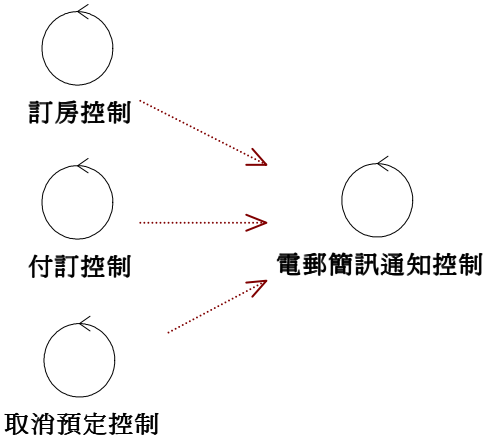

圖 5-10: 透過依賴關係調用

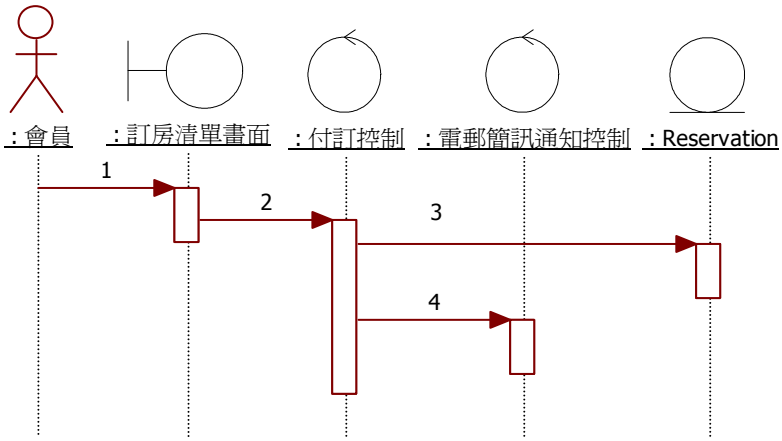

圖 5-11: 「通知已付訂」用例

5.2.4 擴充關係

用例之間有另一種「擴充關係」(extend relationship)，也是可以讓多個用例共用(share)小的用例片段。不過，從「包含」關係和「擴充」關係的名稱上頭，其實很快就可以理解到兩者的不同。包含關係是大包小，大用例的流程包含了小用例的流程；擴充關係則是小擴大，小用例的流程擴充了大用例的流程。

也因此，這兩個關係在圖示的表示上，箭頭方向剛好相反。包含關係是大指向(包含)小，擴充關係則是小指向(擴充)大，如圖 5-12 所示。擴充關係的圖示也是帶箭頭虛線，虛線旁標示<<extend>>字眼代表擴充關係，並且由小的用例片段指向被擴充的大用例(基礎用例)。

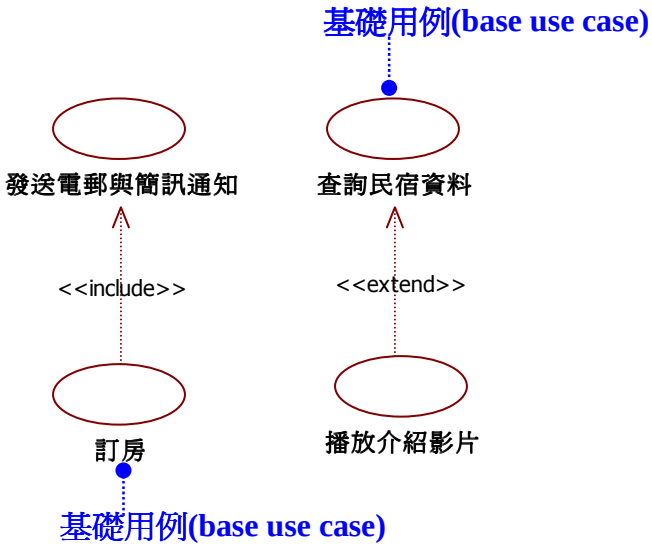

圖 5-12: 包含關係與擴充關係

此外，在包含關係上，執行基礎用例時，一定得一併執行所有的小用

例；但是，在擴充關係上，基礎用例執行期間是依條件執行小用例的，也就是說，當條件皆未符合的情況下，可能未執行任何擴充的小用例。請看到兩者簡單的比較表，如表 5-1 所示。

包含關係	擴充關係
標示<<include>>的帶箭頭虛線	標示<<extend>>的帶箭頭虛線
基礎用例指向被包含的小用例	小用例指向被擴充的基礎用例
一定要執行小用例	條件成立才執行小用例

表 5-1: 包含關係與擴充關係

所以，現在我們看到圖 5-13 的例子，您就能夠理解它其實代表著三條流程，如下：

1. 訂房－執行訂房用例期間，一定會包含一小段的發送電郵與簡訊通知。
2. 查詢民宿資料－單純的查詢民宿資料，並未執行任何的擴充的小用例。
3. 查詢民宿資料－執行查詢民宿資料期間，還額外執行了一小段擴充的播放介紹影片。

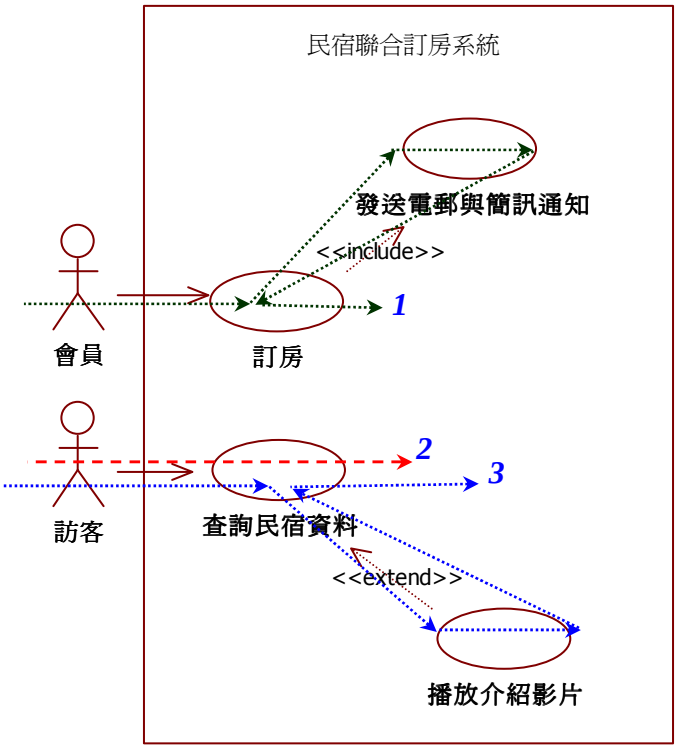

圖 5-13: 三條流程

既然，在訂房系統中，我們已經有了一個「播放介紹影片」的用例片段了，當然也可以透過擴充關係，來擴充另一個查詢房型資料的基礎用例了，如圖 5-14 所示。

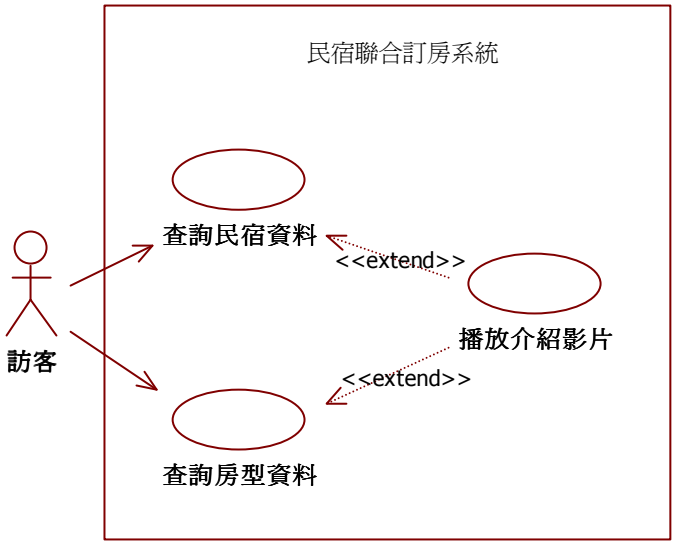

圖 5-14: 「播放介紹影片」用例

最後，我們看到用例控制類別上，就跟包含關係的設計相同，可以透過依賴關係來調用。不過，原先的查詢民宿資料並沒有對應的控制物件，如圖 5-15 所示。所以，我們可以先產生對應的用例控制物件，讓查詢民宿資料的控制物件調用播放介紹影片的控制物件，如圖 5-16 所示；或者，直接產生播放介紹影片的用例控制物件，讓民宿畫面邊界物件直接調用，如圖 5-17 所示。

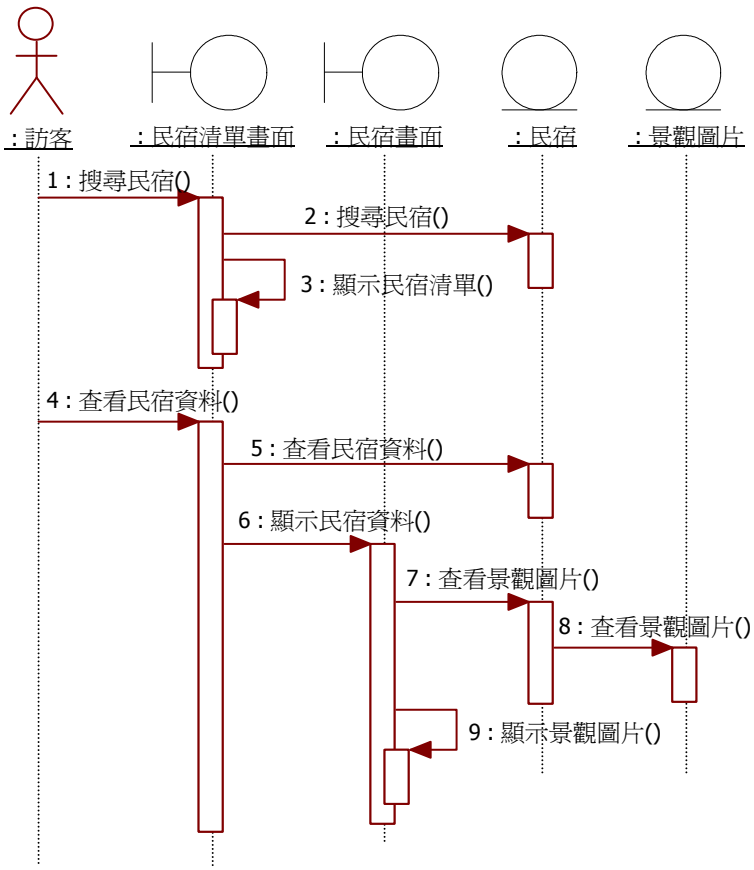

圖 5-15:「查詢民宿資料」用例的循序圖

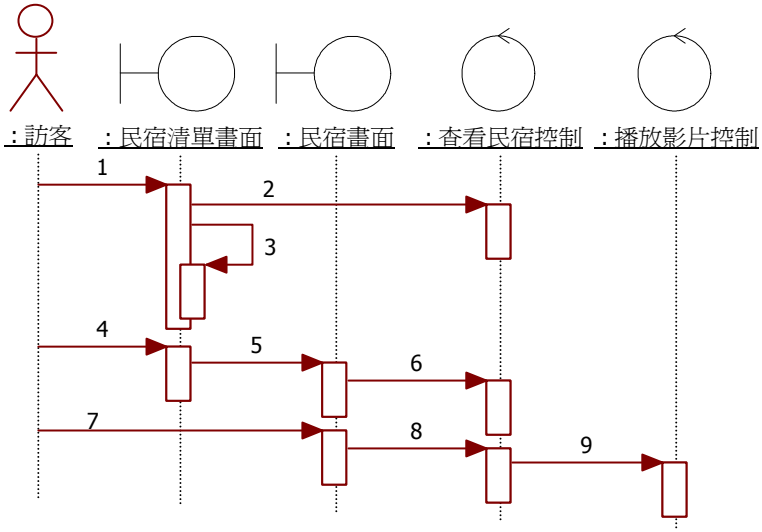

圖 5-16: 查看民宿控制物件

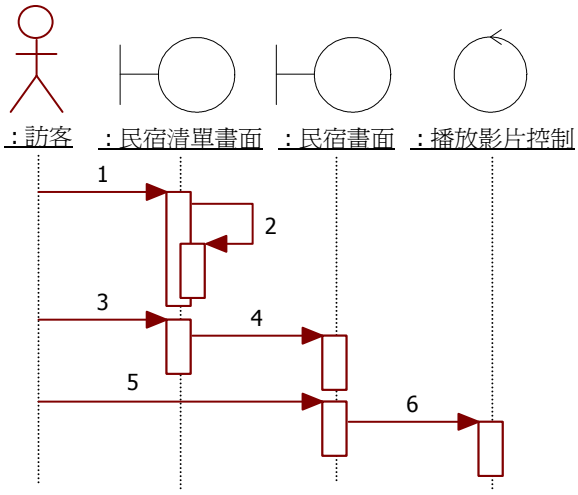

圖 5-17: 撥放影片控制物件

5.3 用例敘述

分析師在一開始撰寫用例敘述時，總是難以描述詳細，這是很正常的，因爲當時對於領域、系統一無所知，能夠捕捉到重點資訊，就已經貢獻良多了。

所以，設計師接下來，必須爲用例敘述加工，添加更多程序員所需的細節資訊，重點加工之處如下：

1. 用例之間的關係—如果有使用包含關係和擴充關係的話，在用例敘述處，就要記得說明何時會執行被包含的小用例，或者何時會執行擴充的小用例。
2. 邊界物件的溝通介面—參與者物件與邊界物件之間的溝通介面，必須描述的更明確些，包括參與者物件傳送哪些資料給邊界物件，而邊界物件又傳回了哪些資料。
3. 加入僞畫面—前面，我們曾經提到分析師可以繪製出僞畫面協助溝通，如果分析師在專案一開始未繪製出僞畫面的話，在設計師開始爲分析文件細部加工的此時，最好已經是繪製出僞畫面了，這樣就可以搭配用例敘述把邊界物件說得更詳細些。同時，還可以將這個僞畫面一併交付給程序員，請程序員依照此僞畫面設計出功能測試畫面。
4. 更新 BCE 類別—前面我們已經更新過實體類別了，所以此時更新的重點會擺在邊界類別與控制類別。當然，趁這個機會，設計師可以再跑一次循序圖，因爲用例敘述加工後，可能會動到循序圖或實體類別。

以訂房系統的「會員登入」用例爲例，我們增加了兩個僞畫面，直接用僞畫面協助說明參與者物件該輸入哪些資料，以及系統會回應什麼樣的資料，同時局部修改了用例敘述，如表 5-2 所示。

用例	會員登入		
啓動者	會員	**支援者**	

主要流程

1. 會員輸入電郵和密碼，參考圖 5-18 的會員登入畫面。
2. 系統驗證會員身分。
3. 系統顯示歡迎訊息，其中包含會員姓名，以及等待付訂金交易筆數，參考圖 5-19 會員登入成功後的畫面。

僞畫面

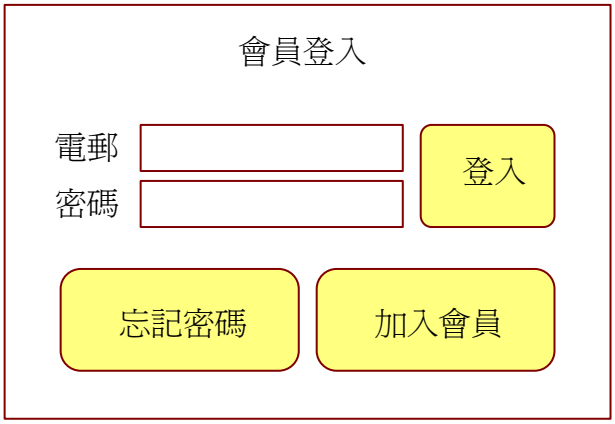

圖 5-18: 登入畫面(登入前)

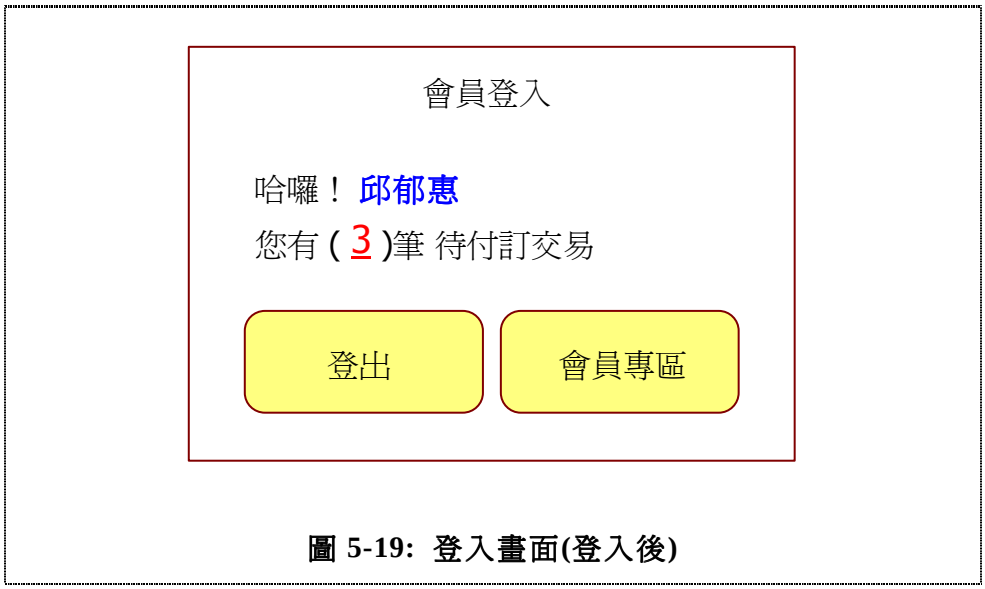

圖 5-19: 登入畫面(登入後)

表 5-2: 「會員登入」的主要流程

其實，我們也不一定要使用特殊的軟體來繪製僞畫面，像圖 5-18~19 的僞畫面，我就直接使用 StarUML 提供的矩形圖示、文字方塊等簡單的圖案元件來繪製。不過，這樣的效能差多了，如果是開發正式的專案，還是建議別省這筆錢，買個實用的繪製僞畫面軟體，相信一定能夠會大大提升開發品質和產能的。

好了，廢話不多說，接下來我們就直接來加工訂房系統的用例吧！

5.4 民宿聯合訂房系統

5.4.1 用例—會員登入

會員登入的用例敘述，請直接參考上述的表 5-2，此處就不再重複說

明了。至於，會員登入用例的循序圖，之前因爲已經更新過實體類別，所以相對應的訊息名稱會一併更新，目前最新的循序圖如圖 5-20 所示。

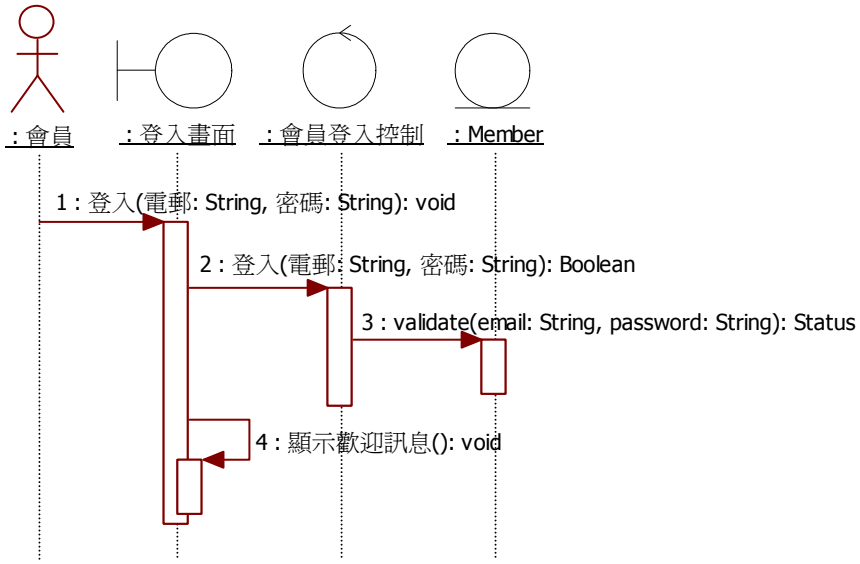

圖 5-20: 「會員登入」的循序圖(更新實體類別之後)

接著，我們就直接把登入畫面和會員登入控制這兩個類別，從中文改成英文，同時釐清輸出入參數。因爲登入成功之後，必須顯示會員姓名和待付訂交易數量，所以修改了登入畫面中的「顯示歡迎訊息」操作，如圖 5-21 所示。

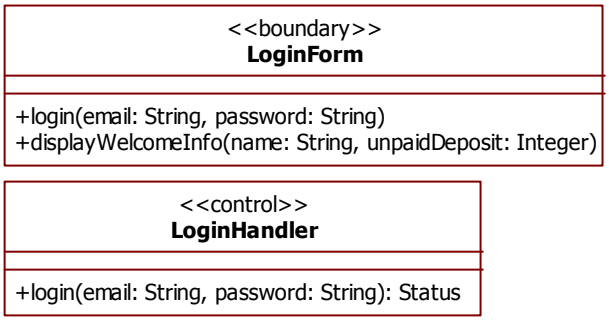

圖 5-21: 登入畫面與會員登入控制

最後，設計師可以配合用例敘述、僞畫面，用手指和腦袋跑一次圖 5-22，特別是輸出入參數的地方，看看是否需要調整。

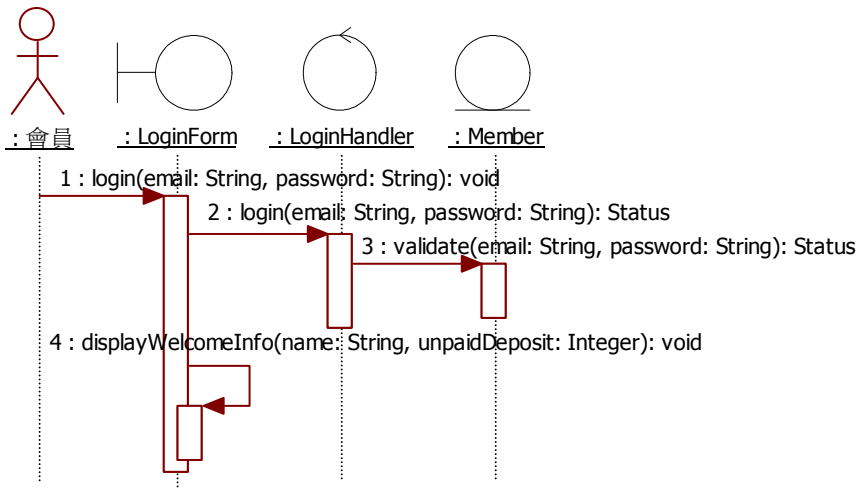

圖 5-22: 「會員登入」的循序圖

跑完圖 5-22 之後，您可能跟我一樣已經發現 3 號訊息到 4 號訊息之間可能有些問題，因爲 3 號訊息沒有回傳會員姓名和待付訂交易數量，4 號訊息根本無法顯示出這兩項資訊。所以，我們修改了循序圖，主要增加了 4~7 號訊息，如圖 5-23 所示。

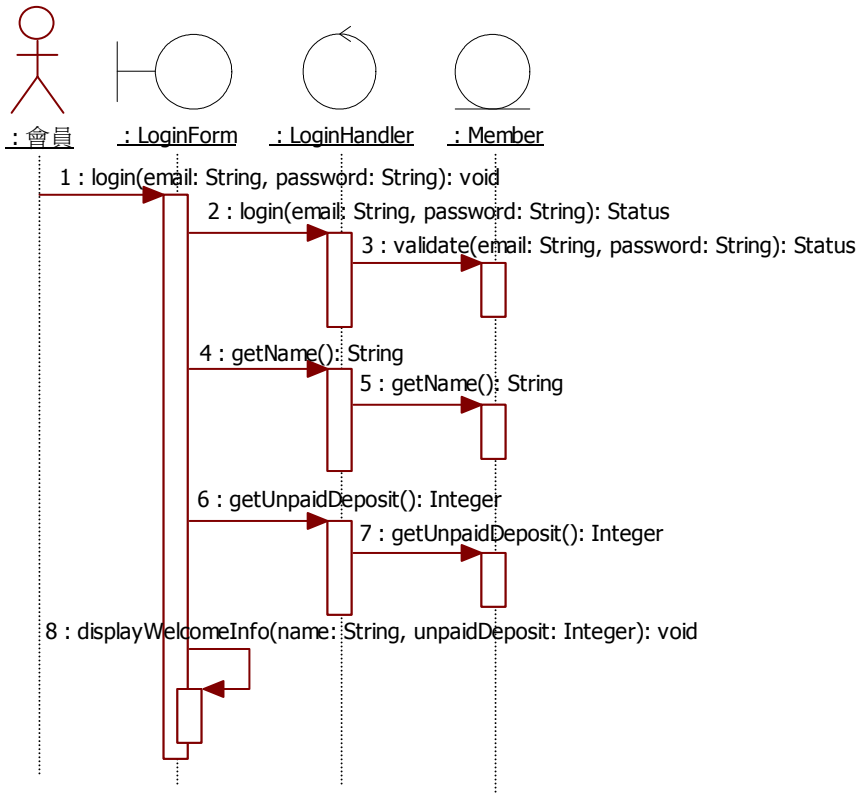

圖 5-23: 增加了 4~7 號訊息

每修改一次循序圖，我們就得用手指和腦袋重新跑一次，再跑一次圖 5-23，發現兩個問題：

1. 會員的「驗證」(validate)操作是類別層級，原先的構想是當確認真的有這個會員時，才會在記憶體中建構一個對應的會員物件。接下來，控制物件才能夠跟會員物件取得姓名等屬性資料。不過，此處的驗證操作回傳的是「狀態」(status)，所以控制物件其實還未取得會員物件。因此，我們要修改這個驗證操作，讓它回傳會員物件。
2. 剛才在圖 5-23 中，我們爲會員實體類別新增了「取得待付訂交易數量」(getUnpaidDeposit)操作，但是之前我們並未在實體類別圖或其他用例中，計算或記錄過待付訂交易數量，而這件事情顯然不該是這個用例該做的事情。這邊的處理方式是，直接在會員實體類別中新增一個「待付訂交易數量」屬性，並且交由「訂房」用例負責增加待付訂交易數量，如表 5-3 中的 7 號步驟。再由「通知已付訂」用例負責減少待付訂交易數量，更新這兩個用例敘述，如如表 5-4 中的 4 號步驟。

用例	訂房		
啓動者	會員	**支援者**	
主要流程 1. 會員挑選一家民宿。 2. 系統秀出這家民宿所有的房型名稱、床型、空房數和房價。 3. 會員挑選預定的房型、房間數以及預訂日期。			

4. 系統顯示出訂房總價。
5. 系統新增一筆訂房交易。
6. 系統減少可預訂的空房數。
7. 系統增加一筆未付訂交易筆數。
8. 系統發送訂房通知給民宿主人。
9. 系統發送訂房通知給會員。
10. 系統秀出交易代號、訂金與總價。
11. 系統提醒會員需要 48 小時內付訂金。

表 5-3: 修改「訂房」的主要流程

用例	通知已付訂		
啓動者	會員	**支援者**	
主要流程			
1. 會員選擇一筆未付訂的訂房交易。			
2. 會員填入付訂金額、付訂帳號、付訂時間。			
3. 系統記錄付訂資料。			
4. 系統減少一筆未付訂交易筆數。			
5. 系統發送付訂通知電郵或簡訊給民宿主人。			
6. 系統發送付訂通知電郵或簡訊給會員。			

表 5-4: 修改「通知已付訂」的主要流程

最後，再修訂一次循序圖，增加一個取得會員狀態的 4 號訊息，如圖 5-24 所示。除了邊界類別外，其餘的控制類別和會員實體類別都已經變動，所以我們重新列出更新的最後模樣，如圖 5-25 所示。

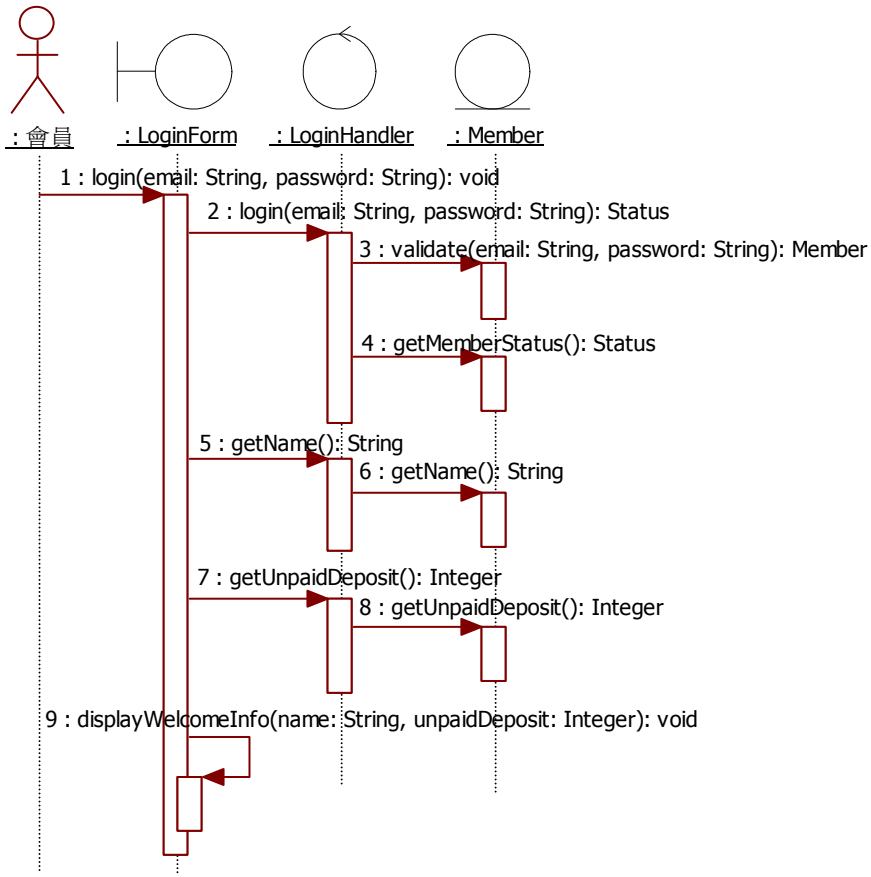

圖 5-24: 「會員登入」的循序圖

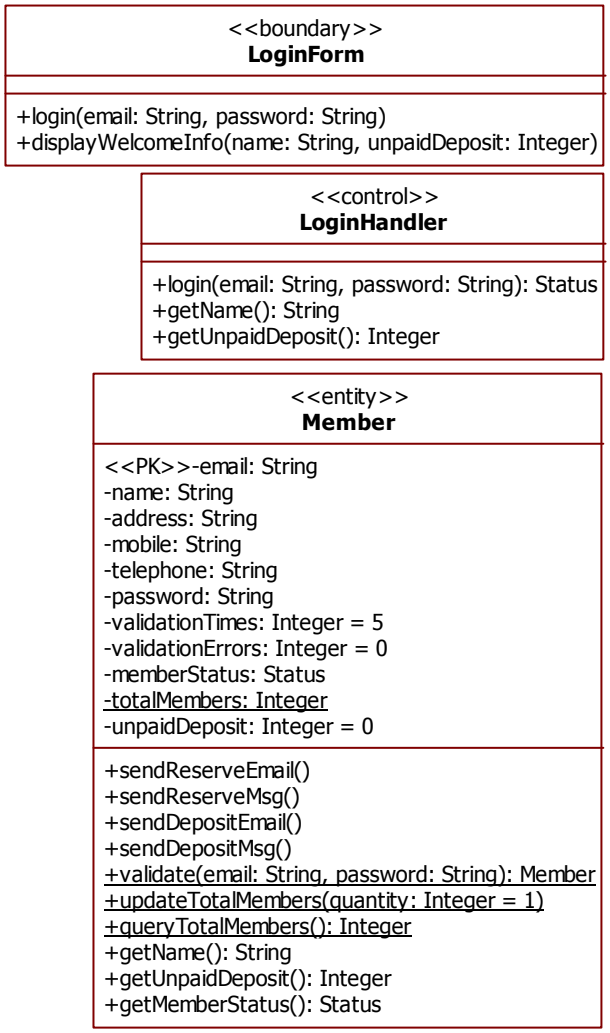

圖 5-25: 修改會員實體類別

5.4.2 用例—通知已付訂

之前，我們已經決定將訂房和通知已付訂中的「發送電郵與簡訊通知」

流程抽離出來，如圖 5-26 所示。而且將原先表 5-4 中的 5~6 號步驟先搬到表 5-5 中，讓通知已付訂用例直接執行「發送電郵與簡訊通知」用例。等我們先處理完此處的通知已付訂用例，稍後我們再來處理發送電郵與簡訊通知。

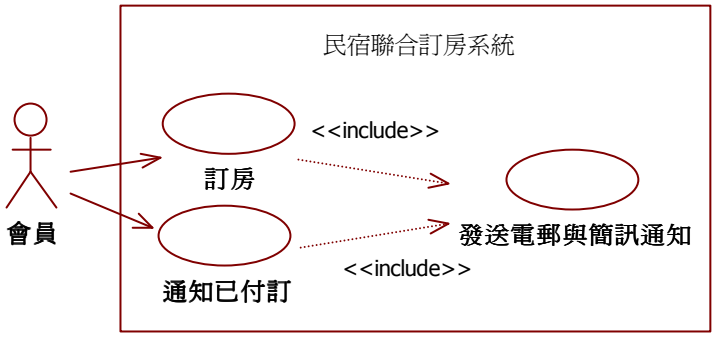

圖 5-26: 發送電郵與簡訊通知

用例	發送電郵與簡訊通知		
啓動者	會員	**支援者**	
主要流程 1. 系統發送付訂通知電郵或簡訊給民宿主人。 2. 系統發送付訂通知電郵或簡訊給會員。			

表 5-5:「發送電郵與簡訊通知」的主要流程

然後，我們先看到之前加工實體類別後，最新的「通知已付訂」循序圖，如圖 5-27 所示。

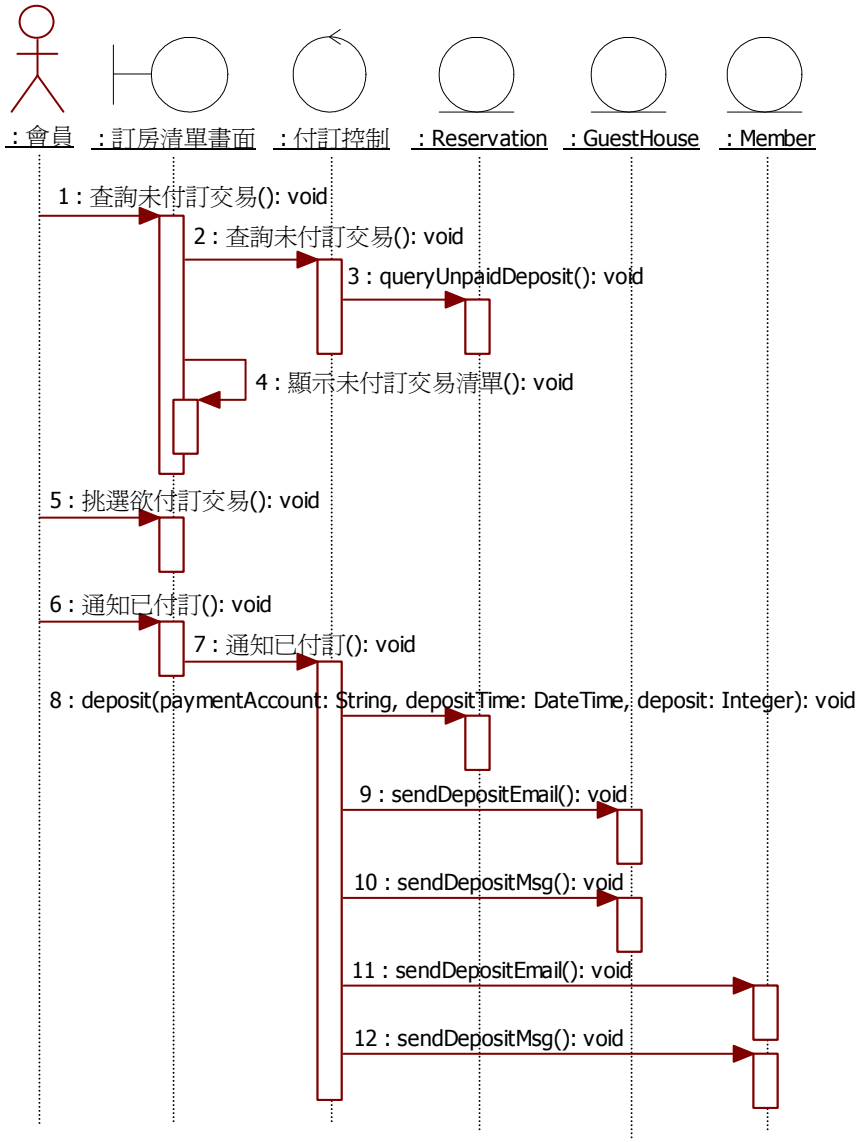

圖 5-27: 「通知已付訂」的循序圖(更新實體類別之後)

接著，我們直接更新「通知已付訂」用例敘述，並且增加僞畫面，如表 5-6 所示。特別注意到表 5-6 中，我們增加了一個欄位記錄相關用例。

用例	通知已付訂		
啓動者	會員	**支援者**	
相關用例	包含—發送電郵與簡訊通知		
主要流程 1. 會員查詢所有待付訂交易，參考圖 5-28 會員登入成功後的畫面。 2. 會員選擇一筆未付訂的訂房交易，參考圖 5-29 訂房清單畫面。 3. 會員挑選匯款銀行(代號-名稱)、填寫銀行帳號末 5 碼、挑選匯款日期、選擇匯款時間、填入匯款金額，參考圖 5-30 填寫付訂資料畫面。 4. 系統記錄付訂資料，包含匯款銀行(代號-名稱)、銀行帳號末 5 碼、匯款日期、匯款時間、匯款金額。 5. 系統更新訂房交易狀態。 6. 系統減少一筆未付訂交易筆數。 7. 系統執行「發送電郵與簡訊通知」用例。 8. 系統回到第 2 步驟，更新訂房清單畫面。			
僞畫面			

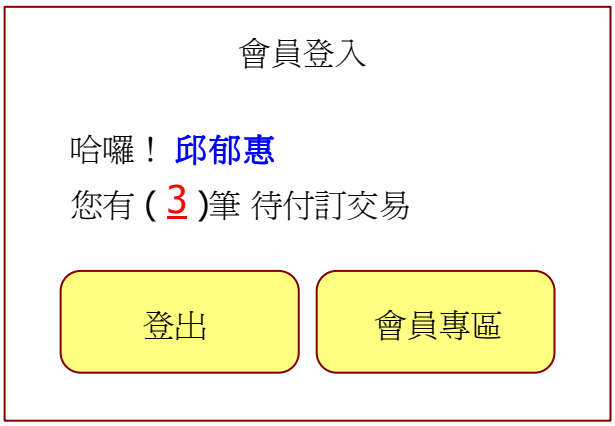

圖 5-28: 登入畫面(登入後)

訂房交易

民宿	訂房日期	預訂日期	付訂日期	總金額	訂金	狀態
浪漫滿屋	2009/12/10	2009/12/24	----/--/--	10,000	1,000	未付訂
星光部落	2009/12/11	2009/12/25	----/--/--	8,000	800	未付訂
開心民宿	2009/12/12	2009/12/26	----/--/--	6,000	600	未付訂

圖 5-29: 訂房清單畫面(所有未付訂交易列表)

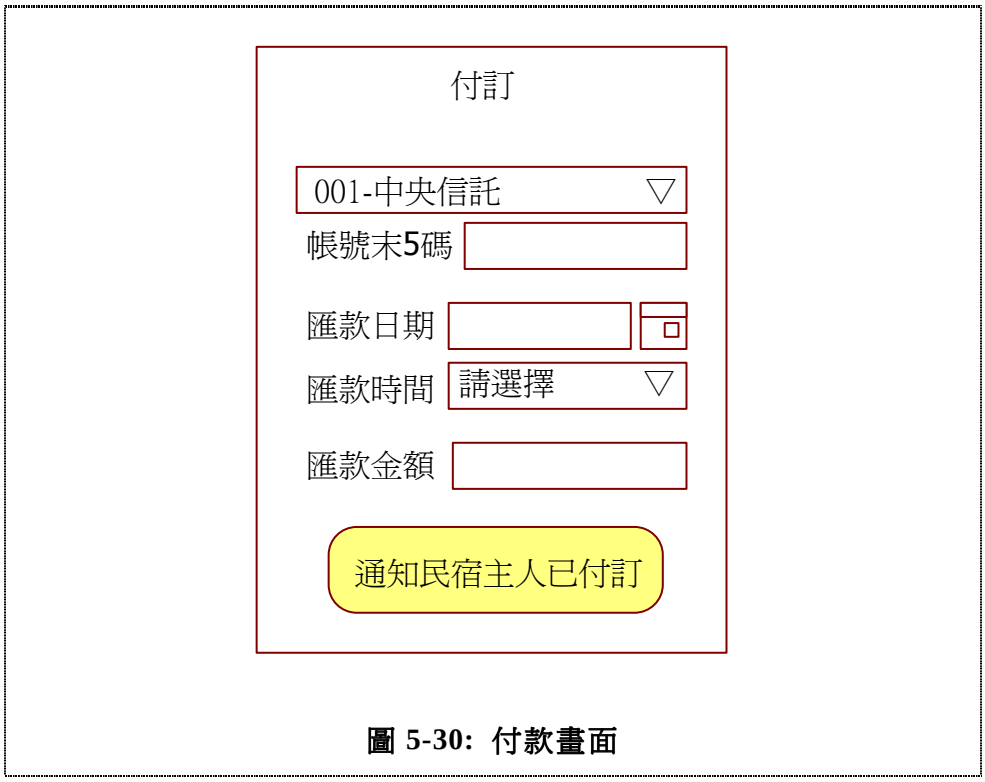

圖 5-30: 付款畫面

表 5-6: 修改「通知已付訂」的主要流程

回過頭看圖 5-27 的循序圖，由於切出小用例，所以民宿和會員實體物件就用不到了，但是要增加一個新的「通知控制」(NotifyHandler)物件，以及一個登入畫面(LoginForm)物件和另一個新的「付訂畫面」(DepositForm)物件，如圖 5-31 所示。

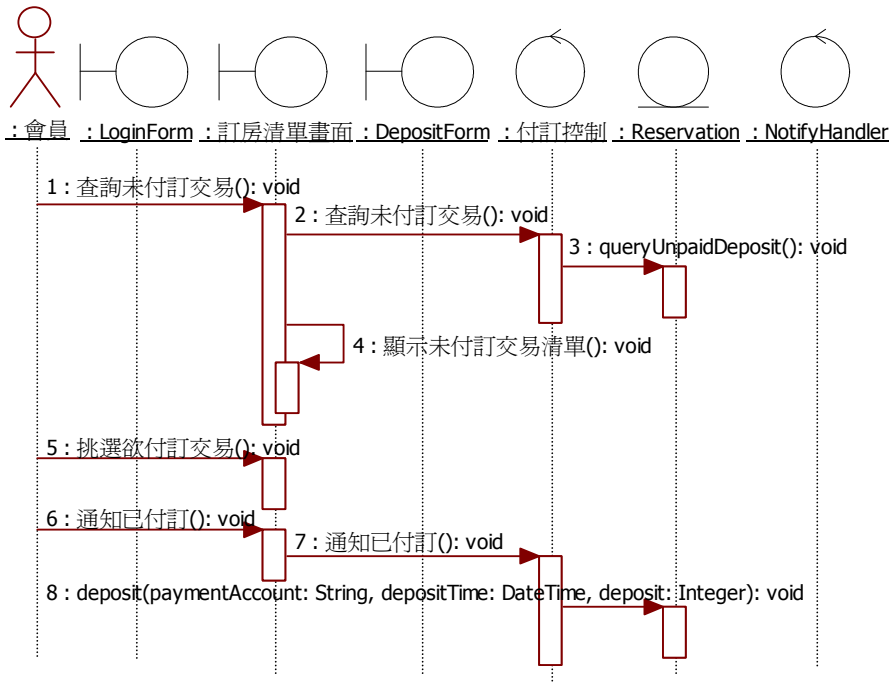

圖 5-31: 增刪了一些物件

接著，我們針對修改後的表 5-6 用例敘述，初步修改出如圖 5-32 的循序圖。除了把中文改成英文外，我們發現需要會員實體物件來記錄未付訂交易數量，所以把會員實體物件又重新加進來了。再者，因爲畫面物件和控制物件都變多了，所以責任的分派上也有變動。

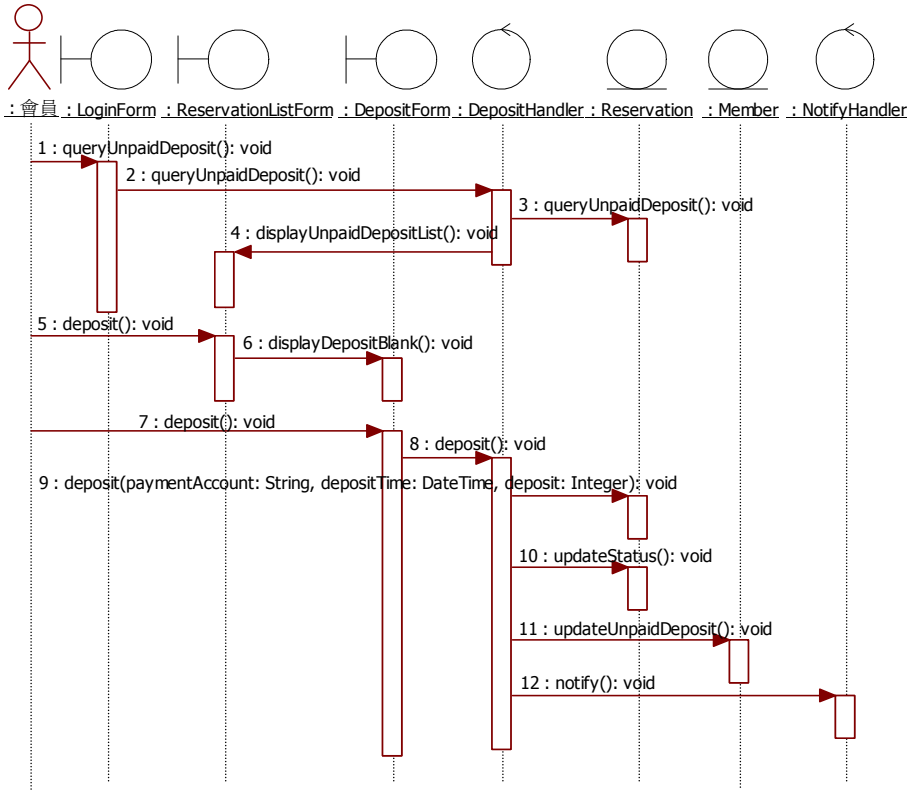

圖 5-32: 「通知已付訂」的循序圖(初步修改)

現在，我們再用手指和腦袋來跑一下循序圖 5-32，修正了輸出入參數，修改成圖 5-33 的樣子。

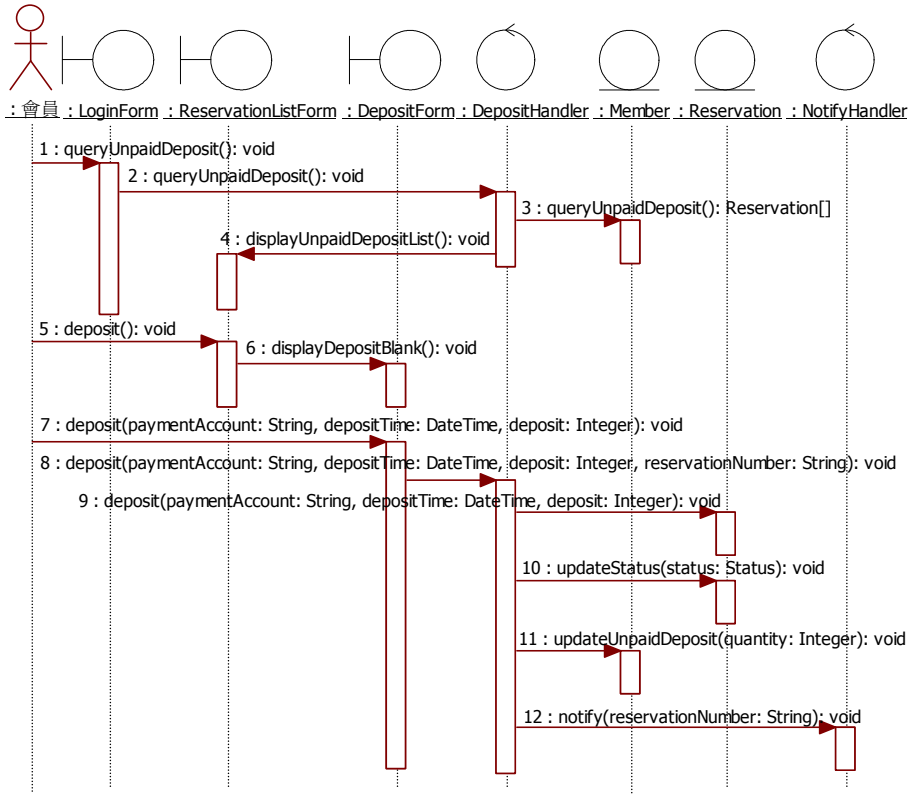

圖 5-33: 「通知已付訂」的循序圖(修改輸出入參數)

在圖 5-33 中，我們主要修改了下列幾項訊息：

1. 3 號訊息—原本請訂房類別查出未付訂交易，現在改成請會員物件查出名下未付訂交易。所以，會員物件會傳回「一組」未付訂的訂房物件，不止一個喔。
2. 8 號訊息—付訂畫面物件多傳了一項訂房序號(reservationNumber)

給付訂控制物件，讓付訂控制物件可從 3 號訊息中的一組訂房物件中，找出正確的訂房物件。

3. 10 號訊息—狀態(Status)列舉型別多了一個「已付訂」(deposited)列舉值，此操作的輸入參數值是「已付訂」狀態。

4. 11 號訊息—請會員實體物件負責更新已付訂交易，其輸入參數的預設值爲「-1」。

5. 12 號訊息—發送電郵與簡訊給民宿主人和會員的流程已經搬到「發送電郵與簡訊通知」用例了，所以此處透過控制物件傳送欲發送通知的訂房序號。

最後，我們再跑一次圖 5-33，發現漏了表 5-6 用例敘述中的 8 號步驟，所以更新了循序圖，新增了圖 5-34 中的 13~15 號訊息。

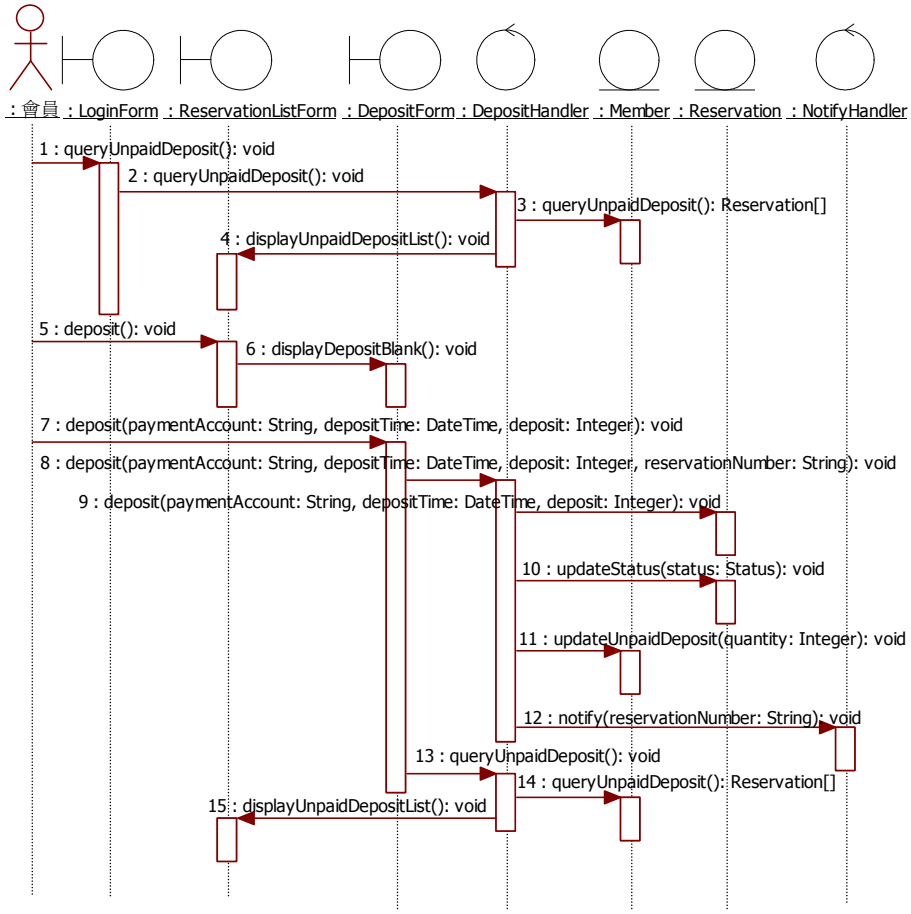

圖 5-34: 「通知已付訂」的循序圖(新增 13~15 號訊息)

最後，我們在圖 5-35~36 列出更新的 BCE 類別。

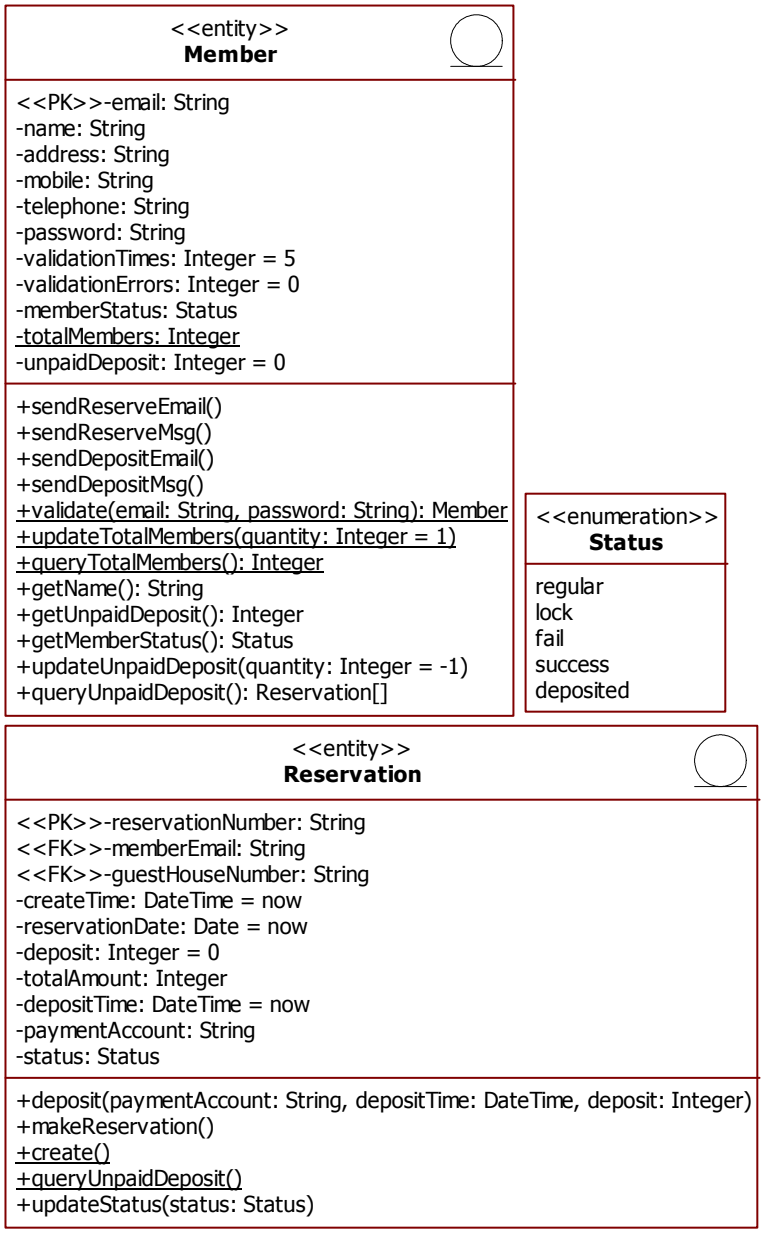

圖 5-35：更新的實體類別和列舉型別

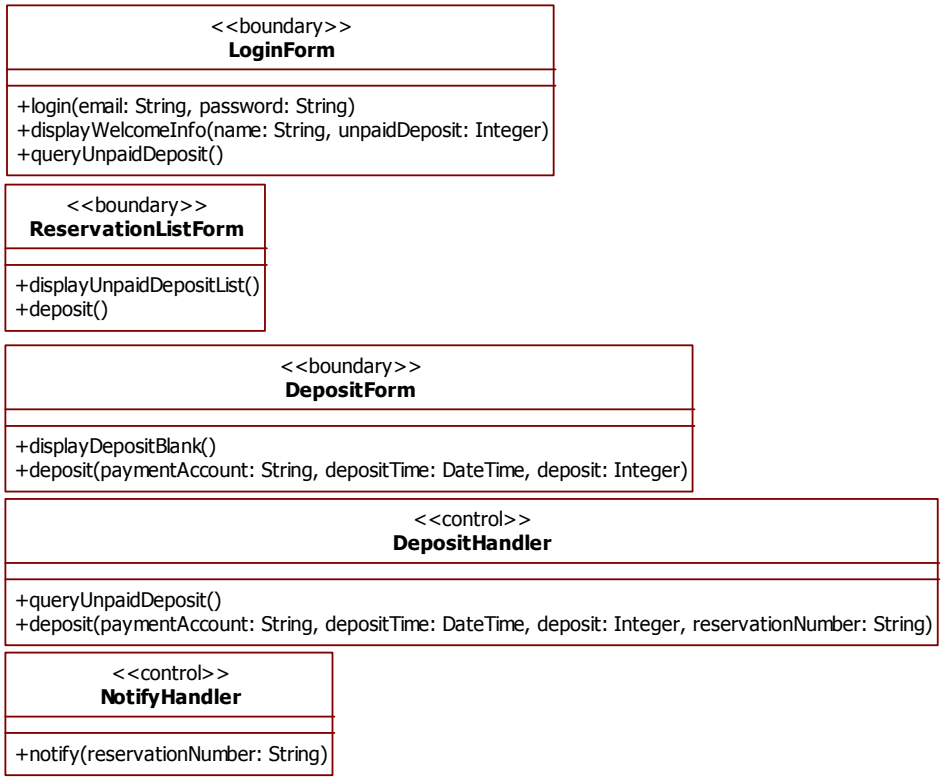

圖 5-36: 更新的邊界類別和控制類別

5.4.3 用例—發送電郵與簡訊通知

由於，通知已付訂用例和訂房用例會共用「發送電郵與簡訊通知」用例，所以我們重新撰寫了用例敘述，如表 5-7 所示。在表 5-7 中，我們特別增加了一個小欄位，註記這個小用例沒有畫面。再者，我們也把相關的控制物件和實體物件都先擺到循序圖中了，如圖 5-37 所示。

用例	發送電郵與簡訊通知		畫面	無
啓動者	會員	支援者		

主要流程

1. 系統依據訂房序號，找出該訂房交易的會員與民宿主人。
2. 系統產生電郵，內容包含：訂房序號、訂房狀態、訂房日期、預定日期、付訂日期、總金額。
3. 系統發送電郵給民宿主人。
4. 系統發送電郵給會員。
5. 系統產生簡訊，內容包含：訂房序號、訂房狀態、訂房日期、預定日期、付訂日期、總金額。
6. 系統發送簡訊給民宿主人。
7. 系統發送簡訊給會員。

表 5-7:「發送電郵與簡訊通知」的主要流程

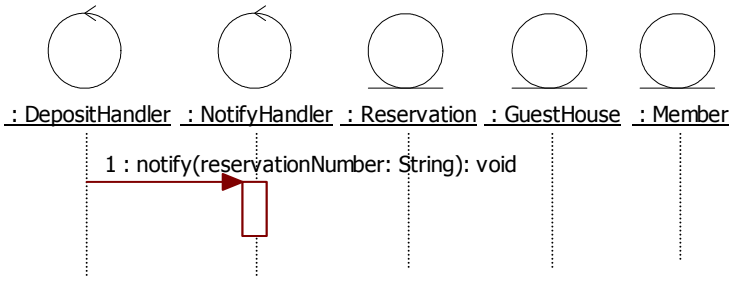

圖 5-37: 「發送電郵與簡訊通知」的控制物件和實體物件

接下來，我們就按照用例敘述的步驟直接設計循序圖了，初步設計出如圖 5-38 的循序圖。

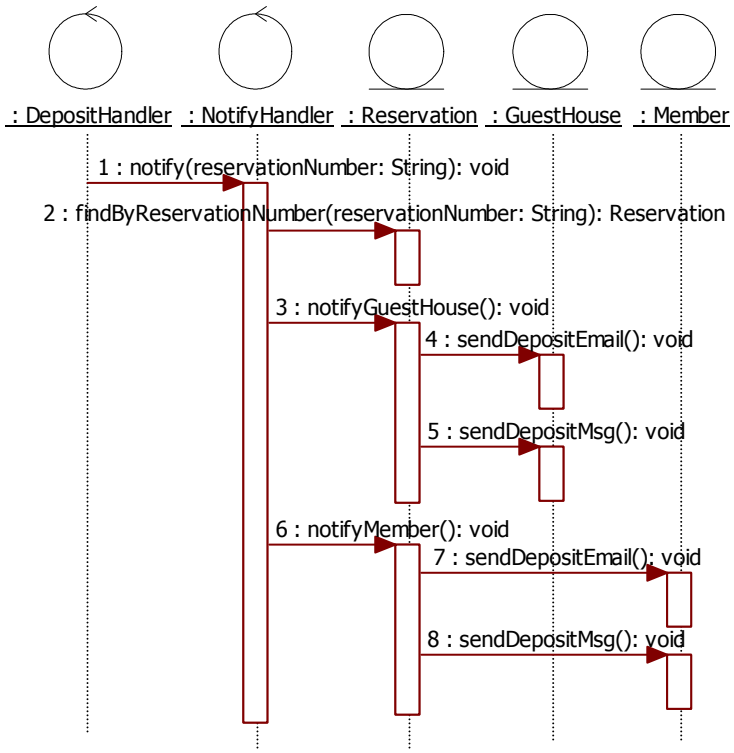

圖 5-38: 加上訊息

不過，圖 5-38 的 4~5 號訊息和 7~8 號訊息可能會有問題，因爲訂房用例也會用到「發送電郵與簡訊通知」用例。請看到圖 5-39 中，民宿和會員實體類別之前針對發送付訂通知和訂房通知設計了不同的操作，所以現在

我們要將這些操作合併起來，更新爲「發送電郵」(sendEmail)和「發送簡訊」(sendMsg)。

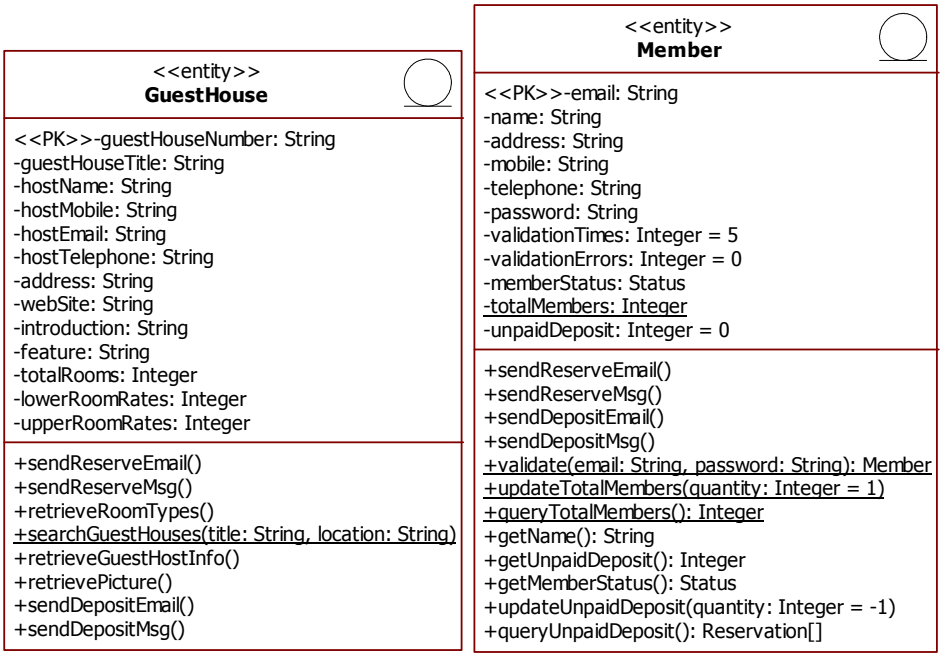

圖 5-39: 民宿與會員實體類別

所以，我們把循序圖同步更新成圖 5-40，圖中的 4~5 號訊息和 7~8 號訊息已經更新。

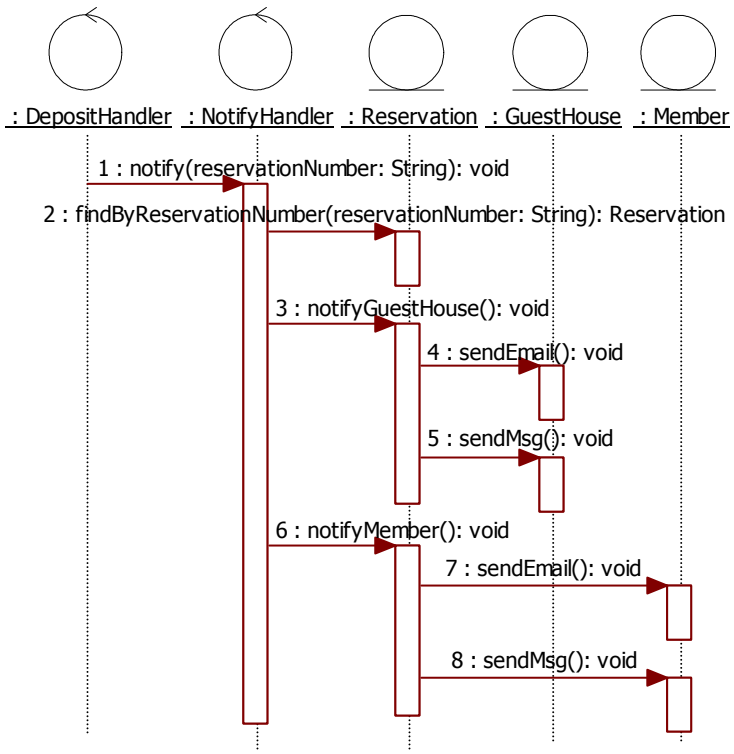

圖 5-40: 更新 4~5 訊息和 7~8 號訊息

接著，我們跑了一下圖 5-40 之後，發現如果將發送電郵和簡訊的工作交由民宿和會員實體物件來做的話，訂房實體物件必須傳遞多項交易資料給民宿和會員。但是，如果反過來讓訂房實體物件自行發送通知的話，它只需取得取得民宿主人和會員的電郵和手機號碼就可以了，所以我們把循序圖改成圖 5-41 的樣子。

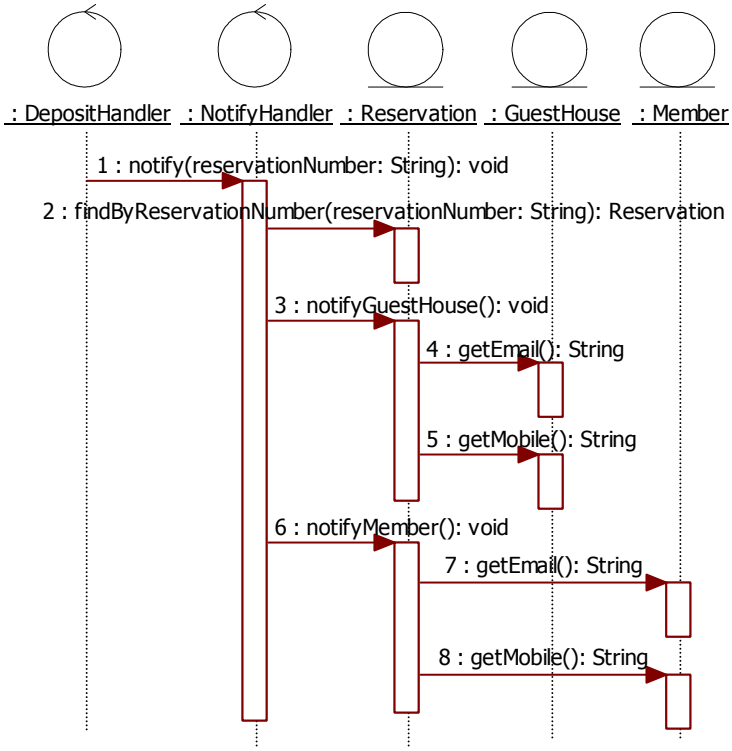

圖 5-41: 「發送電郵與簡訊通知」的循序圖

最後，我們列出更新之後的控制類別和實體類別，如圖 5-42~43 所示。

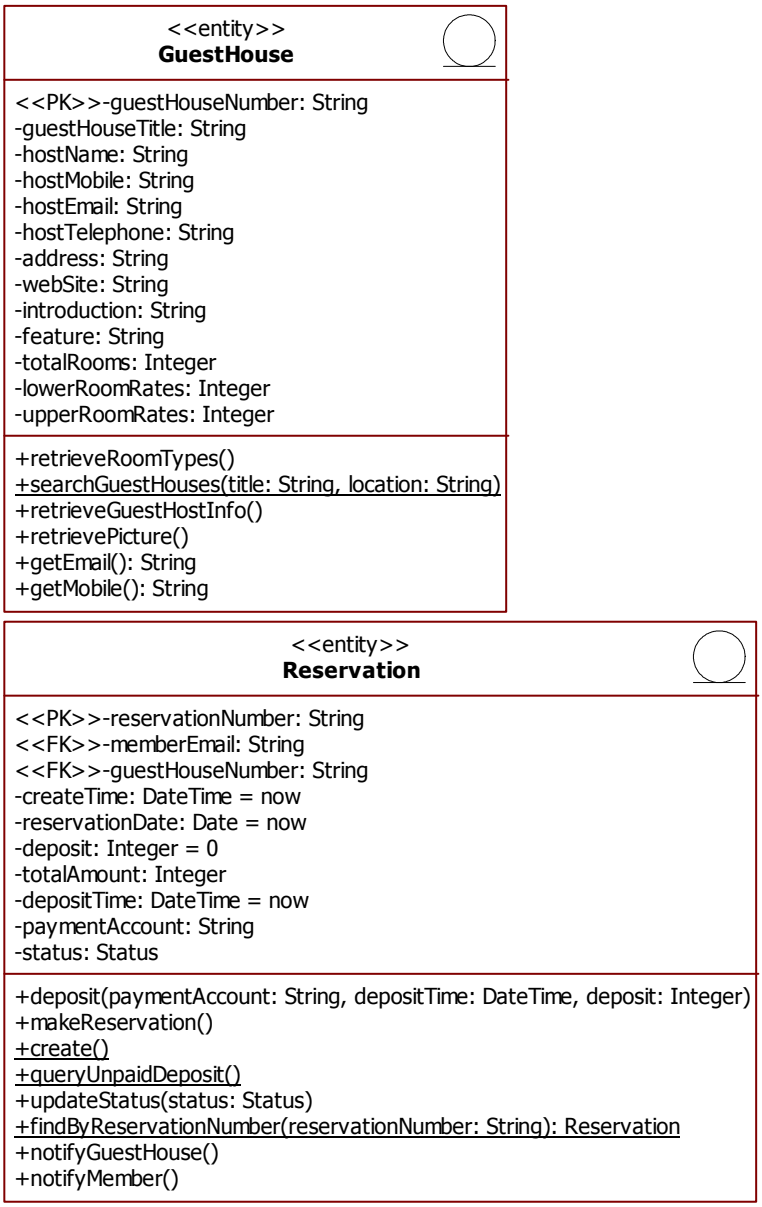

圖 5-42: 民宿與訂房實體類別

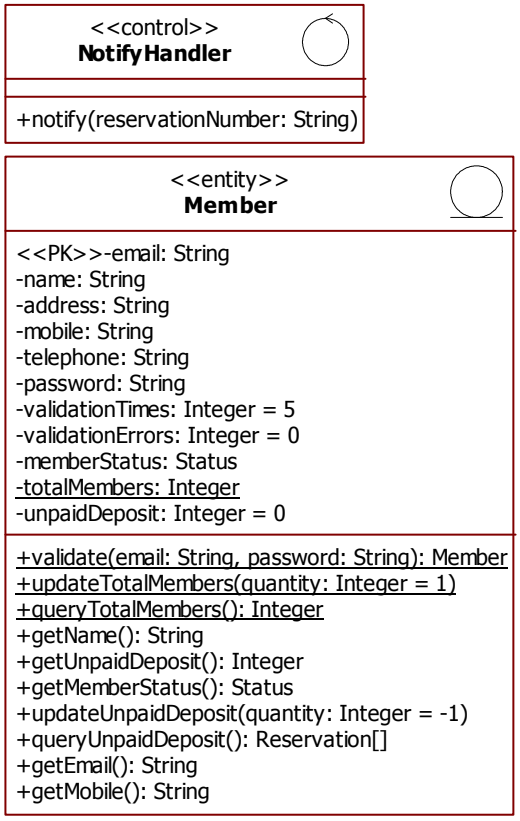

圖 5-43: 控制類別和會員實體類別

5.4.4 用例—查詢民宿資料

請您直接看表 5-8 的「查詢民宿資料」用例敘述，除了增加兩個僞畫面外，我們稍微修改了敘述內容，在顯示清單的部分多秀了民宿小圖，不同於景觀圖片的大圖，所以在民宿實體類別中會需要多加一個「照片」(photo)屬性。

用例	查詢民宿資料		
啓動者	訪客	**支援者**	
主要流程 1. 訪客輸入欲搜尋的民宿地點或名稱。 2. 系統找出符合搜尋條件的民宿清單，內容包含：小圖、民宿名稱、地址、房間數、房間價位，參考民宿清單畫面如圖 5-44 所示。 3. 訪客從中點選某一家民宿，查看民宿資料。 4. 系統顯示民宿資料，除了上述第 2 步驟的資料外，還額外包含民宿網址、簡介、特色、景觀照片，參考民宿畫面如圖 5-45 所示。			
僞畫面			


民宿清單
蘇澳
地點
搜尋
民宿小圖
浪漫滿屋
地址：宜蘭縣蘇澳鎮民眾路一段254號
房間：8間
價位：NT$6,000~12,000
民宿小圖
星光部落
地址：宜蘭縣蘇澳鎮星光路二段20號
房間：6間
價位：NT$3,000~10,000
民宿小圖
開心民宿
地址：宜蘭縣蘇澳鎮光明路104號
房間：8間
價位：NT$2,000~8,000

圖 5-44: 民宿清單畫面

圖 5-45: 民宿畫面

表 5-8: 「查詢民宿資料」的主要流程

先看到圖 5-46，這是之前加工實體類別之後，同步更新的循序圖，等會我們就以此循序圖再進行加工。

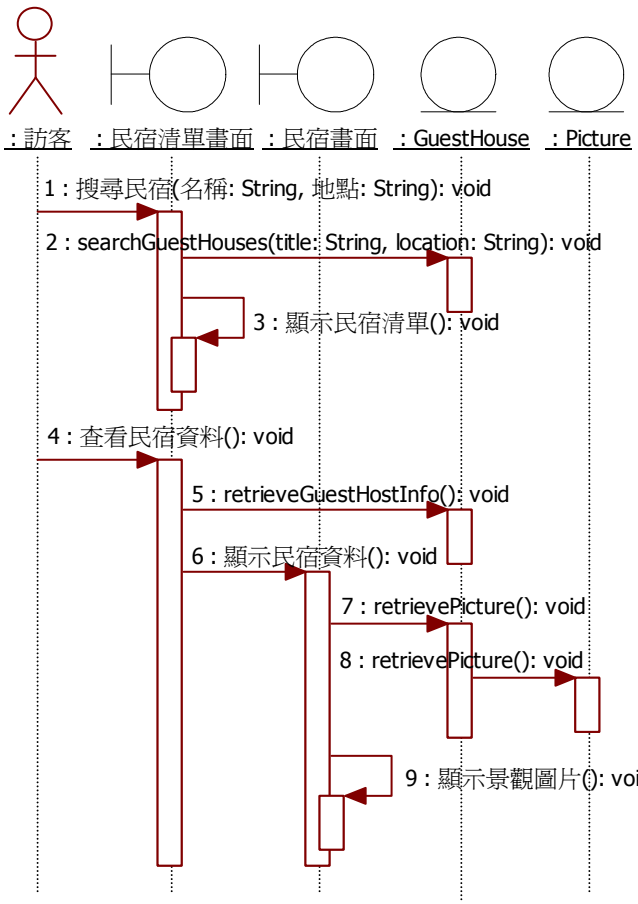

圖 5-46: 「查詢民宿資料」的循序圖(更新實體類別之後)

請參照圖 5-47，除了中文改英文外，額外加工的幾個重點如下：

1. 2 號訊息－爲求設計風格的統一，所以我們還是加上用例控制物件。

2. 3 號訊息—民宿實體類別回傳的是「一組」符合搜尋條件的民宿實體物件。
3. 7 號訊息—民宿實體物件回傳的是打包過的資料陣列。
4. 8 號訊息—不過，民宿的景觀圖片不放到 7 號訊息的資料陣列中。

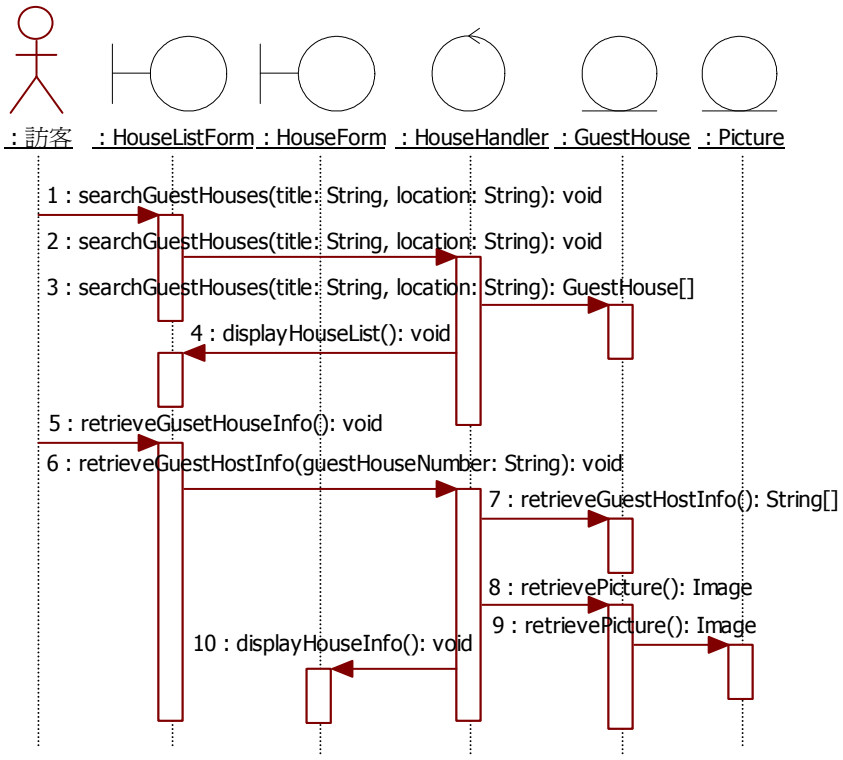

圖 5-47: 「查詢民宿資料」的循序圖

按照慣例，設計師要學著用手指和腦袋跑一下循序圖 5-47，看起來似

乎沒什麼問題，我們就直接列出更新的 BCE 類別了，如圖 5-48~49 所示。在「訂房」用例中，會用到民宿清單畫面的「挑選民宿」操作，所以此處我們就先保留起來，等到了訂房用例再來調整。

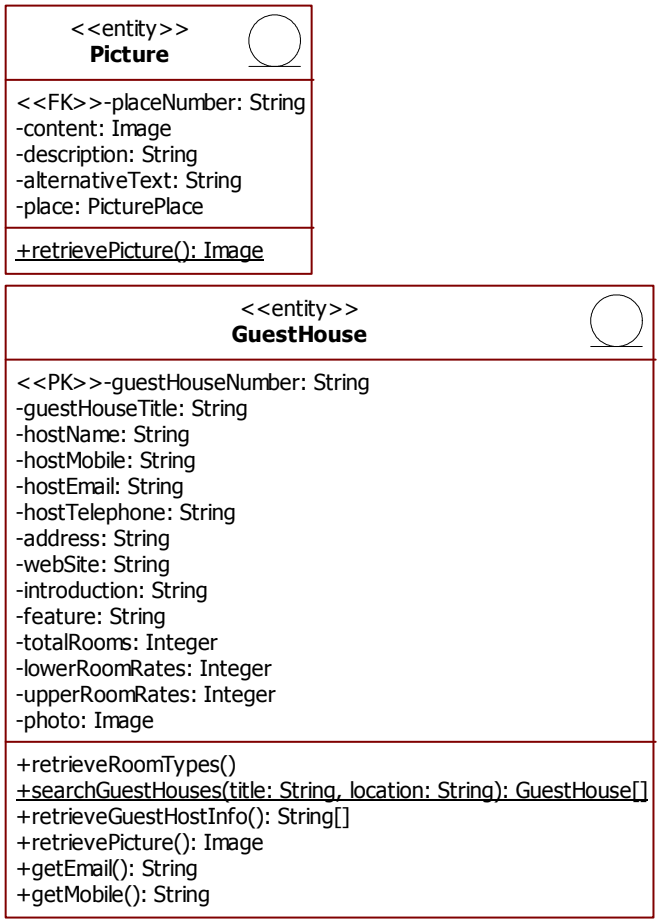

圖 5-48: 實體類別

圖 5-49: 邊界類別和控制類別

5.4.5 用例—查詢房型資料

通常一間民宿的房型不多，所以我們修改了用例敘述，去掉了搜尋房型的動作，而是直接查看所有房型資料，如表 5-9 所示。

用例	查詢房型資料		
啓動者	訪客	**支援者**	
主要流程 1. 訪客查看某一家民宿的所有房型，參考民宿畫面圖 5-50。 2. 系統顯示房型清單，包含房型名稱、床型、房間數、房價，參考			

房型清單畫面圖 5-51。

3. 訪客從中點選某一個房型，查看房型的細部資料。
4. 系統顯示房型資料，除了上述第 2 步驟的資料外，還額外包含房間設備、簡介、特色、景觀照片，參考圖 5-52 的房型畫面。

僞畫面

圖 5-50: 民宿畫面


房型清單
房型小圖
浪漫情人房
床型：超大雙人床
房間：4間
價位：NT$12,000
房型小圖
單身貴族房
床型：超大單人床2張
房間：2間
價位：NT$6,000
房型小圖
經典雙人房
床型：標準雙人床
房間：2間
價位：NT$6,000

圖 5-51: 房型清單畫面

圖 5-52: 房型畫面

表 5-9: 「查詢房型資料」的主要流程

接著，我們先來看圖 5-53，這是實體類別加工之後的循序圖。然後，我們一樣模仿前面的「查詢民宿資料」用例的循序圖，很快就獲得如圖 5-54 的設計了。

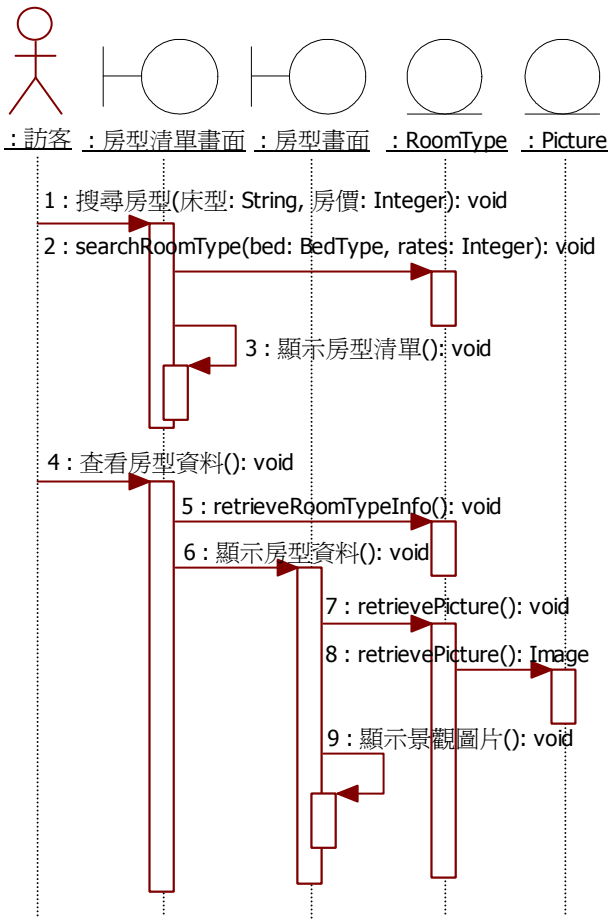

圖 5-53: 「查詢房型資料」的循序圖(更新實體類別之後)

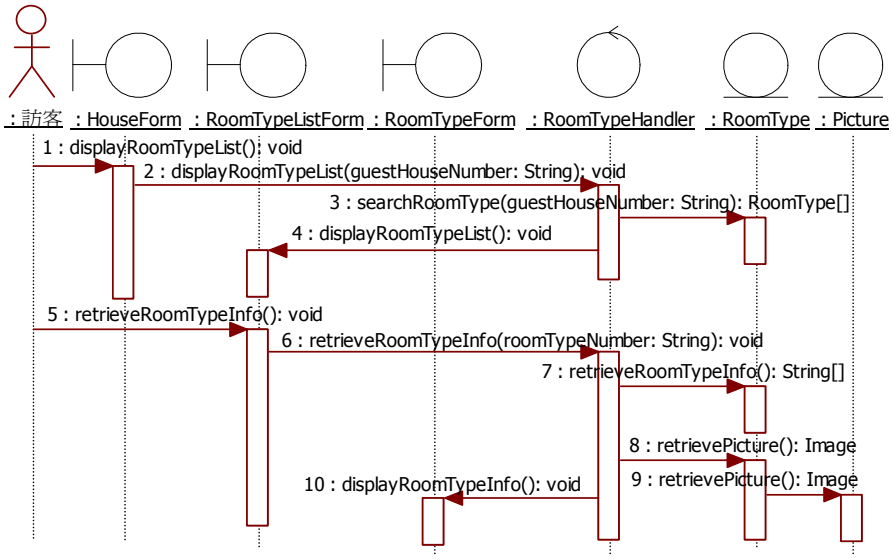

圖 5-54: 「查詢房型資料」的循序圖

最後，我們列出更新後的 BCE 類別，如圖 5-55~56 所示。

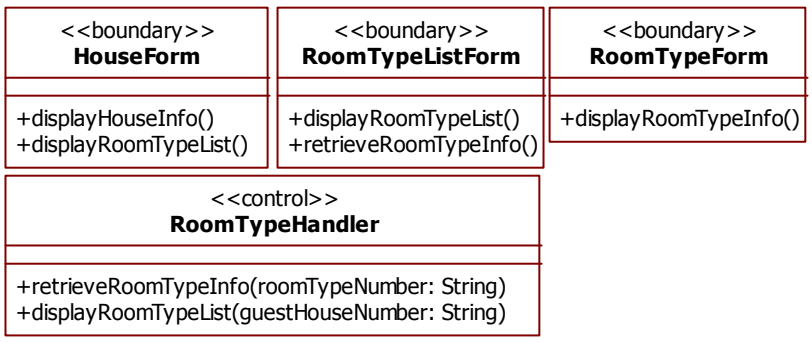

圖 5-55: 邊界類別與控制類別

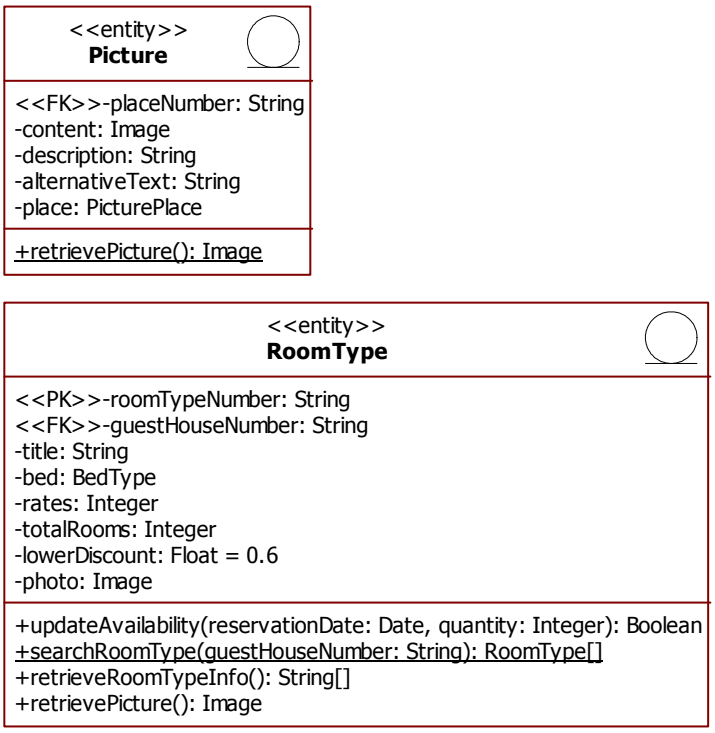

圖 5-56: 實體類別

5.4.6 用例—訂房

由於，「訂房」用例比較複雜，所以我們一開始就決定把它移到最後來談。這樣一來，一些共用的實體類別都已經在前頭先更新過了，所以可以簡化這邊的加工，同時也避免做白工。請先看到表 5-10，這是「訂房」用例原本的主要流程。

用例	訂房		
啓動者	會員	**支援者**	
相關用例	包含—發送電郵與簡訊通知		
主要流程			
1. 會員挑選一家民宿。			
2. 系統秀出這家民宿所有的房型名稱、床型、空房數和房價。			
3. 會員挑選預定的房型、房間數以及預訂日期。			
4. 系統顯示出訂房總價。			
5. 系統新增一筆訂房交易。			
6. 系統減少可預訂的空房數。			
7. 系統增加一筆未付訂交易筆數。			
8. 系統執行「發送電郵與簡訊通知」用例。			
9. 系統秀出交易代號、訂金與總價。			
10. 系統提醒會員需要 48 小時內付訂金。			

表 5-10: 「訂房」的主要流程

至於，圖 5-57 是分析階段的循序圖，跟表 5-10 已經不同步，因爲我們已經將發送電郵與簡訊通知的部分獨立出來了。

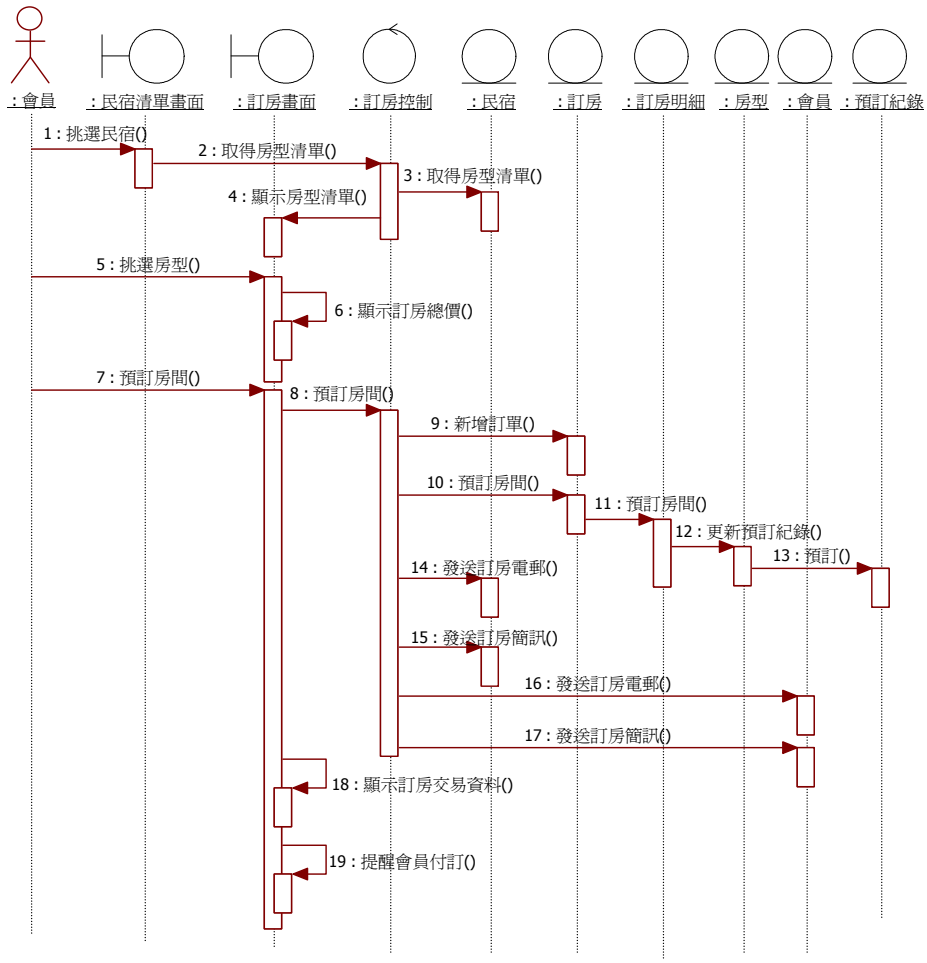

圖 5-57: 分析階段的「訂房」循序圖

所以，在更新圖 5-57 之際，我們發現可能需要大幅度修改，所以設計師找了分析師一塊來討論，幾個修改重點如下：

1. 1~5 號訊息—圖 5-57 的五條訊息可以刪掉，因爲我們打算從圖 5-58 的房型僞畫面當訂房用例的起頭，所以不再需要挑選民宿、取得房型清單、顯示房型清單、挑選房型，這一段流程了。

圖 5-58: 房型畫面

2. 民宿清單畫面物件－也因此，民宿清單畫面物件可以刪去，同時刪去之前保留下來的「挑選民宿」操作，更新之後的民宿清單畫面類別如圖 5-59 所示。

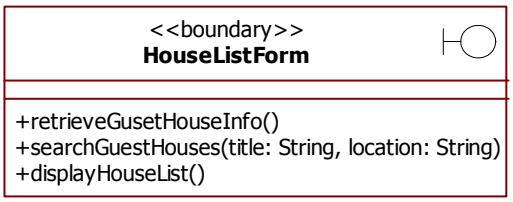

圖 5-59: 民宿清單畫面類別

3. 房型畫面物件－以房型畫面物件入口，加上「查看訂房紀錄」(checkReservationRecord)操作，如圖 5-60 所示。

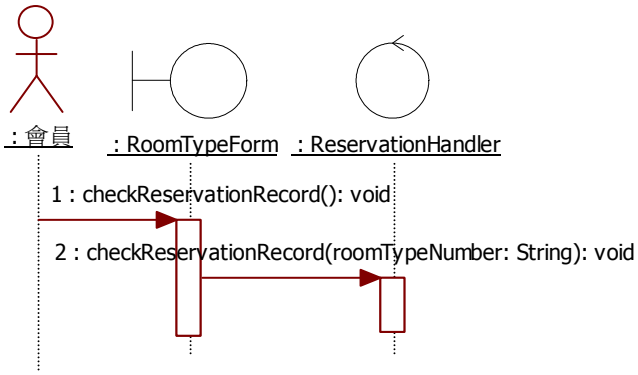

圖 5-60: 查看訂房紀錄

4. 查看訂房紀錄－爲了查看訂房紀錄，所以我們需要用到房型和預訂紀錄實體物件，如圖 5-61 所示。

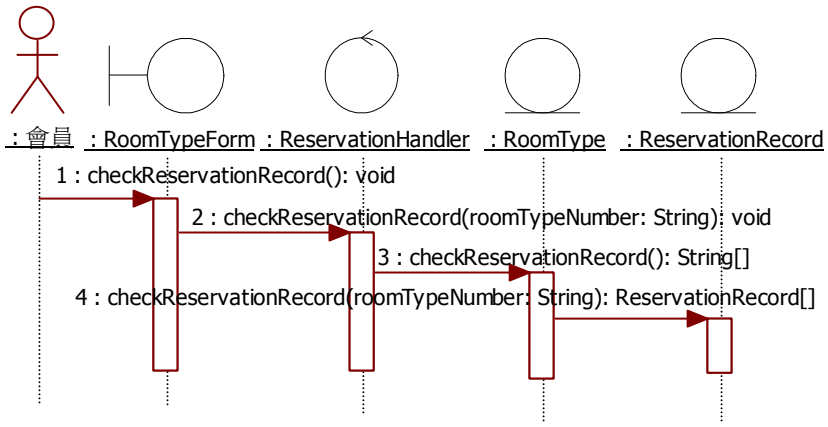

圖 5-61: 查看訂房紀錄

5. 預訂紀錄僞畫面－因爲需要秀出預訂紀錄給會員看，所以我們設計了預訂紀錄僞畫面，如圖 5-62 所示。在預訂紀錄畫面中，如果當日房間已滿，或者任何無法被訂房的原因，在該日的空格中，會是空白，否則就會出現可以選擇訂幾間房間的下拉選單，以及訂房的按鈕。

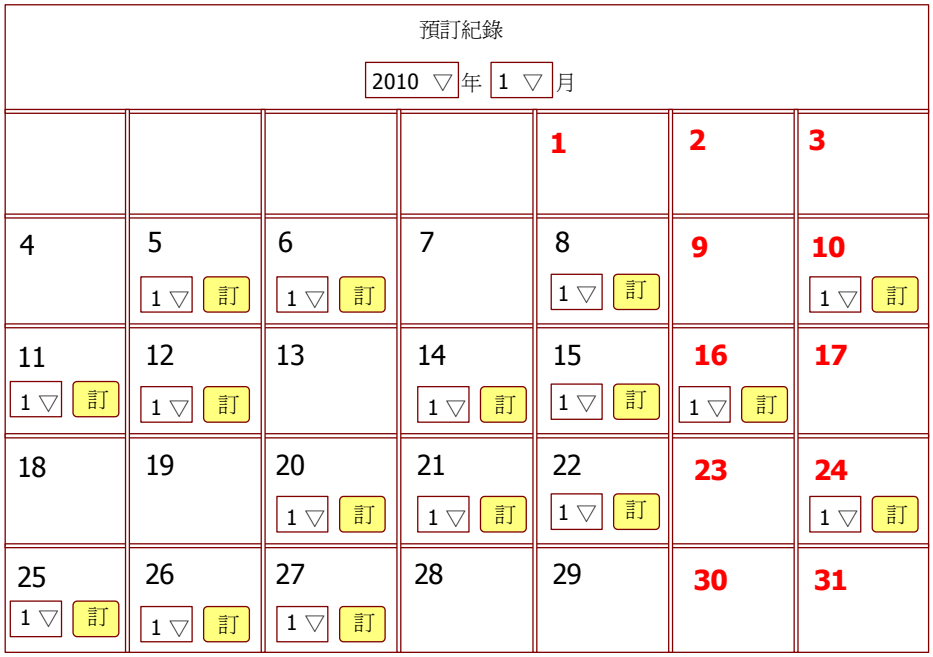

圖 5-62: 預訂紀錄畫面

6. 預訂紀錄畫面物件—所以，相對的我們還需要一個「預訂紀錄畫面」(ReservationRecordForm)邊界物件，來顯示預訂紀錄資料，如圖 5-63 所示。

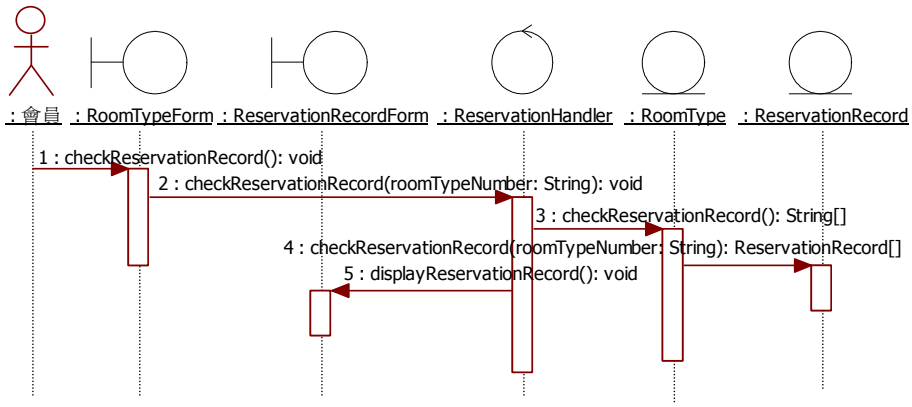

圖 5-63: 預訂紀錄畫面物件

7. 計算房價—原本房型實體類別裡頭有個「計算房價」，但是被我們刪掉了，所以這邊會需要增加個「計算房價」(figureOutRate)的操作，如圖 5-64 所示。

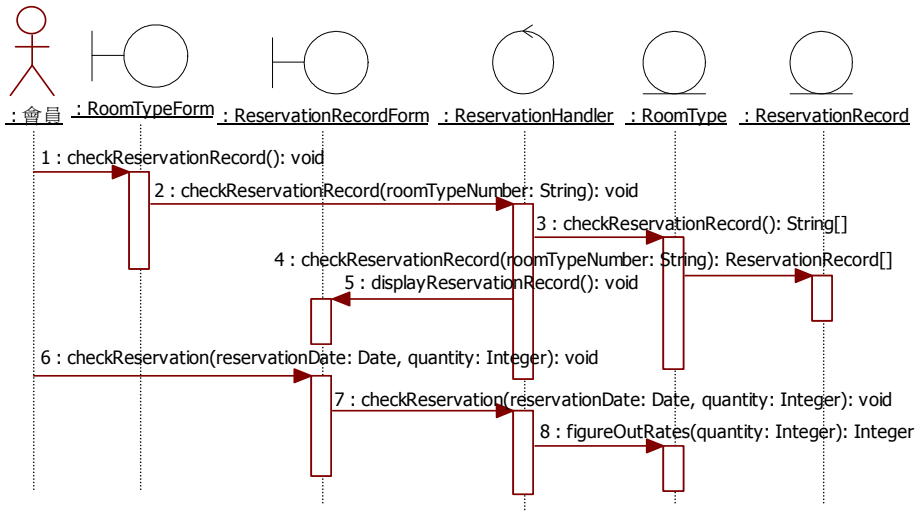

圖 5-64: 計算房價

8. 訂房資訊僞畫面—因爲有了預訂紀錄畫面，而且會員將由此畫面來預訂房間，因此原先圖 5-57 中的訂房畫面物件就不需要了。然後，我們設計個訂房資訊畫面來呈現訂房資訊，如圖 5-65 所示。

圖 5-65: 訂房資訊畫面

9. 訂房資訊畫面物件—所以，同樣會需要一個對應的「訂房資訊畫面」(ReservationInfoForm)邊界物件，來顯示訂房資訊，如圖 5-66 所示。

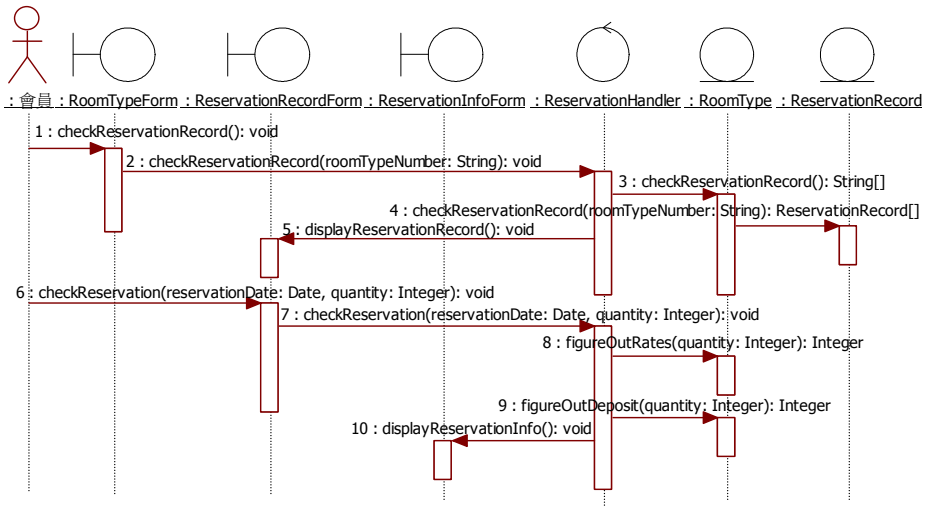

圖 5-66: 訂房資訊畫面物件

10. 訂房一接著，會員可以透過訂房資訊畫面物件，確認訂房資訊，而且系統也可以真正產生訂房交易了，如圖 5-67 所示。比較特別的是，在訂房實體物件的 14 號訊息中，會回傳預訂序號。

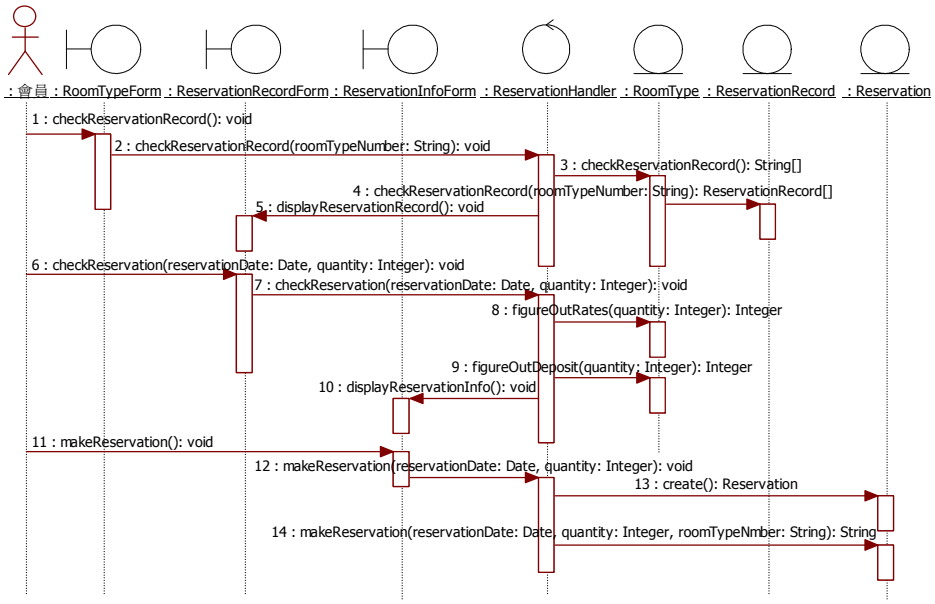

圖 5-67: 訂房

11. 查看預訂紀錄—這張循序圖實在太複雜了，所以我們可以使用「片段」(frame)大方框，把第 1~10 號訊息框住，另外命名爲「查看預訂紀錄」(checkReservation)片段，如圖 5-68 所示。然後，新增另一張循序圖，引用這個片段，繼續往下設計循序圖，如圖 5-69 所示。特別看到原先的互動片段，片段名稱前會標示「sd」字眼，代表一個互動片段，在引用的循序圖中，則會標示「ref」字眼，代表引用一個片段。我們在下一章循序圖主題中，會仔細談到片段的概念，這邊就先拿來用，不多加解釋了。

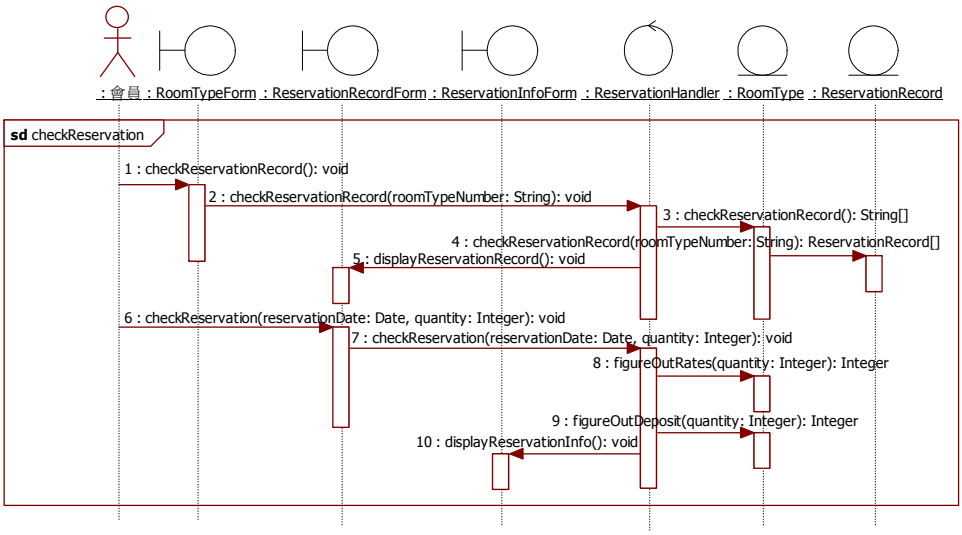

圖 5-68: 「查看預訂紀錄」片段

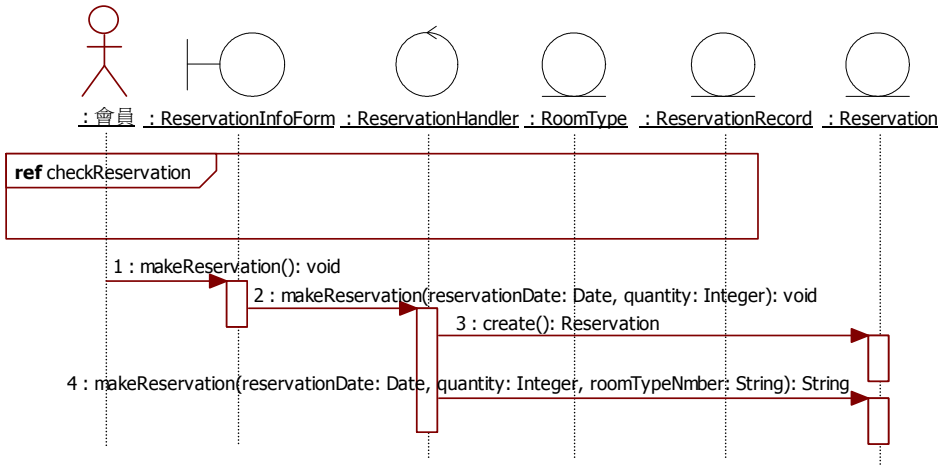

圖 5-69: 「查看預訂紀錄」片段

12. 通知控制物件—原先圖 5-57 中的 14~17 號訊息這四條訊息可以刪掉，也因此民宿物件和會員物件暫時沒有作用了，所以一併刪去。但是，必須增加「通知控制」(NotifyHandler)物件，以及從「訂房控制」(ReservationHandler)物件發送給它的「notify」訊息，如圖 5-70 所示。

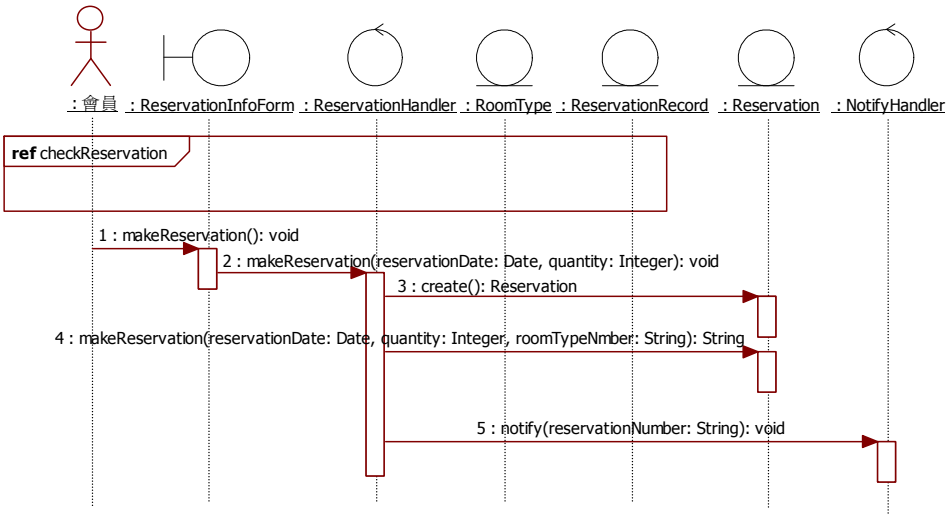

圖 5-70: 通知控制物件

13. 顯示訂房成功資訊—原先圖 5-57 中的 18~19 號訊息的顯示訊息，我們改由「訂房資訊畫面」物件來顯示，如圖 5-71 所示。接著，我們同樣把通知和顯示訂房的互動，先框出來、再引用，如圖 5-72 所示。這邊這麼做只是爲了降低圖面的複雜度，因爲前面的「預訂」(makeReservation)動作，其實還未完成呢！

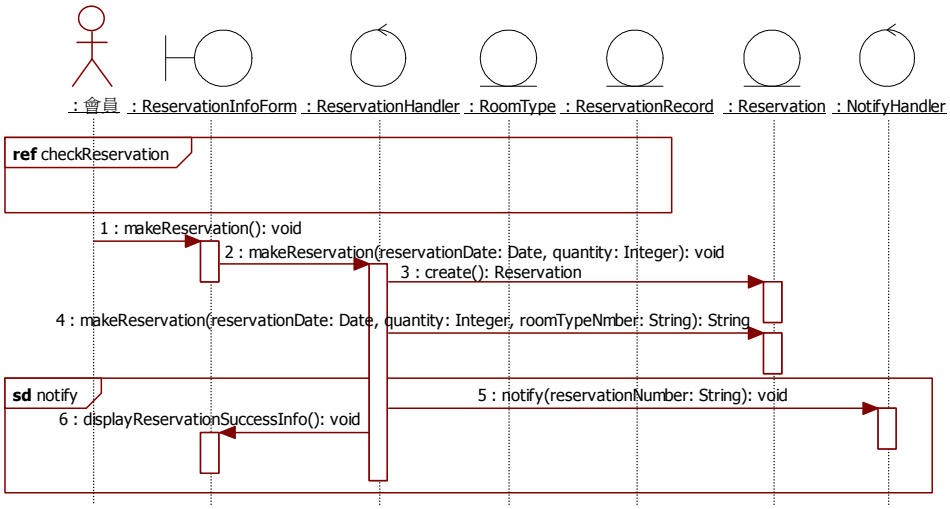

圖 5-71: 顯示訂房成功資訊

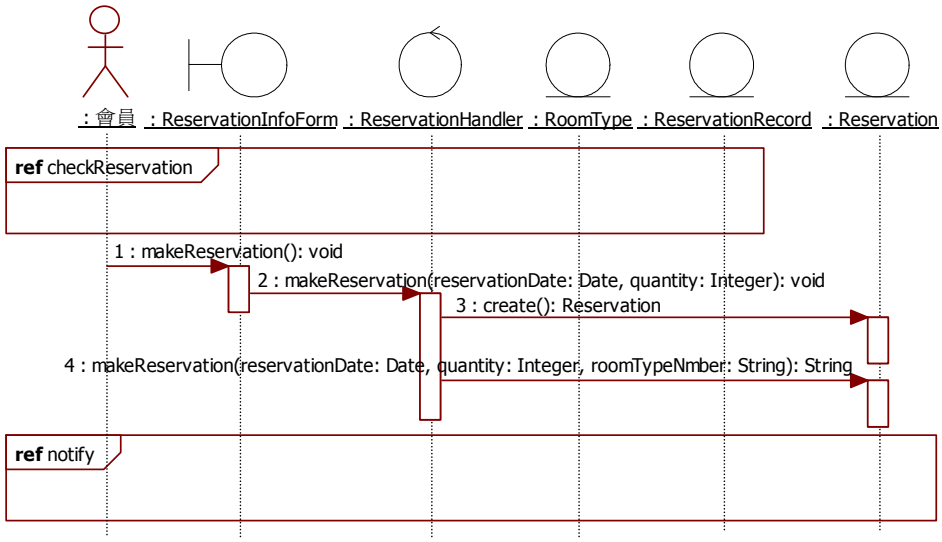

圖 5-72: 通知片段

現在，我們可以來好好處理預訂的部分了。請先看到圖 5-73 的實體類別圖，原先我們構想一個會員可能一次預定多間不同房型的房間，可是大部分的情況卻是一次預訂一個房間。

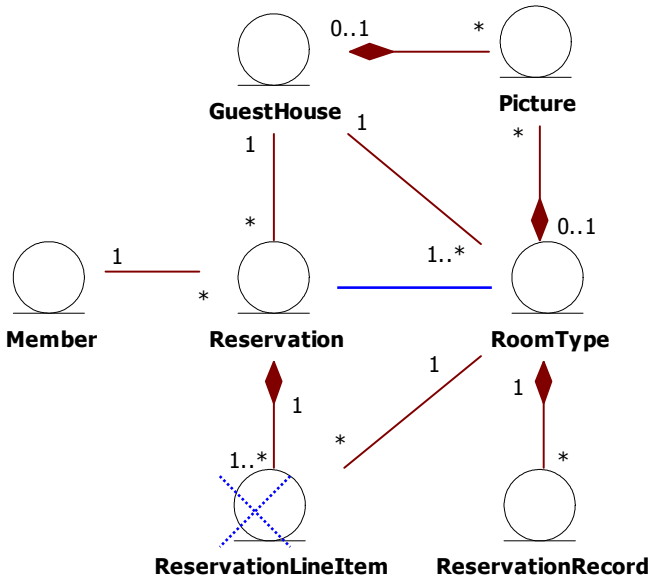

圖 5-73: 刪除訂房明細實體類別

如果，按照原先的構想，設計起來比較複雜，於是我們改成一次預訂一種房型，當然同一種房型的房間可以一次預訂多間。換言之，只要一次預訂一種房型，這樣的作法就簡單多了。因此，我們可以刪掉原先的訂房明細(ReservationLineItem)，但是在訂房和房型之間必須加上結合關係，形成如圖 5-74 的靜態結構。

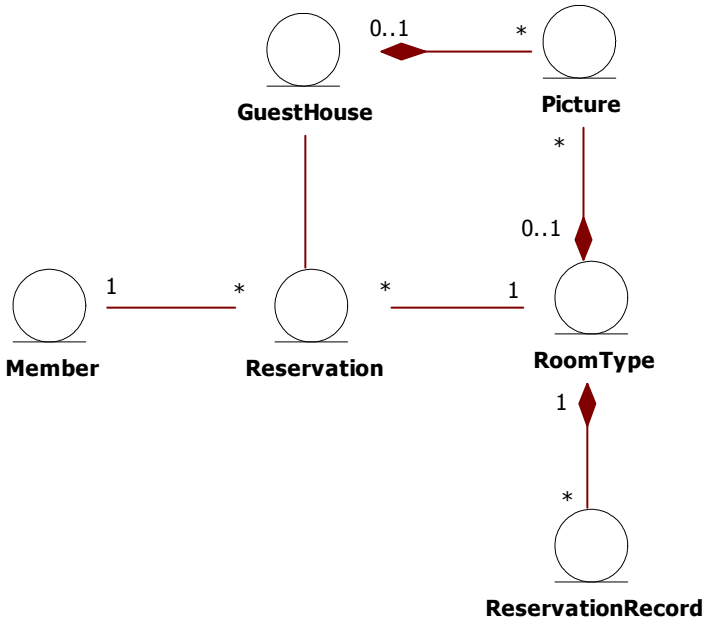

圖 5-74: 修改後的實體類別圖

當然，實務上並不總是能夠簡化結構，因爲可能涉及到領域概念的限制，或者是客戶的不允許。不過，在客戶沒有意見的情況下，我會建議採用能用簡單的方法解決問題，就用簡單的方法，這樣的開發成本和日後的維護成本最省。

在決定把訂房明細實體類別刪掉之際，我們必須先將訂房明細裡頭的部分屬性和操作併回訂房，如圖 5-75~76 所示。主要將訂房數量放到訂房實體類別中，還有爲了關聯房型會用到的外鍵，兩者一併搬回，其餘就直接刪除了。

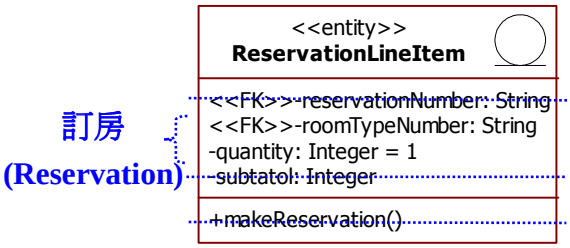

圖 5-75: 訂房明細內的屬性併回訂房

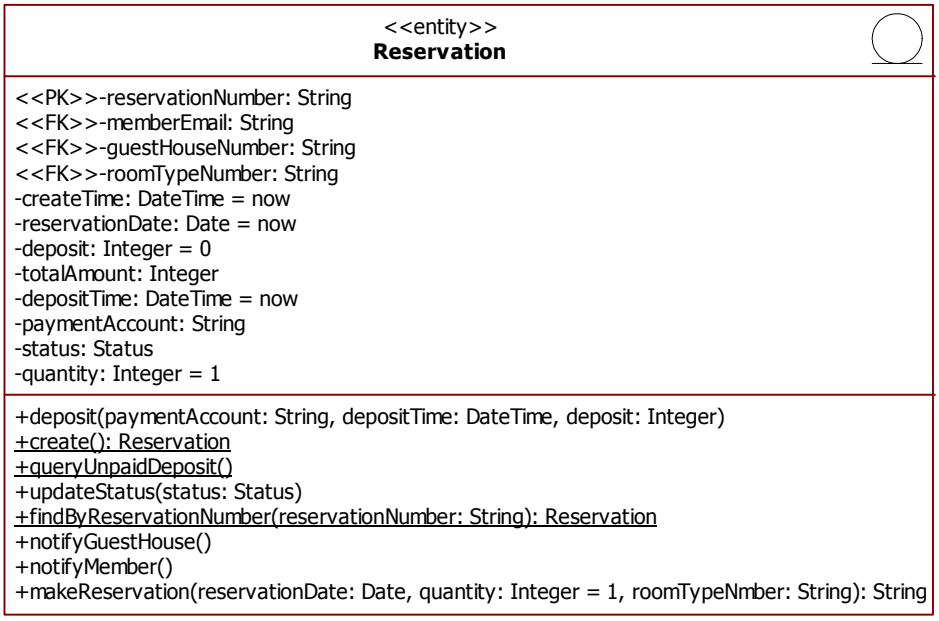

圖 5-76: 訂房實體類別

好了，在修改完實體類別圖之後，我們就可以繼續把循序圖完成了，如圖 5-77 所示，增加了第 5~6 號訊息更新空房數量。

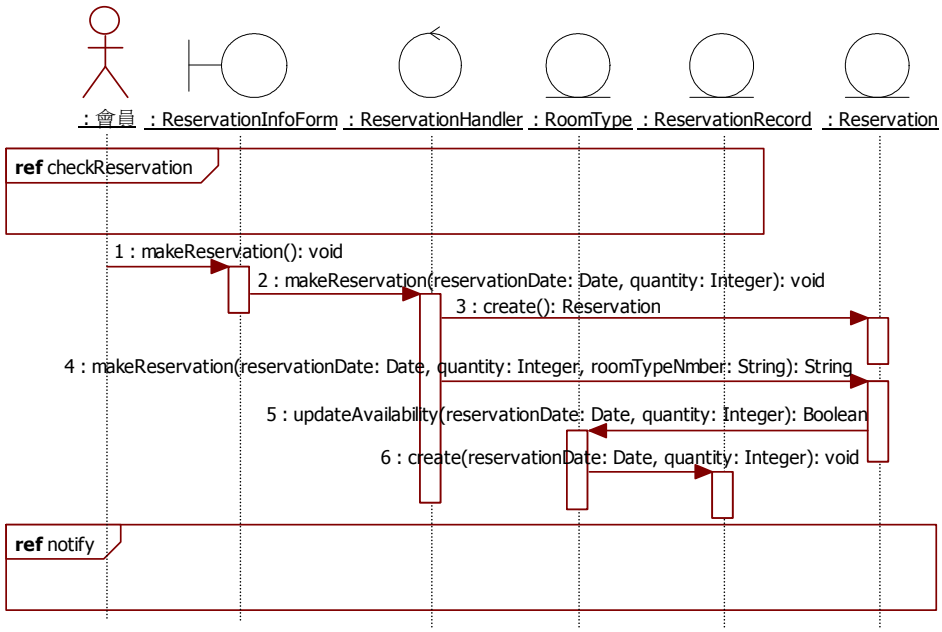

圖 5-77: 訂房實體類別

最後，我們根據原先的用例敘述，重新修改出「訂房」用例的主要流程，如表 5-11 所示。

用例	訂房		
啓動者	會員	**支援者**	

相關用例	包含—發送電郵與簡訊通知

主要流程

1. 會員挑選預定的房型，查看其訂房紀錄，參考房型畫面如圖 5-58 所示。
2. 系統顯示出房型當月的預訂紀錄，參考預訂紀錄畫面如圖 5-62 所示。
3. 會員選定某一天預訂日期、房間數量。
4. 系統顯示出訂房資訊，包含有：房型名稱、預訂日期、預定房數、計算出訂房總金額、訂金。
5. 會員確認訂房資訊後，決定要訂房。
6. 系統新增一筆訂房交易。
7. 系統減少可預訂的空房數。
8. 系統增加一筆未付訂交易筆數。
9. 系統產生交易序號。
10. 系統執行「發送電郵與簡訊通知」用例。
11. 系統顯示出訂房成功資訊，包含有：房型名稱、交易序號、訂金，以及請會員於 48 小時內付訂金的訊息。

僞畫面

1. 請回頭參考圖 5-58 的房型畫面。
2. 請回頭參考圖 5-62 的預訂紀錄畫面。
3. 請回頭參考圖 5-65 的訂房資訊畫面。

表 5-11: 「訂房」的主要流程

按照慣例，我們還是要配合用例敘述，先用手指和腦袋跑一下循序圖，發現漏了用例敘述中的第 8~9 步驟，所以我們修改了循序圖。同時，爲了便於閱讀，我們調整了物件擺放的順序，讓訊息可以盡量由左向右射出，這樣配合左向右的訊息名稱，閱讀起來會比較順暢。請看到圖 5-78，我們增加了第 7~8 號訊息。

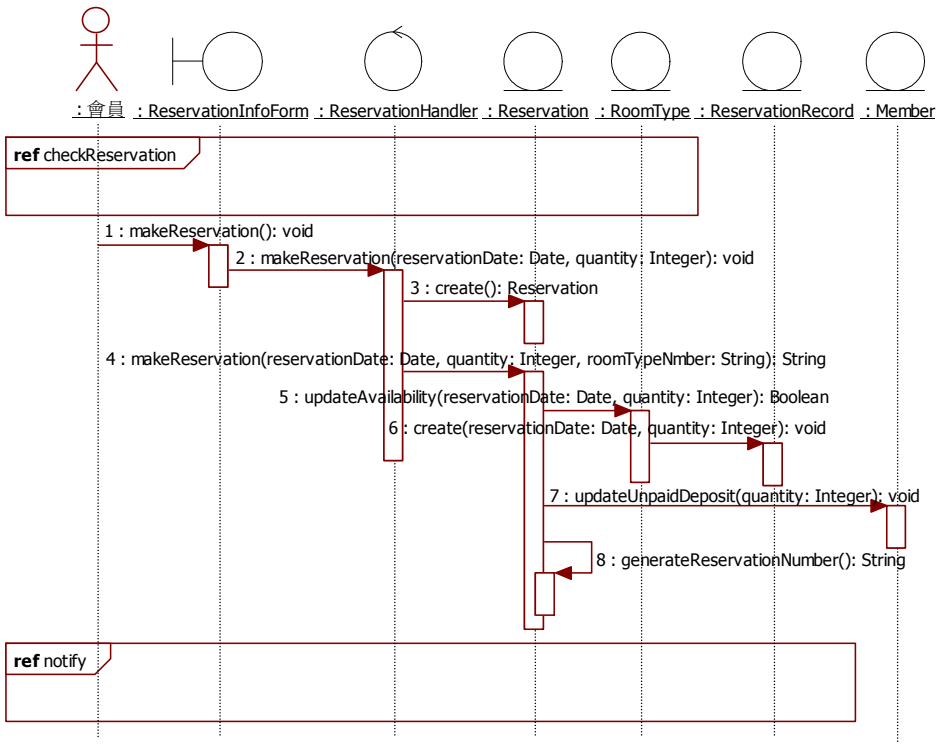

圖 5-78: 「訂房」用例的循序圖

再跑一次循序圖，沒有發現新的問題，所以我們就先列出更新後的 BCE 類別了，如圖 5-79~82 所示。

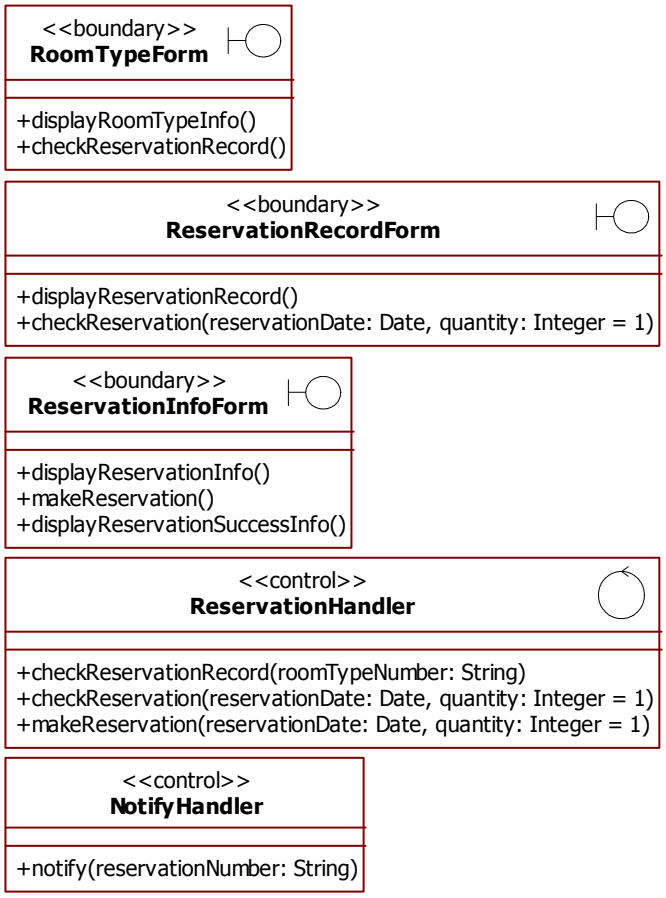

圖 5-79: 邊界類別與控制類別

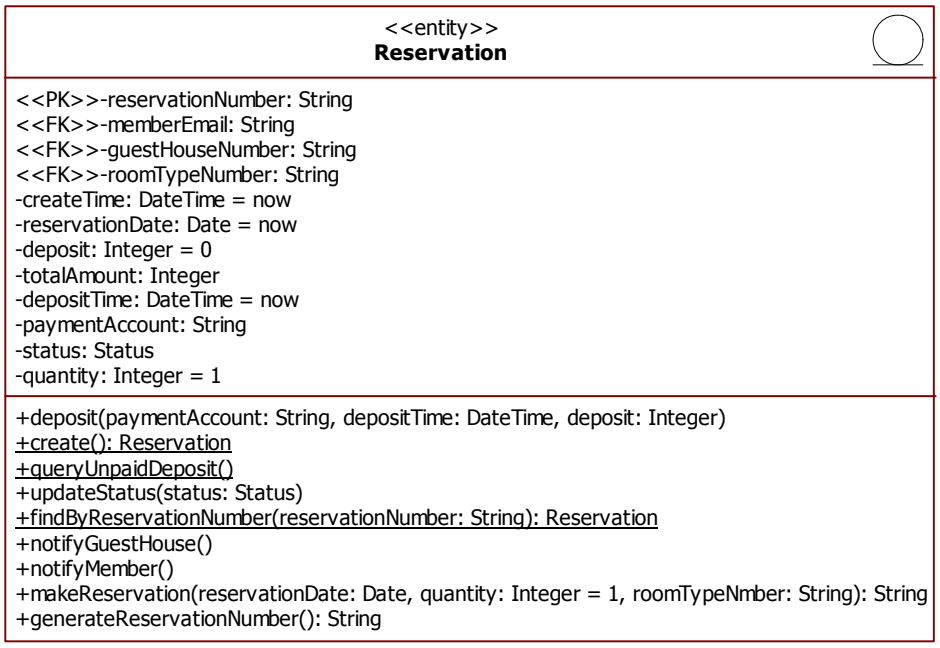
<<entity>>
Reservation
<<PK>>-reservationNumber: String
<<FK>>-memberEmail: String
<<FK>>-guestHouseNumber: String
<<FK>>-roomTypeNumber: String
-createTime: DateTime = now
-reservationDate: Date = now
-deposit: Integer = 0
-totalAmount: Integer
-depositTime: DateTime = now
-paymentAccount: String
-status: Status
-quantity: Integer = 1
+deposit(paymentAccount: String, depositTime: DateTime, deposit: Integer)
+create(): Reservation
+queryUnpaidDeposit()
+updateStatus(status: Status)
+findByReservationNumber(reservationNumber: String): Reservation
+notifyGuestHouse()
+notifyMember()
+makeReservation(reservationDate: Date, quantity: Integer = 1, roomTypeNmber: String): String
+generateReservationNumber(): String

圖 5-80: 訂房實體類別

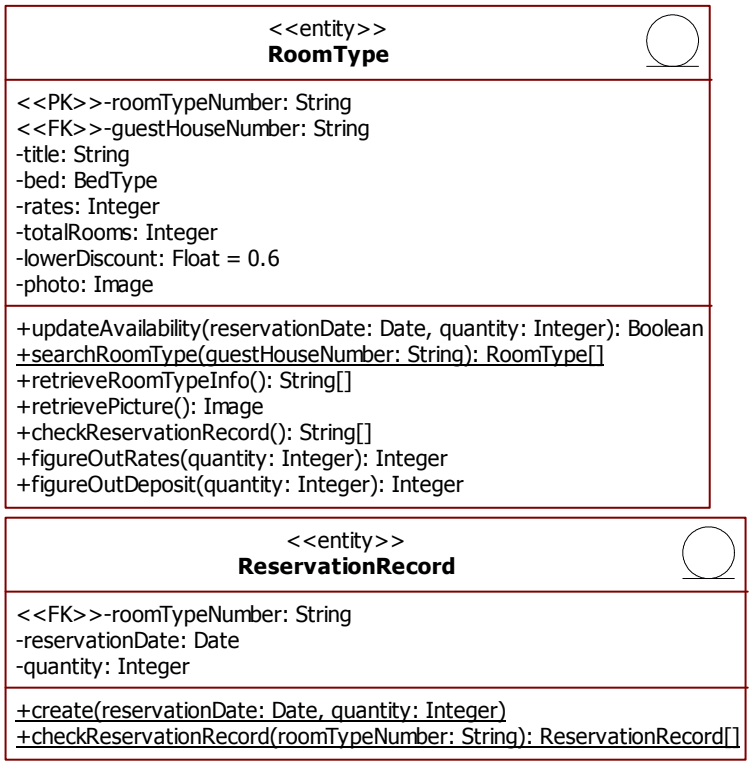

圖 5-81: 房型與預訂紀錄實體類別

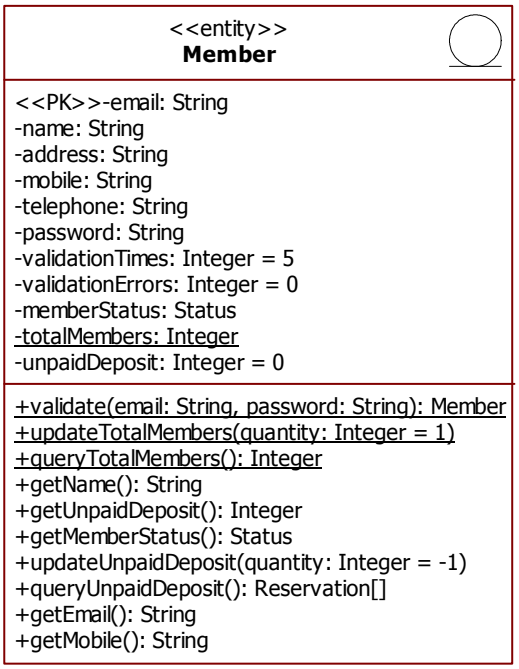

圖 5-82: 會員實體類別

5.5 後話

在撰寫這一章時，冒出許多想法，不容易有系統的安置在正文中，所以我把它們列在此處：

- 想要重用或共用類別或流程，向來都就不是一件簡單和廉價的事情，總是需要多次的修修改改，修改、修改、再修改，才能獲得一點點可以重用或共用的好設計。
- 複雜的用例，不一定要先做，如果先做了，幾乎都會需要重做。

但是，別害怕重做，過程中獲得的技能和練習，會因爲一次次的重做，而獲得更多。

- UML 可以提高溝通品質，但無法完全取代人跟人之間的直接溝通。所以，設計師銜接分析師的文件後，通常兩者免不了還是需要直接溝通的，但是比較能夠聚焦，溝通的效果品質也較佳。
- 聰明的方法，不一定是複雜的，如果可以用簡單的方式行事，開發和維護成本會比較低，這或許是聰明方法的另一種定義。所以，實體類別圖可以簡單，就別讓它複雜了。
- 來來回回的修修改改，是最正常不過的事情了，別以爲是自己的能力有問題。當然，持續提升自己的專業能力，一直是刻不容緩的事情。
- 千萬別認爲用例敘述沒有用，事實上，因爲先有簡單的用例敘述，才會一步步引導設計出複雜的循序圖。
- 僞畫面功用很大，但是畫面物件是隨時會被汰換的，別忘了重點在藉此發展出控制物件和實體物件。

6
(D3)循序圖

※※

6.1 按圖施工

6.2 設計師必學元素

6.3 民宿聯合訂房系統

6.4 UML 囈語

6.1 按圖施工

經過了前面幾章的洗禮，您肯定發現到，循序圖對於思考用例的運作細節有很大的幫助，甚至可以說，如果循序圖設計的完善的話，想要讓程序員按照這些 UML 設計圖來按圖施工，絕對不是問題。

當然，類別圖、用例圖文和循序圖缺一不可，而且在開發時，必須學習且習慣盡快進入下一款圖，以便用另一個角度來修正之前的產出。對於修正類別圖和用例圖文而言，循序圖絕對功不可沒，但是也別因此而誤認爲類別圖和用例圖文沒用喔。

所以，在這本書的最後一章，我們會交給設計師幾項關於「循序圖」的概念，讓設計師可以依照下列重點，進一步加工循序圖：

1. 訊息參數—設計師產出的循序圖愈細膩，就愈能夠讓程序員按圖施工，所以設計師對循序圖中的訊息參數要在特別加工一下，讓它可以儘量貼近實作。
2. 互動與引用—其實，在上一章中，我們已經使用過「片段」(fragment)了，在本章中，我們會再細談相關概念。
3. 互動操作子—搭配片段框架，設計師可以針對這個互動片段執行迴圈、並行等等的控制操作。
4. 操作內部互動—最後，設計師要注意操作內部互動的一致性。當物件接收到訊息、執行某一個操作時，物件會發送訊息給其他物件，像這樣一個操作範圍內的小小互動片段，如果出現在不同循序圖中，也都應該保持一致。以圖 6-1 爲例，通知控制(NotifyHandler)物件接收到「通知」(notify)訊息後，它會依序發出三個訊息給訂房、民宿和會員實體物件。假設在另一張循序圖中，也有用到通知控制物件，而它同樣接收到通知訊息的話，也

應該會依序發出三個訊息給訂房、民宿和會員實體物件。設計師必須確保不同循序圖中，針對相同物件的相同訊息，其內部的互動是一致的。

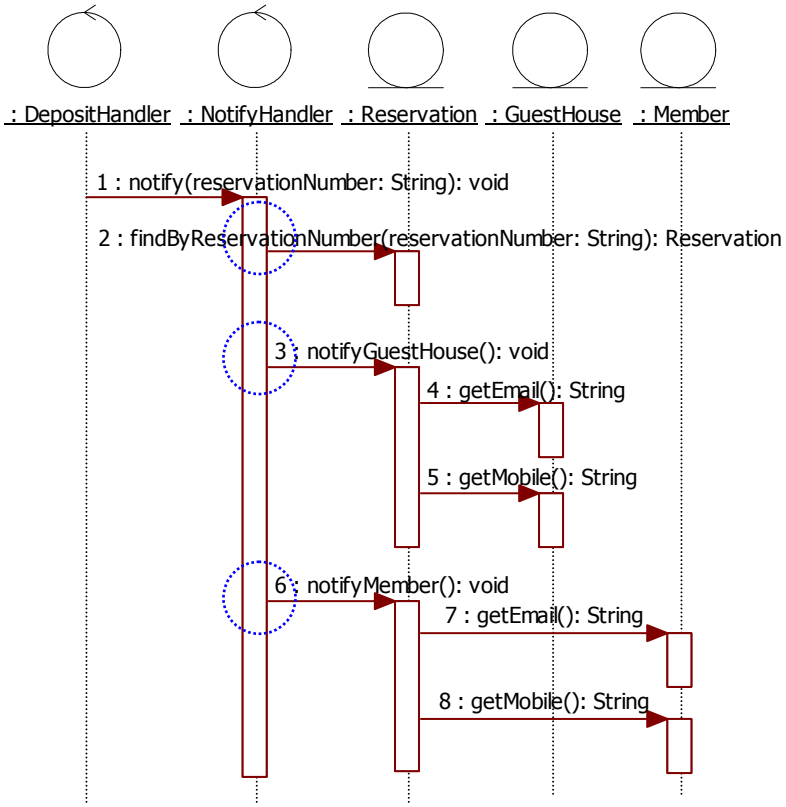

圖 6-1: 操作內部的互動

6.2 設計師必學元素

「片段」(fragment)是 UML2 的新元素，它改善了循序圖一直令人詬病的兩個現象：

1. 循序圖無法表達過於複雜的控制流程，可是偏偏有許多設計細節，都需要藉由循序圖來傳達。
2. 互動設計又多又雜，許多互動片段都可以重用，可是 UML1 就是沒提供重用的機制。所幸，UML2 提供了片段的圖示，讓設計師可以鎖定某一互動片段，進行局部的重用或更複雜的設計。

因此，雖然在 OCUP/UML 認證的分級上，片段屬於中級概念。但是，由於它在實務上的實用性高，所以我還是建議設計師學習這個新概念。

6.2.1 互動與引用

「片段」(fragment)的概念相當簡單，透過一個有名稱的大方框，框住幾條連續的訊息，就可以形成一個我們可以指稱的「互動單元」(interaction unit)。然後，我們就可以詳加說明這個互動片段，或者針對整個片段做特殊的操作。

請看到圖 6-2 的大方框，大方框左上角的五角型內標示「sd」是循序圖(sequence diagram,sd)的縮寫，sd 後面標示互動片段的名稱爲「查看預訂紀錄」(checkReservationRecord)。

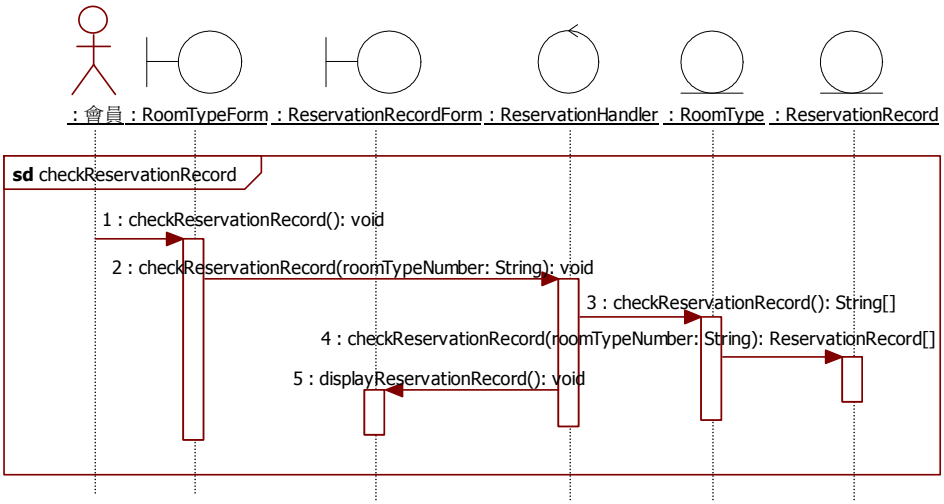

圖 6-2: *互動片段*

以往我們如果需要重用用例中的某些流程時，可能會透過包含關係或擴充關係，將可以重用的流程切割出去，這當然不是個好方法。現在，我們有了互動片段，就可以框住任何一段可以重用的互動片段，不需要再胡亂切割用例了。

如果，我們需要重用某一個互動片段時，可以透過「ref」關鍵字，標示出「引用片段」(InteractionUse)。請看圖 6-3，我們在這張循序圖中引用了圖 6-2 中的查看預訂紀錄互動片段。

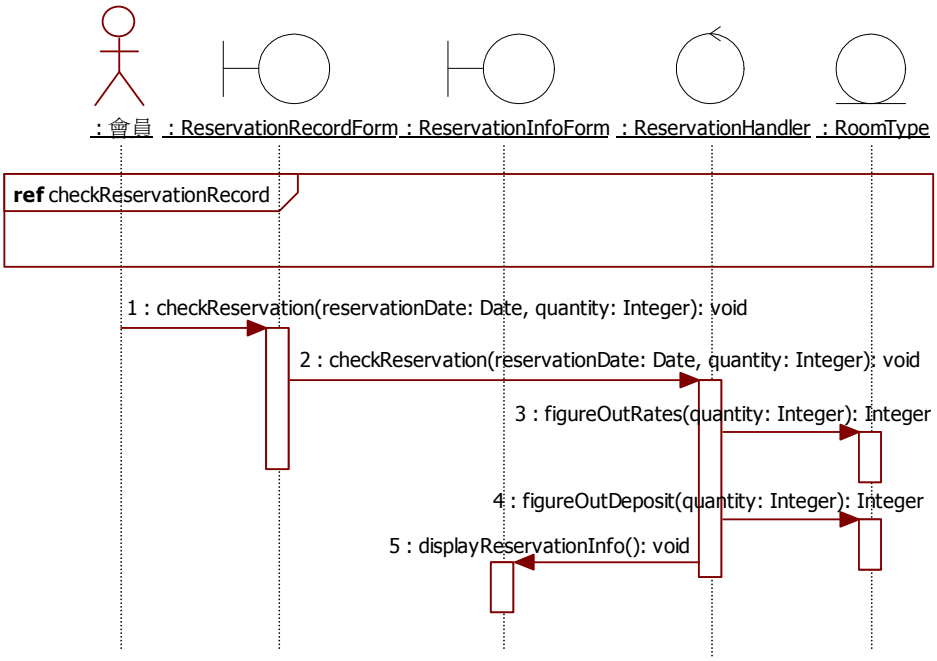

圖 6-3: 引用片段

包含 sd/ref 在內，UML2 一共定義了十多個「互動操作子」(interaction operator)，用來控制互動片段。在接下來的次小節中，我們會提到另外 4 個比較實用的互動操作子，分別爲：迴圈(loop)、選擇(option,opt)、替代(alternative,alt)和並行(parallel,par)。

6.2.2 迴圈片段

針對一個互動片段，執行「迴圈」(loop)控制，這是十分常見的現象。在訂房系統中，會員只能累積 5 次的登入錯誤，這時就可以放個 1~5 次的迴圈控制。請看圖 6-4 中，關鍵字 loop 後面放置次數小括號，先放最少迴

圈數，再放置最多迴圈數。

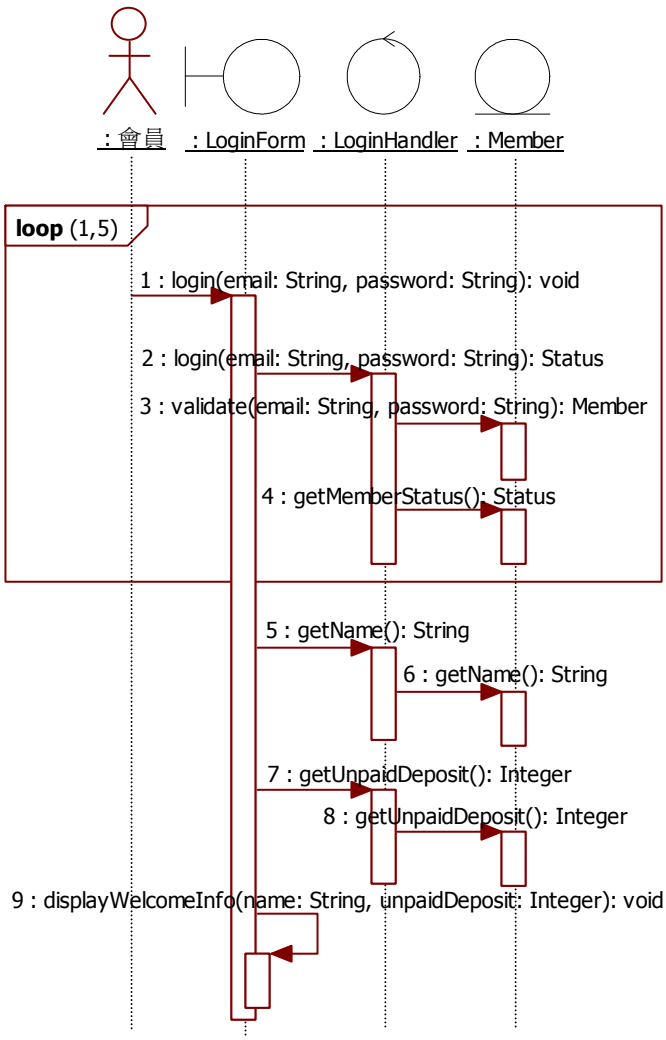

圖 6-4: 迴圈片段

6.2.3 選擇片段

設計師可以標示關鍵字「opt」，代表「不一定」得執行的選擇片段，譬如「發送電郵與簡訊通知」用例中，我們讓會員勾選不發送通知給自己，僅發送通知給民宿主人就好，如圖 6-5 所示。

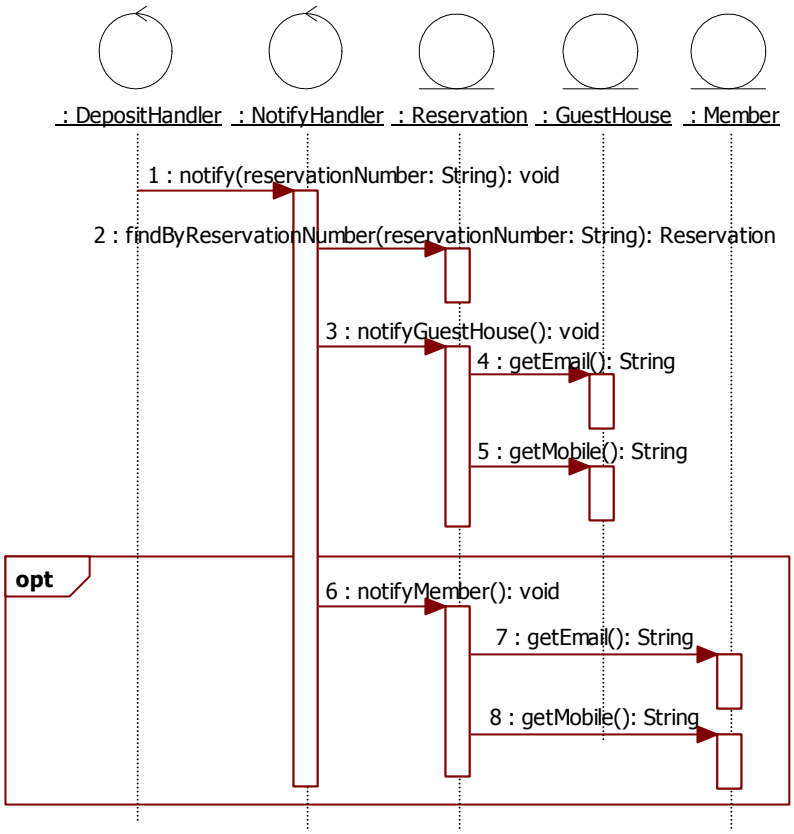

圖 6-5: 選擇片段

6.2.4 替代片段

選擇片段像程式語言中的 if 子句，可能執行，也可能不執行。但是，有時候，我們想表達多選一，像 case 子句的操作控制時，可以改用標示「alt」關鍵字的替代片段。請看圖 6-6，代表發送電郵通知，或者發送簡訊通知，擇一執行即可。

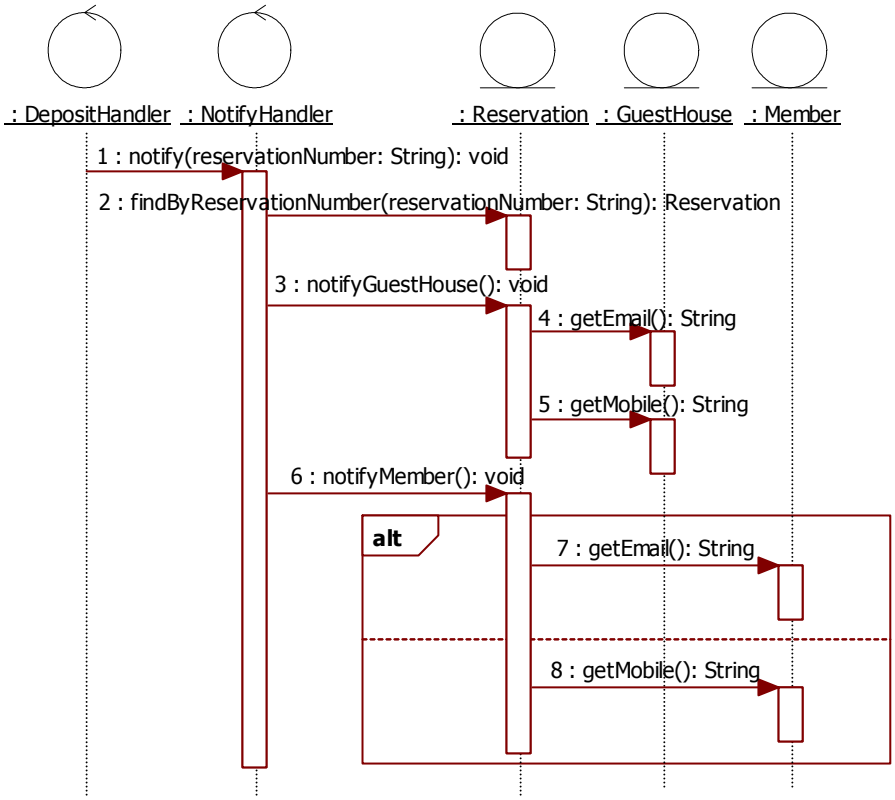

圖 6-6: 替代片段

6.2.5 並行片段

通常，循序圖的訊息的發送順序是由上而下依序，如果想表達並行發送的話，可以標示關鍵字「par」表示多個互動片段可以並行執行，不需要依序。請看圖 6-7，設計師可以用並行片段，表達發送通知給民宿主人和會員，可以並行，不需要有先後次序。

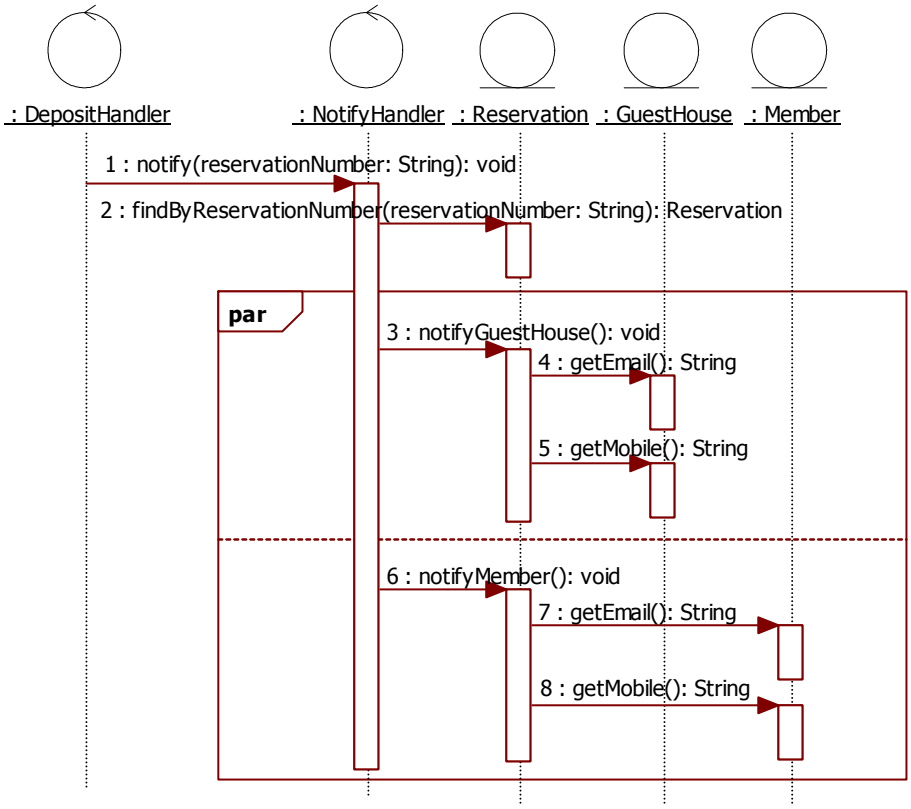

圖 6-7: 並行片段

6.3 民宿聯合訂房系統

首先，我們先來確認一下，目前爲止，有進行分析設計的用例，如圖 6-8 所示。

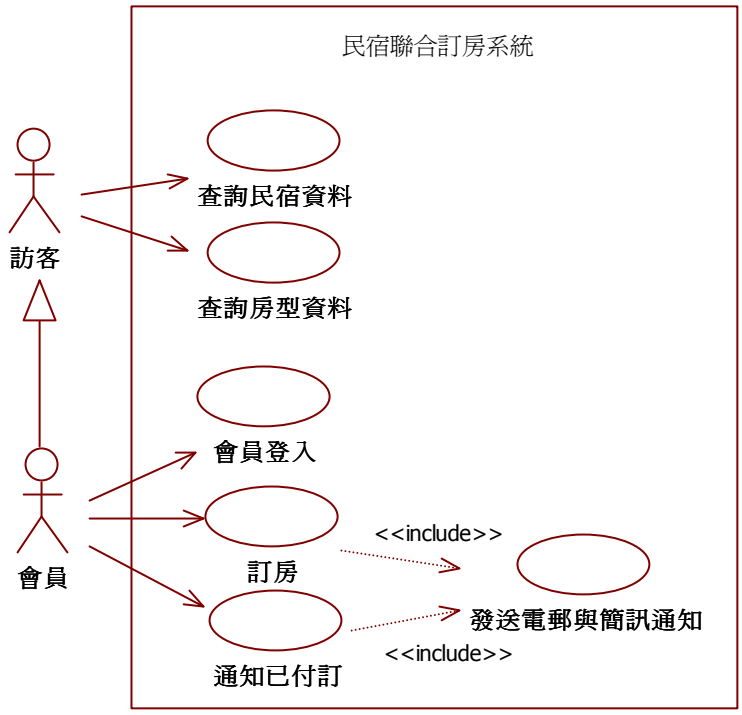

圖 6-8: 用例圖

這是設計師最後一次回顧並加工用例了，之前已經加工了好幾次，應該不會再出現切分用例、找到新用例或新類別之類比較大的變動。此處，設計師就是針對操作參數的部分再修正一下，還有操作內部互動的一致性

也要在注意一下，再來就是看看需不需要設置一些互動片段，然後就可以交給程序員按圖施工了。

6.3.1 用例—會員登入

還記得用例敘述主要分爲，主要流程和替代流程兩部分；通常，分析師和設計師都會先處理主要流程，畢竟要是連主要流程都沒辦法處理好了，哪有什麼資格談替代流程，是吧！

所以，我會建議用例敘述也可以依此分爲兩個部分，核心的主要流程、僞畫面等放在一起，次要的替代流程和其他項目放一起，如表 6-1~2 所示。

用例	會員登入		
啓動者	會員	**支援者**	
主要流程 1. 會員輸入電郵和密碼，參考圖 6-9 的會員登入畫面。 2. 系統驗證會員身分。 3. 系統顯示歡迎訊息，其中包含會員姓名，以及等待付訂金交易筆數，參考圖 6-10 會員登入成功後的畫面。			
僞畫面			

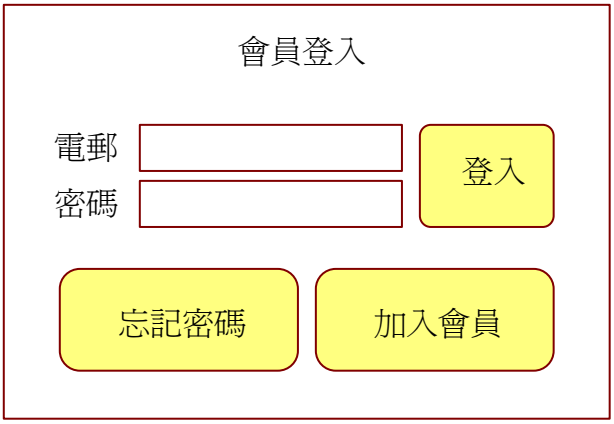

圖 6-9: 登入畫面(登入前)

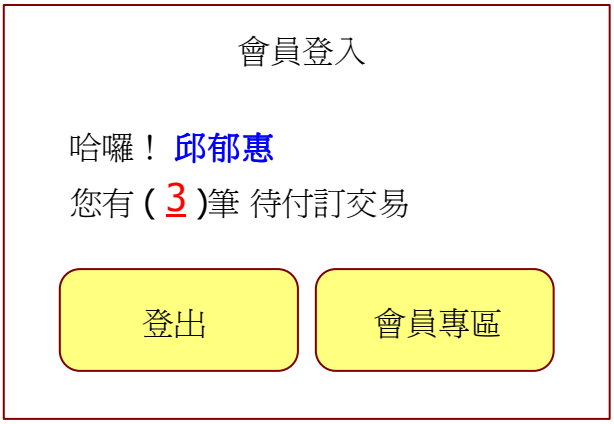

圖 6-10: 登入畫面(登入後)

表 6-1: 「會員登入」用例敘述-主要流程

用例	會員登入		
啓動者	會員	**支援者**	
替代流程 ☐ 資料不完整：客戶端提醒會員填入資料，直到資料完整才傳送到伺服端。 ☐ 驗證失敗：累積 5 次登入失敗，即鎖定，並出現請會員主動聯絡系統管理員的訊息。			
企業規則 BR1：以會員電郵做爲會員代號。 BR2：會員累積 5 次登入失敗，即鎖定該會員帳號。只要登入成功，則失敗次數歸零。 BR3：一個人只能申請一個會員身份。			
議題與其他 1. 由於，一個人只能申請一個會員身份，所以將類別圖中的個人與會員合併爲一。			

表 6-2: 「會員登入」用例敘述-替代流程及其他

至於，用例敘述要跟循序圖有多高程度的一致性，一直都是實務上令人難以拿捏之處。不過，如果兩者的細膩度一樣的話，一來那就不需要兩種工具了，不是嗎？再者，那麼用例敘述可能會變成是接力式的從分析師、設計師、一直到程序員，都會需要回頭去增添細節，而這真的是有價值的嗎？

因此，我會認爲，就細膩度而言：用例圖文最粗糙，再來是類別圖和

循序圖，程式碼則最細膩。同時，也是因爲這樣，才會讓後續接手的開發人員有發揮創意、加入心思和填寫細節的空間。

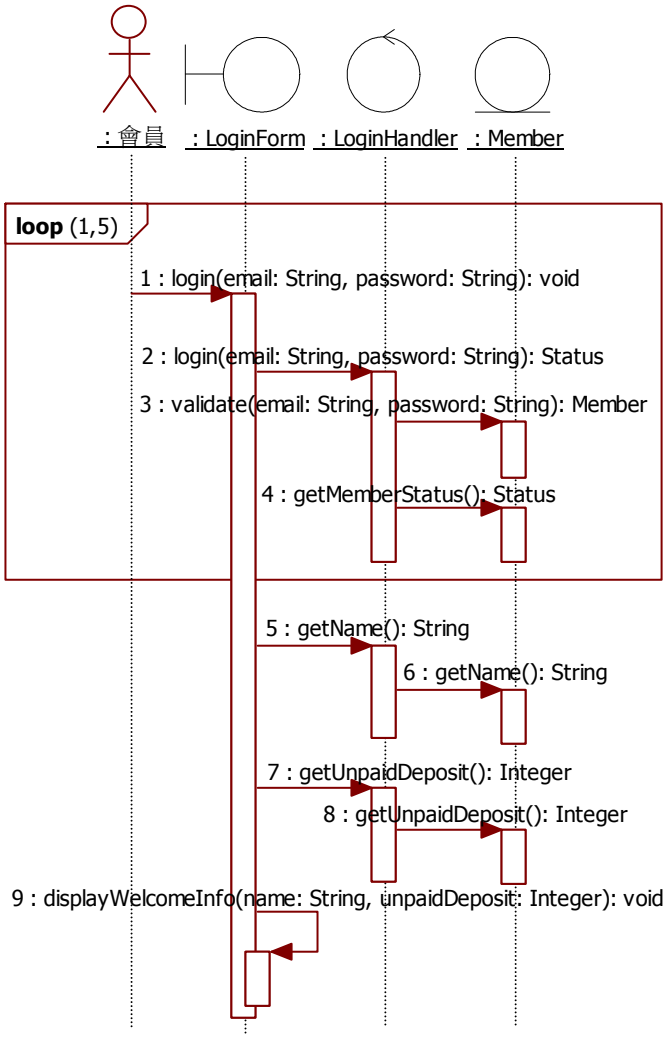

圖 6-11: 迴圈片段

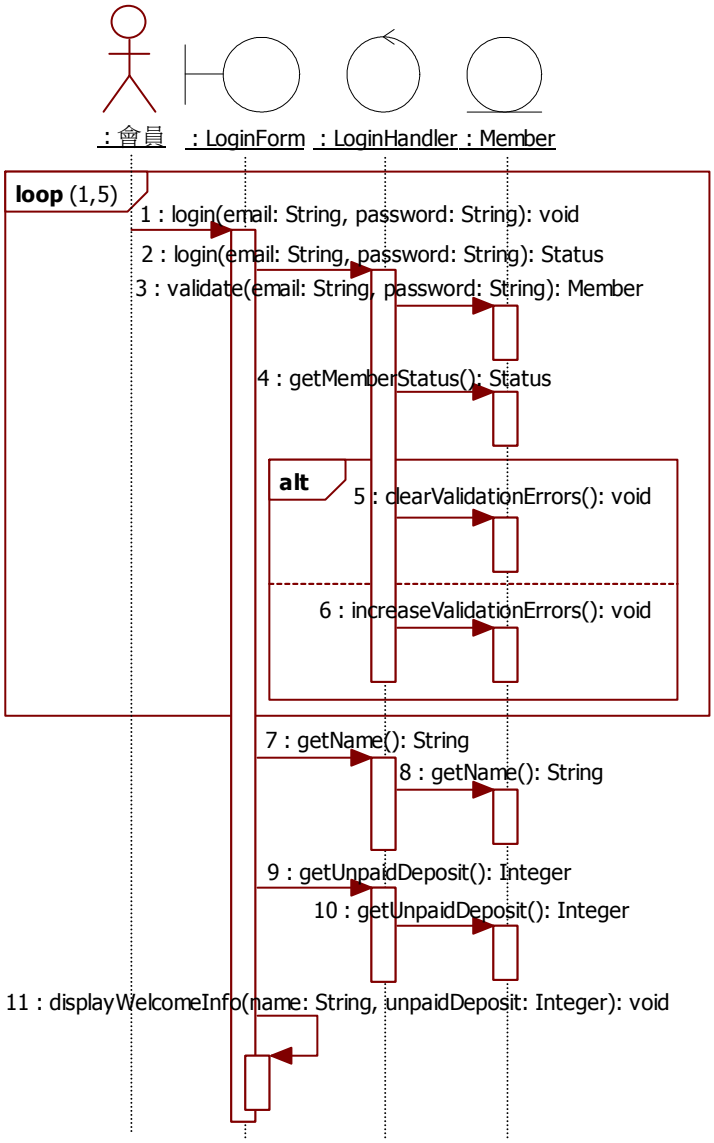

圖 6-12: 「會員登入」用例的循序圖(更細膩)

也因此，到底應該畫到圖 6-11 的細膩度，還是到圖 6-12 的細膩度，其實很難拿捏，也沒個標準。而且，似乎也很難評估這兩者哪個比較好。所以，比較保險的方法，可能還是盡快往下走，把這些 UML 設計圖和相關文件打包，交付給程序員，讓他們盡快寫出可以執行的陽春版用例出來，再來調整細膩度。

此處，假設我們的設計師最後繪製出圖 6-12 的循序圖，更新了會員實體類別，並且列出相關的 BCE 類別，如圖 6-13~14 所示。

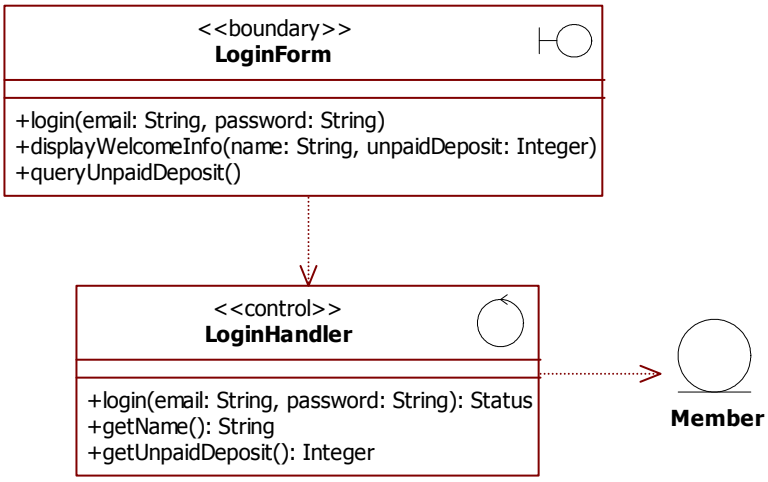

圖 6-13: BCE 類別

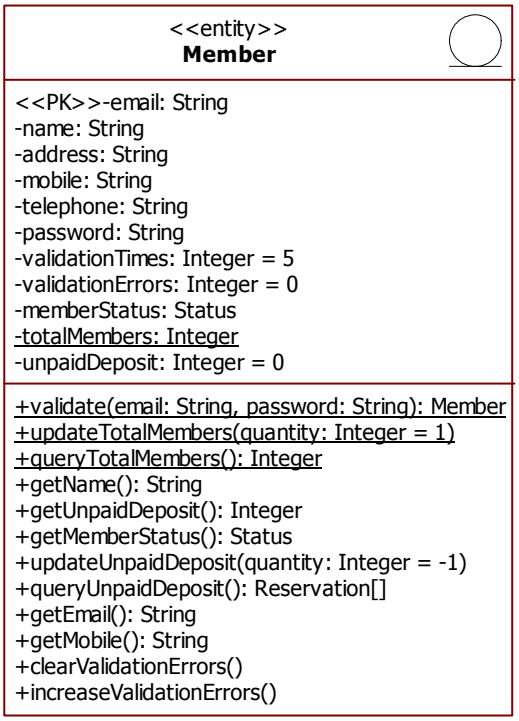

圖 6-14: 會員實體類別

接下來，設計師就可以先把下述文件打包，交付給程序員先行編碼了，如下：

1. 用例敘述-主要流程，如表 6-1 所示。
2. 針對主要流程所繪製的循序圖，如圖 6-12 所示。
3. 用例涉及到的類別圖，如圖 6-13~14 所示。

6.3.2 用例—通知已付訂

關於「通知已付訂」用例，設計師就可以打包下述文件，交付給程序員按圖施工，如下：

1. 「通知已付訂」用例敘述-主要流程，如表 6-3 所示。
2. 針對主要流程所繪製的循序圖，此處有經過修改，更加細膩了，如圖 6-18 所示。
3. 用例涉及到的類別圖，更新之後，如圖 6-19~22 所示。

用例	通知已付訂		
啓動者	會員	**支援者**	
相關用例	包含—發送電郵與簡訊通知		
主要流程 1. 會員查詢所有待付訂交易，參考圖 6-15 會員登入成功後的畫面。 2. 會員選擇一筆未付訂的訂房交易，參考圖 6-16 訂房清單畫面。 3. 會員挑選匯款銀行(代號-名稱)、填寫銀行帳號末 5 碼、挑選匯款日期、選擇匯款時間、填入匯款金額，參考圖 6-17 填寫付訂資料畫面。 4. 系統記錄付訂資料，包含匯款銀行(代號-名稱)、銀行帳號末 5 碼、匯款日期、匯款時間、匯款金額。 5. 系統更新訂房交易狀態。 6. 系統減少一筆未付訂交易筆數。			

7. 系統執行「發送電郵與簡訊通知」用例。
8. 系統回到第 2 步驟，更新訂房清單畫面。

僞畫面

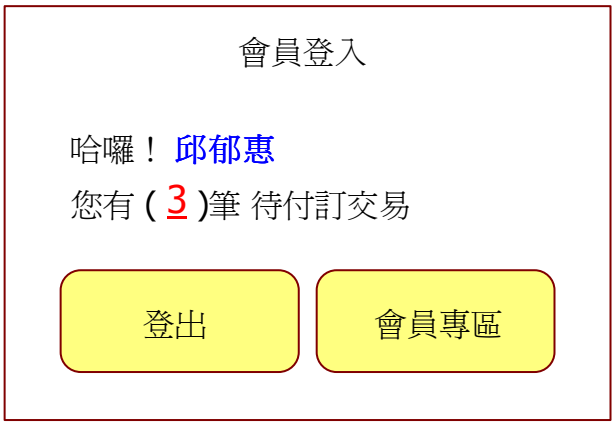

圖 6-15: 登入畫面(登入後)

訂房交易

民宿	訂房日期	預訂日期	付訂日期	總金額	訂金	狀態
浪漫滿屋	2009/12/10	2009/12/24	----/--/--	10,000	1,000	未付訂
星光部落	2009/12/11	2009/12/25	----/--/--	8,000	800	未付訂
開心民宿	2009/12/12	2009/12/26	----/--/--	6,000	600	未付訂

圖 6-16: 訂房清單畫面(所有未付訂交易列表)

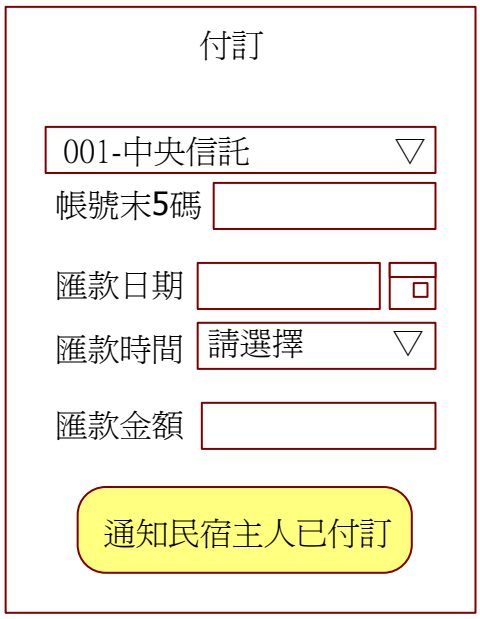

圖 6-17: 付款畫面

表 6-3: 「通知已付訂」用例敘述-主要流程

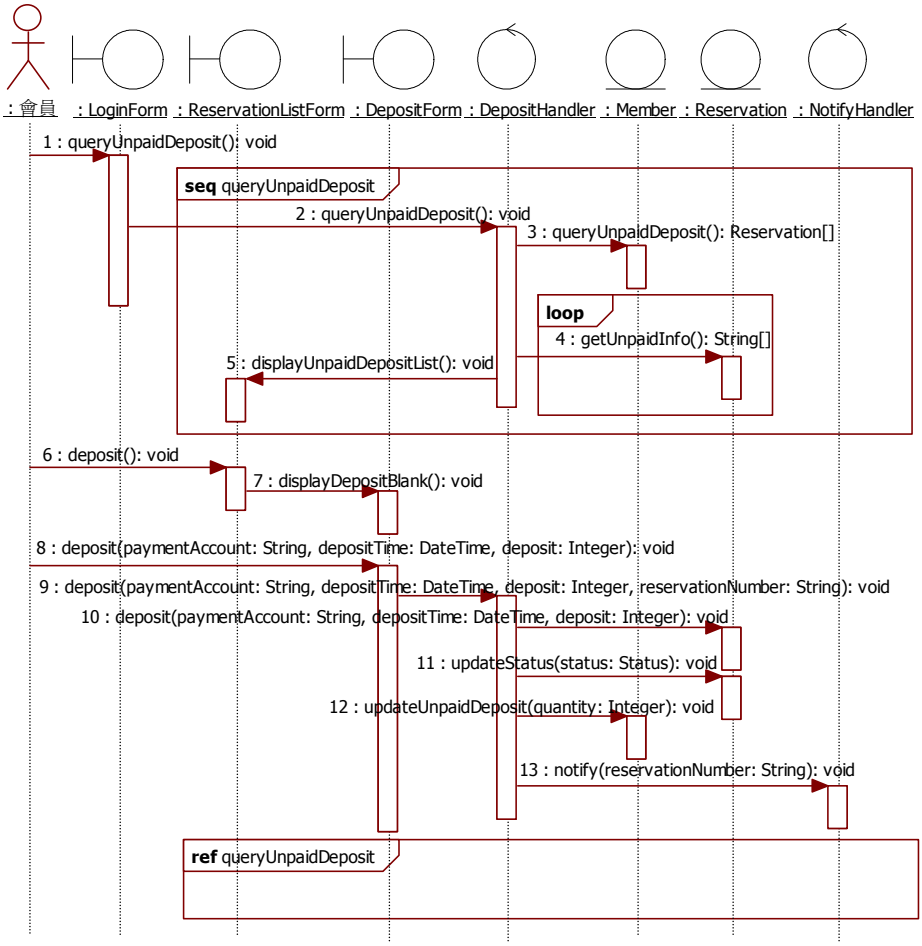

圖 6-18: 「通知已付訂」用例的循序圖(更細膩)

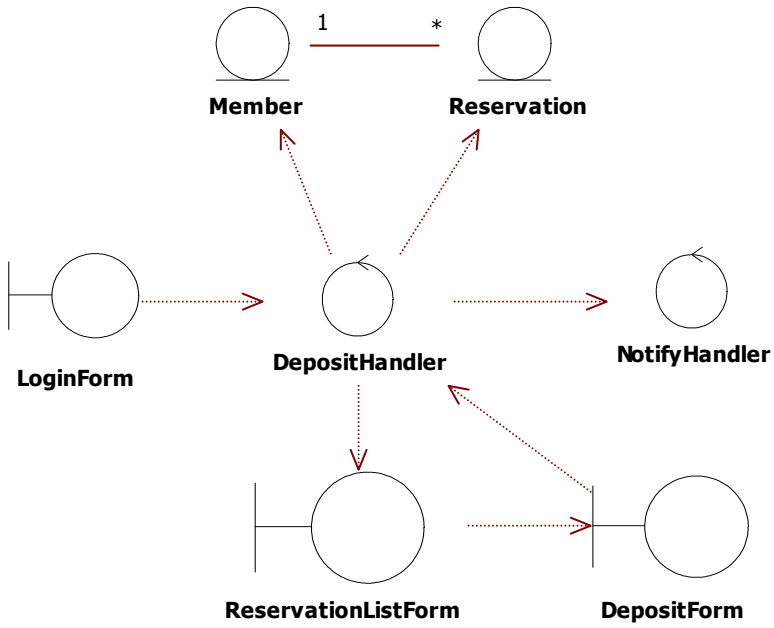
1
*
Member
Reservation
LoginForm
DepositHandler
NotifyHandler
ReservationListForm
DepositForm

圖 6-19: BCE 類別

圖 6-20: 邊界類別與控制類別

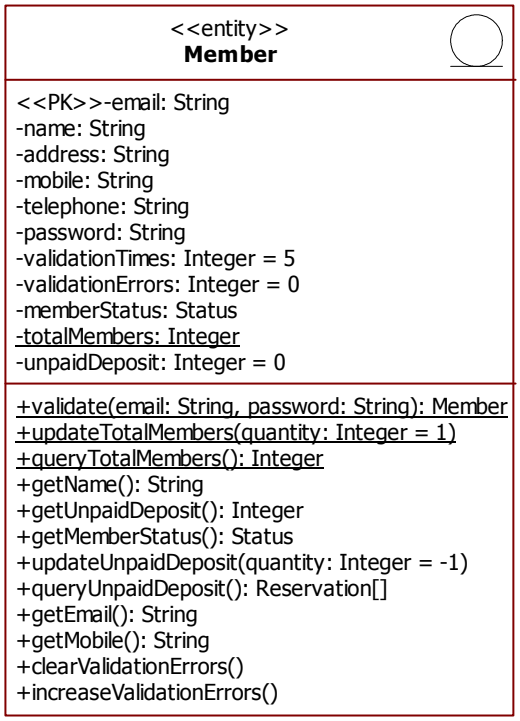
<<entity>>
Member
<<PK>>-email: String
-name: String
-address: String
-mobile: String
-telephone: String
-password: String
-validationTimes: Integer = 5
-validationErrors: Integer = 0
-memberStatus: Status
-totalMembers: Integer
-unpaidDeposit: Integer = 0
+validate(email: String, password: String): Member
+updateTotalMembers(quantity: Integer = 1)
+queryTotalMembers(): Integer
+getName(): String
+getUnpaidDeposit(): Integer
+getMemberStatus(): Status
+updateUnpaidDeposit(quantity: Integer = -1)
+queryUnpaidDeposit(): Reservation[]
+getEmail(): String
+getMobile(): String
+clearValidationErrors()
+increaseValidationErrors()

圖 6-21: 會員實體類別

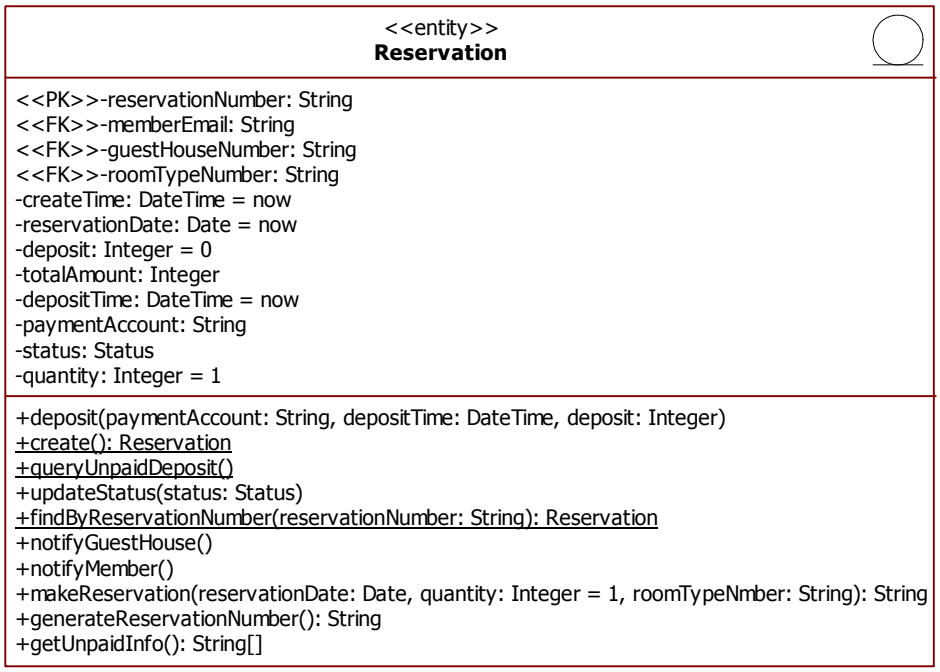

圖 6-22: 訂房實體類別

6.3.3 用例—發送電郵與簡訊通知

關於「發送電郵與簡訊通知」用例，設計師就可以打包下述文件，交付給程序員按圖施工，如下：

1. 「發送電郵與簡訊通知」用例敘述-主要流程，如表 6-4 所示。
2. 針對主要流程所繪製的循序圖，增加了並行片段，如圖 6-23 所示。
3. 用例涉及到的類別圖，如圖 6-24~27 所示。

用例	發送電郵與簡訊通知		**畫面**	無
啓動者	會員	**支援者**		
主要流程 1. 系統依據訂房序號，找出該訂房交易的會員與民宿主人。 2. 系統產生電郵，內容包含：訂房序號、訂房狀態、訂房日期、預定日期、付訂日期、總金額。 3. 系統發送電郵給民宿主人。 4. 系統發送電郵給會員。 5. 系統產生簡訊，內容包含：訂房序號、訂房狀態、訂房日期、預定日期、付訂日期、總金額。 6. 系統發送簡訊給民宿主人。 7. 系統發送簡訊給會員。				

表 6-4:「發送電郵與簡訊通知」用例敘述-主要流程

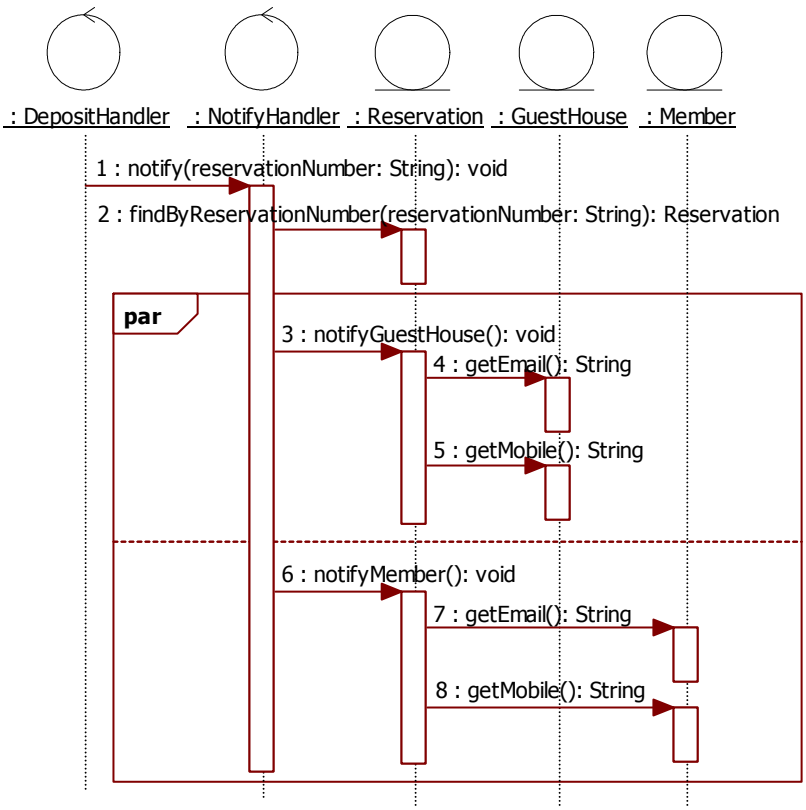

圖 6-23: 「發送電郵與簡訊通知」用例的循序圖

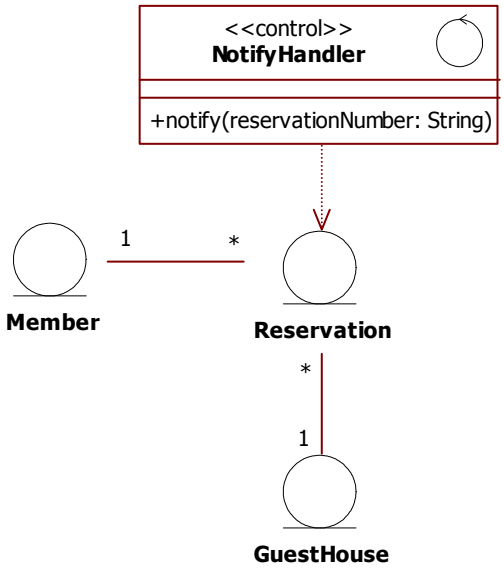
<<control>>
NotifyHandler
+notify(reservationNumber: String)
1
*
Member
Reservation
*
1
GuestHouse

圖 6-24: BCE 類別

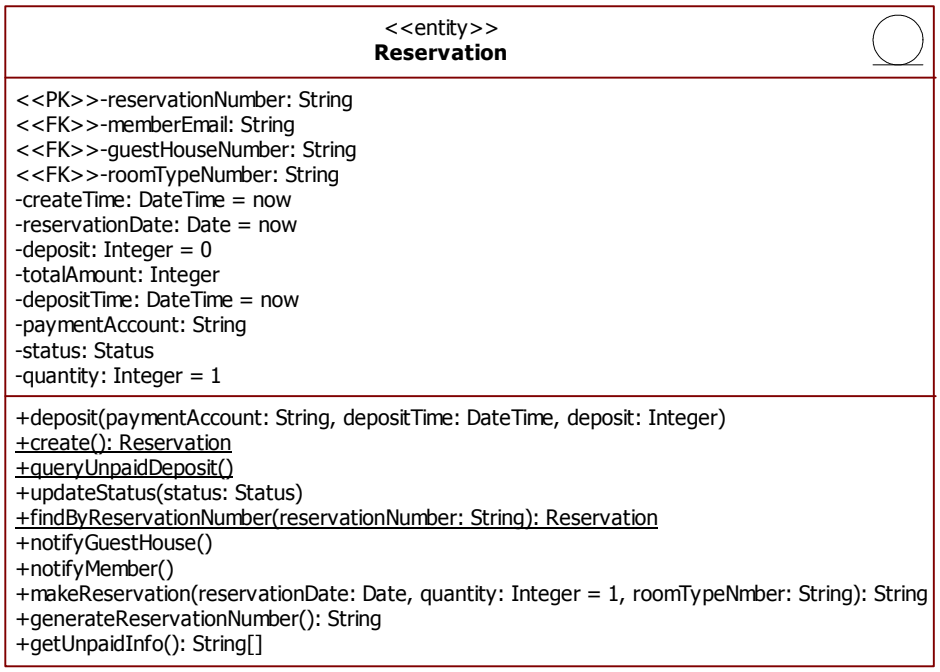
<<entity>>
Reservation
<<PK>>-reservationNumber: String
<<FK>>-memberEmail: String
<<FK>>-guestHouseNumber: String
<<FK>>-roomTypeNumber: String
-createTime: DateTime = now
-reservationDate: Date = now
-deposit: Integer = 0
-totalAmount: Integer
-depositTime: DateTime = now
-paymentAccount: String
-status: Status
-quantity: Integer = 1
+deposit(paymentAccount: String, depositTime: DateTime, deposit: Integer)
+create(): Reservation
+queryUnpaidDeposit()
+updateStatus(status: Status)
+findByReservationNumber(reservationNumber: String): Reservation
+notifyGuestHouse()
+notifyMember()
+makeReservation(reservationDate: Date, quantity: Integer = 1, roomTypeNmber: String): String
+generateReservationNumber(): String
+getUnpaidInfo(): String[]

圖 6-25: 訂房實體類別

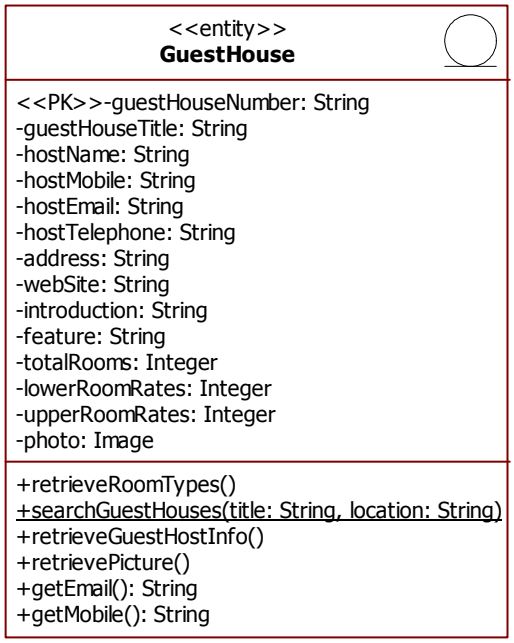
<<entity>>
GuestHouse
<<PK>>-guestHouseNumber: String
-guestHouseTitle: String
-hostName: String
-hostMobile: String
-hostEmail: String
-hostTelephone: String
-address: String
-webSite: String
-introduction: String
-feature: String
-totalRooms: Integer
-lowerRoomRates: Integer
-upperRoomRates: Integer
-photo: Image
+retrieveRoomTypes()
+searchGuestHouses(title: String, location: String)
+retrieveGuestHostInfo()
+retrievePicture()
+getEmail(): String
+getMobile(): String

圖 6-26: 民宿實體類別

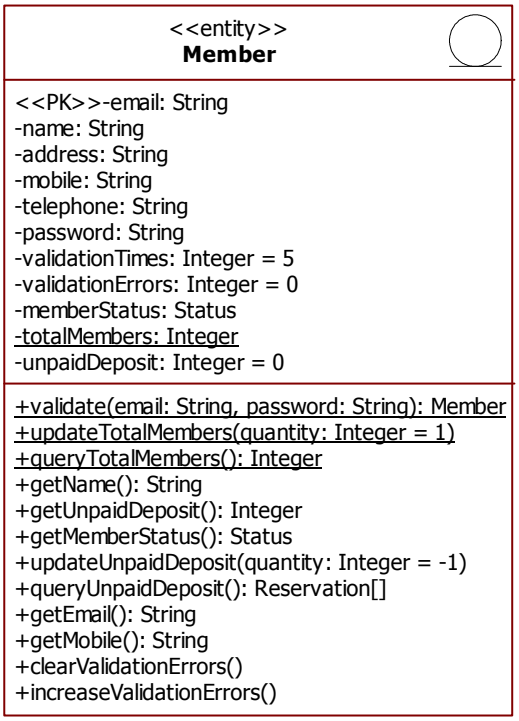

圖 6-27: 會員實體類別

6.3.4 用例—查詢民宿資料

設計師打包的 UML 設計圖與文件，如下：

1. 用例敘述-主要流程，如表 6-5 所示。
2. 「查詢民宿資料」用例的循序圖，如圖 6-30 所示。
3. 相關的 BCE 類別，如圖 6-31~34 所示。

用例	查詢民宿資料		
啓動者	訪客	**支援者**	
主要流程 1. 訪客輸入欲搜尋的民宿地點或名稱。 2. 系統找出符合搜尋條件的民宿清單，內容包含：小圖、民宿名稱、地址、房間數、房間價位，參考民宿清單畫面如圖 6-28 所示。 3. 訪客從中點選某一家民宿，查看民宿資料。 4. 系統顯示民宿資料，除了上述第 2 步驟的資料外，還額外包含民宿網址、簡介、特色、景觀照片，參考民宿畫面如圖 6-29 所示。			
僞畫面			


民宿清單
蘇澳
地點
搜尋
民宿小圖
浪漫滿屋
地址：宜蘭縣蘇澳鎮民眾路一段254號
房間：8間
價位：NT$6,000~12,000
民宿小圖
星光部落
地址：宜蘭縣蘇澳鎮星光路二段20號
房間：6間
價位：NT$3,000~10,000
民宿小圖
開心民宿
地址：宜蘭縣蘇澳鎮光明路104號
房間：8間
價位：NT$2,000~8,000

圖 6-28: 民宿清單畫面

圖 6-29: 民宿畫面

表 6-5: 「查詢民宿資料」用例敘述-主要流程

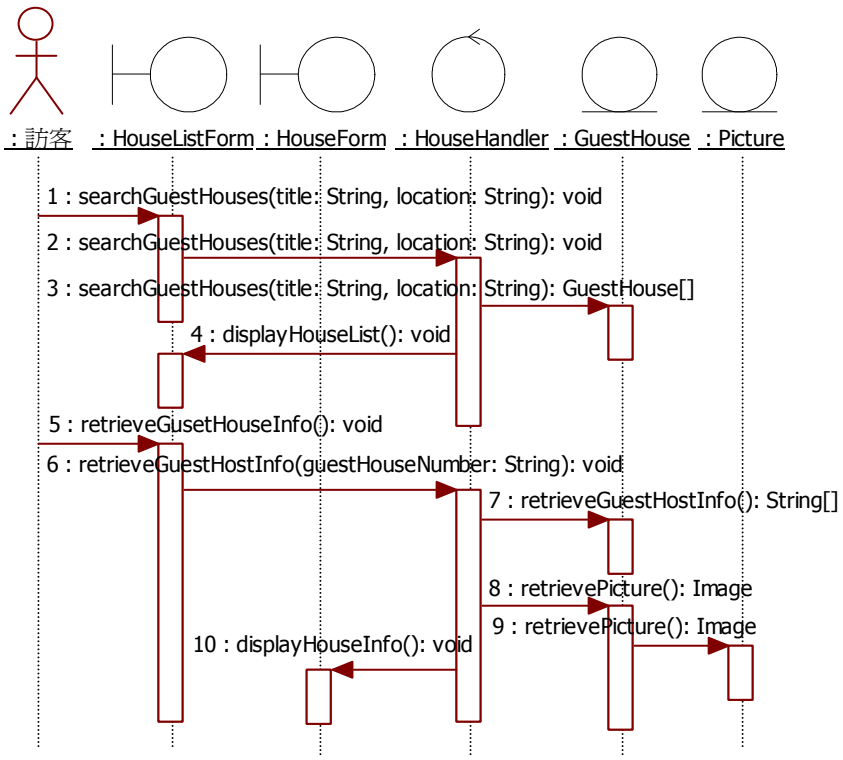

圖 6-30: 「查詢民宿資料」用例的循序圖

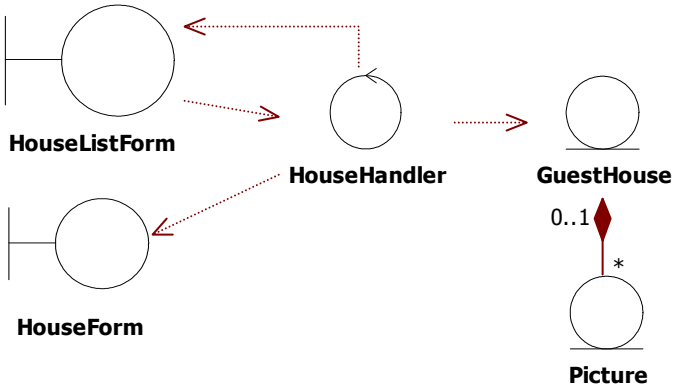

圖 6-31: BCE 類別

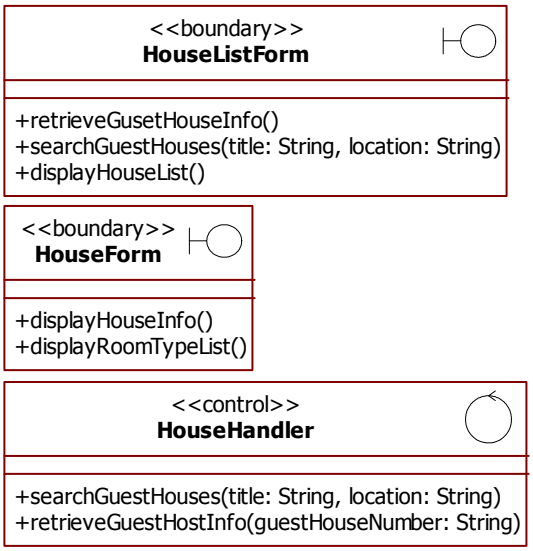

圖 6-32: 邊界類別與控制類別

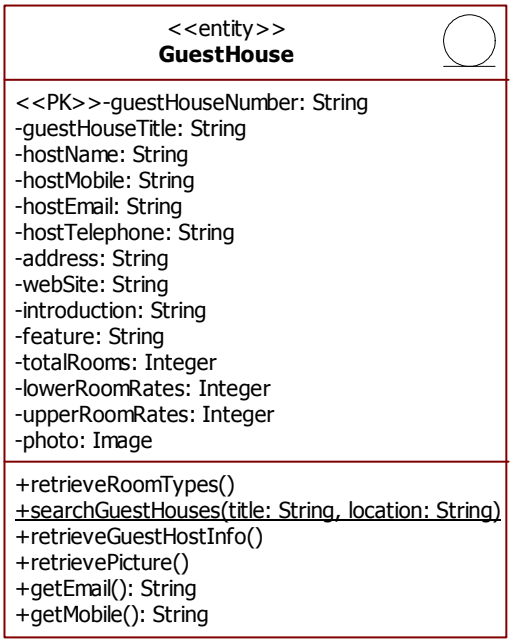

圖 6-33: 民宿實體類別

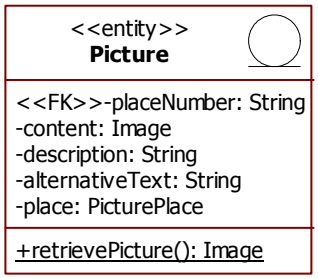

圖 6-34: 景觀圖片實體類別

6.3.5 用例—查詢房型資料

針對「查詢房型資料」用例，設計師需要打包下列 UML 設計圖與文件：

1. 「查詢房型資料」用例的主要流程，如表 6-6 所示。
2. 主要流程的循序圖，如圖 6-38 所示。
3. 此用例相關的 BCE 類別，如圖 6-39~42 所示。

用例	查詢房型資料		
啓動者	訪客	**支援者**	
主要流程 1. 訪客查看某一家民宿的所有房型，參考民宿畫面圖 6-35。 2. 系統顯示房型清單，包含房型名稱、床型、房間數、房價，參考房型清單畫面圖 6-36。 3. 訪客從中點選某一個房型，查看房型的細部資料。 4. 系統顯示房型資料，除了上述第 2 步驟的資料外，還額外包含房間設備、簡介、特色、景觀照片，參考圖 6-37 的房型畫面。			
僞畫面			


民宿介紹
浪漫滿屋
地址：宜蘭縣蘇澳鎮民眾路一段254號
房間：8間
價位：NT$6,000~12,000
網址：http://www.umltw.com
景觀圖片
簡介...
景觀圖片
特色...
景觀圖片
看房去

圖 6-35: 民宿畫面


房型清單
房型小圖
浪漫情人房
床型：超大雙人床
房間：4間
價位：NT$12,000
房型小圖
單身貴族房
床型：超大單人床2張
房間：2間
價位：NT$6,000
房型小圖
經典雙人房
床型：標準雙人床
房間：2間
價位：NT$6,000

圖 6-36: 房型清單畫面

圖 6-37: 房型畫面

表 6-6: 「查詢房型資料」用例敘述-主要流程

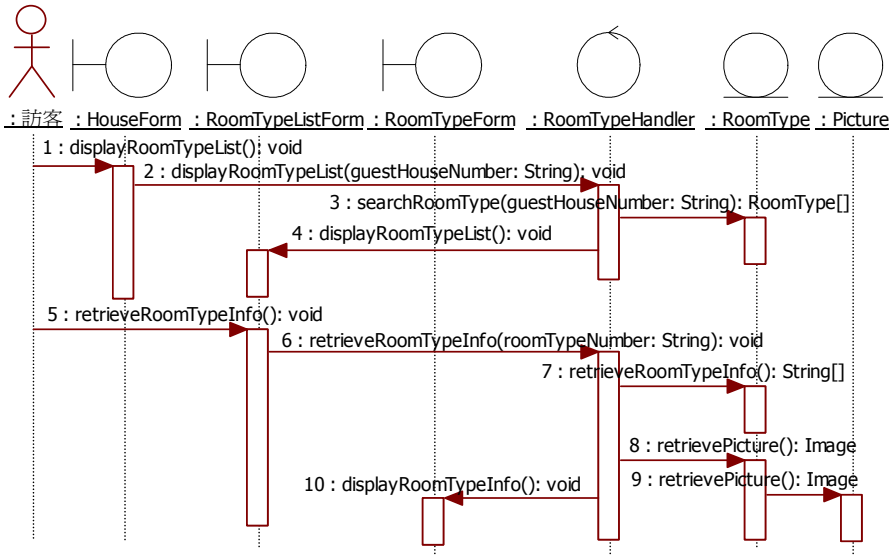

圖 6-38: 「查詢房型資料」用例的循序圖

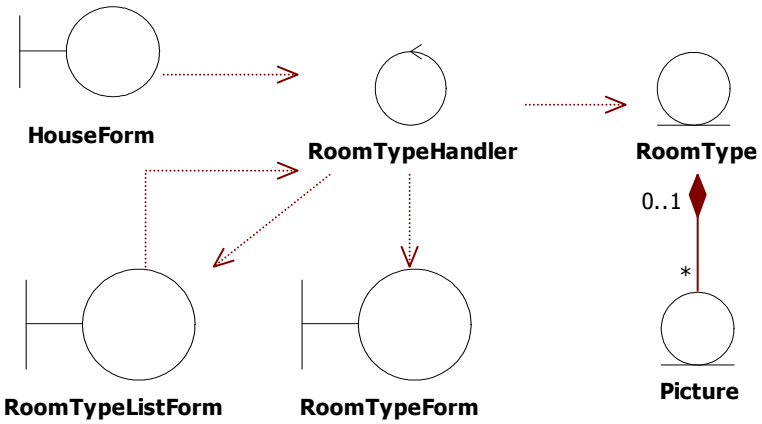

圖 6-39: BCE 類別

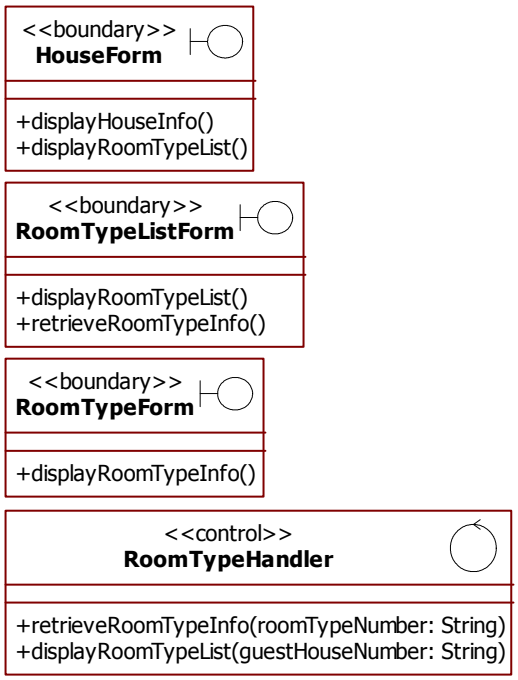

圖 6-40: 邊界類別與控制類別

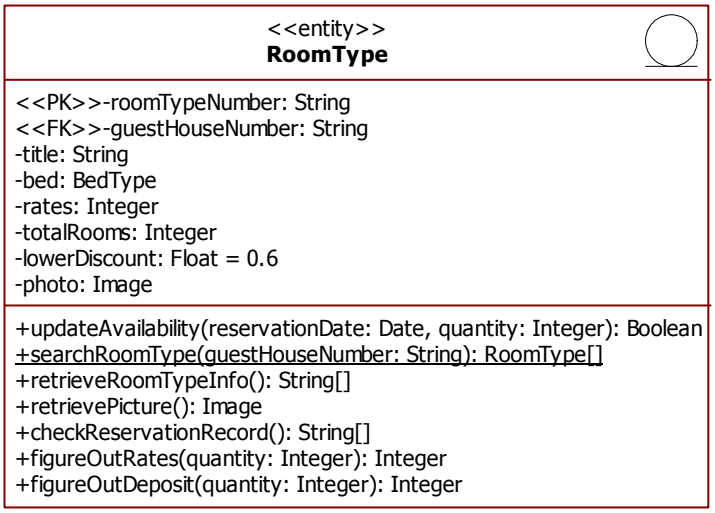

圖 6-41: 房型實體類別

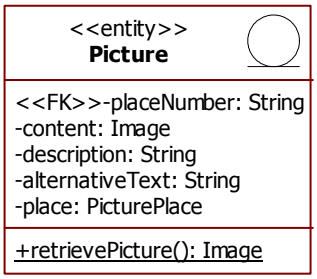

圖 6-42: 景觀圖片實體類別

6.3.6 用例—訂房

又到了最後一個最複雜的「訂房」用例了，設計師需要打包下列 UML

圖與文件：

1. 「訂房」用例敘述的主要流程，如表 6-7 所示。
2. 因爲「訂房」循序圖太複雜了，我們先分出兩張互動片段，以便降低複雜度，所以最後得到三張循序圖，如圖 6-46~48 所示。
3. 相關的 BCE 類別，如圖 6-49~54 所示。

用例	訂房		
啓動者	會員	**支援者**	
相關用例	包含—發送電郵與簡訊通知		
主要流程 1. 會員挑選預定的房型，查看其訂房紀錄，參考房型畫面如圖 6-43 所示。 2. 系統顯示出房型當月的預訂紀錄，參考預訂紀錄畫面如圖 6-44 所示。 3. 會員選定某一天預訂日期、房間數量。 4. 系統顯示出訂房資訊，包含有：房型名稱、預訂日期、預定房數、計算出訂房總金額、訂金。 5. 會員確認訂房資訊後，決定要訂房。 6. 系統新增一筆訂房交易。 7. 系統減少可預訂的空房數。 8. 系統增加一筆未付訂交易筆數。			

9. 系統產生交易序號。
10. 系統執行「發送電郵與簡訊通知」用例。
11. 系統顯示出訂房成功資訊，包含有：房型名稱、交易序號、訂金，以及請會員於 48 小時內付訂金的訊息。

僞畫面

圖 6-43: 房型畫面

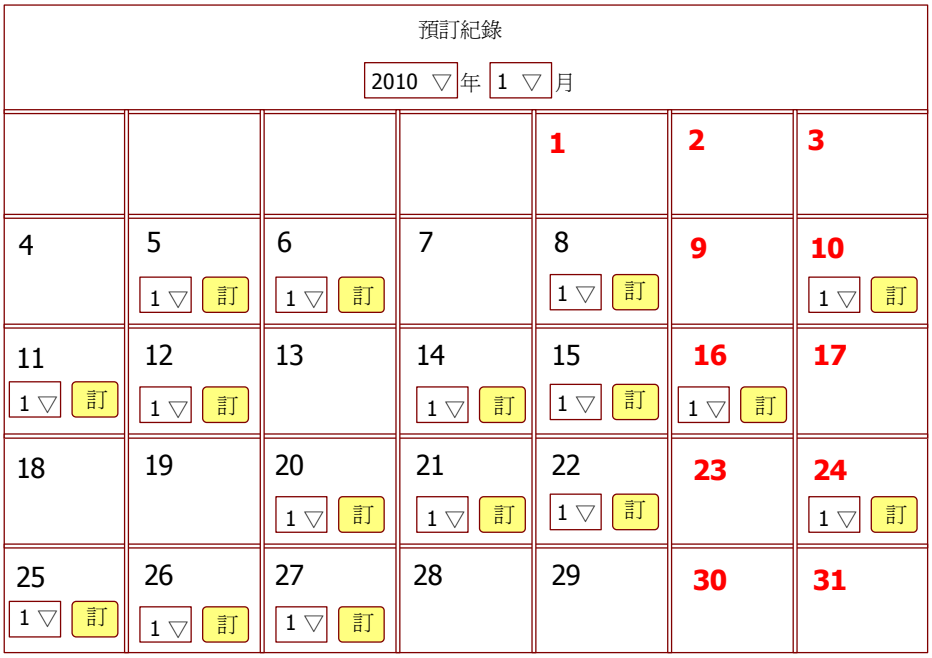
預訂紀錄
2010 ▽年 1 ▽月
1 2 3
4 5 6 7 8 9 10
11 12 13 14 15 16 17
18 19 20 21 22 23 24
25 26 27 28 29 30 31
1 ▽ 訂

圖 6-44: 預訂紀錄畫面

圖 6-45: 訂房資訊畫面

表 6-7: 「訂房」用例敘述-主要流程

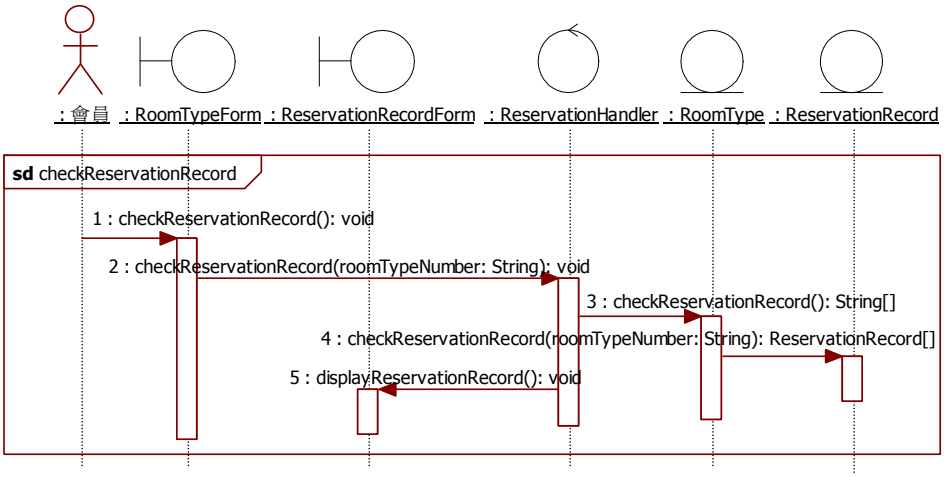

圖 6-46: 訂房-查看預訂紀錄

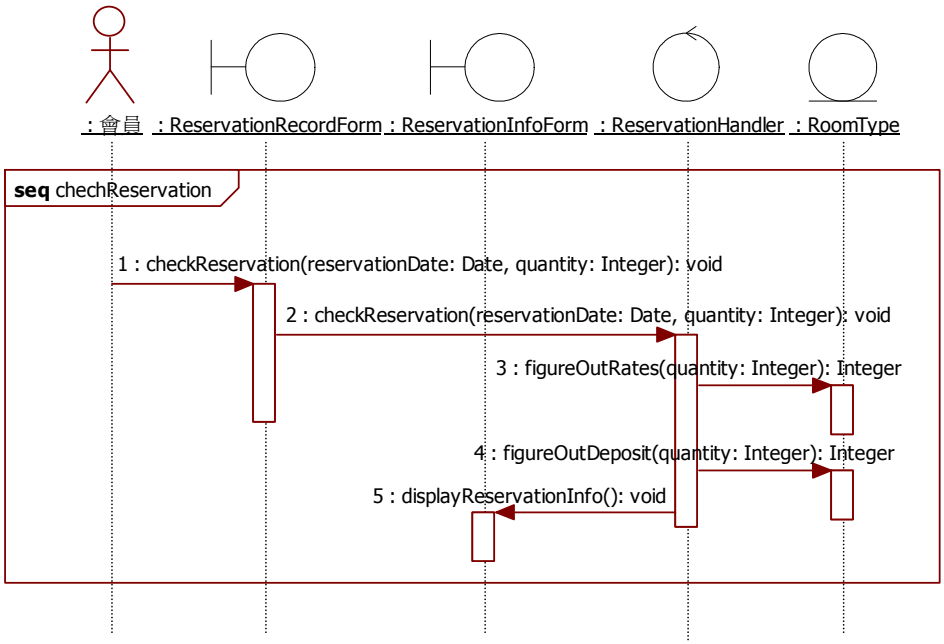
：會員
：ReservationRecordForm
：ReservationInfoForm
：ReservationHandler
：RoomType
seq chechReservation
1 : checkReservation(reservationDate: Date, quantity: Integer): void
2 : checkReservation(reservationDate: Date, quantity: Integer): void
3 : figureOutRates(quantity: Integer): Integer
4 : figureOutDeposit(quantity: Integer): Integer
5 : displayReservationInfo(): void

圖 6-47: 訂房-查看預訂

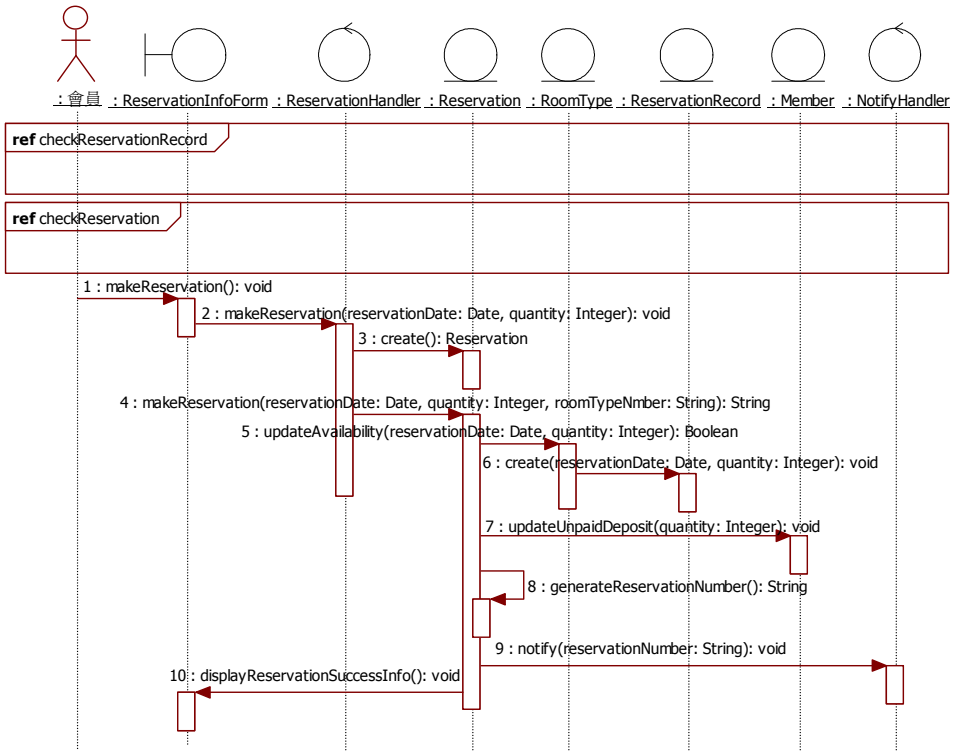
：會員
：ReservationInfoForm
：ReservationHandler
：Reservation
：RoomType
：ReservationRecord
：Member
：NotifyHandler
ref checkReservationRecord
ref checkReservation
1 : makeReservation(): void
2 : makeReservation(reservationDate: Date, quantity: Integer): void
3 : create(): Reservation
4 : makeReservation(reservationDate: Date, quantity: Integer, roomTypeNmber: String): String
5 : updateAvailability(reservationDate: Date, quantity: Integer): Boolean
6 : create(reservationDate: Date, quantity: Integer): void
7 : updateUnpaidDeposit(quantity: Integer): void
8 : generateReservationNumber(): String
9 : notify(reservationNumber: String): void
10 : displayReservationSuccessInfo(): void

圖 6-48: 訂房

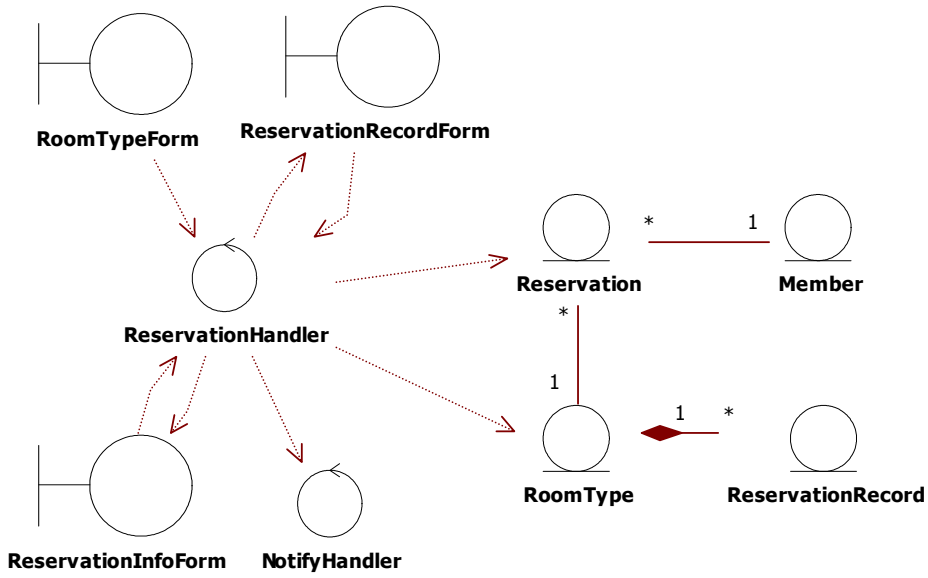
RoomTypeForm
ReservationRecordForm
ReservationHandler
Reservation
Member
RoomType
ReservationRecord
ReservationInfoForm
NotifyHandler
*
1
*
1
1
*

圖 6-49: BCE 類別

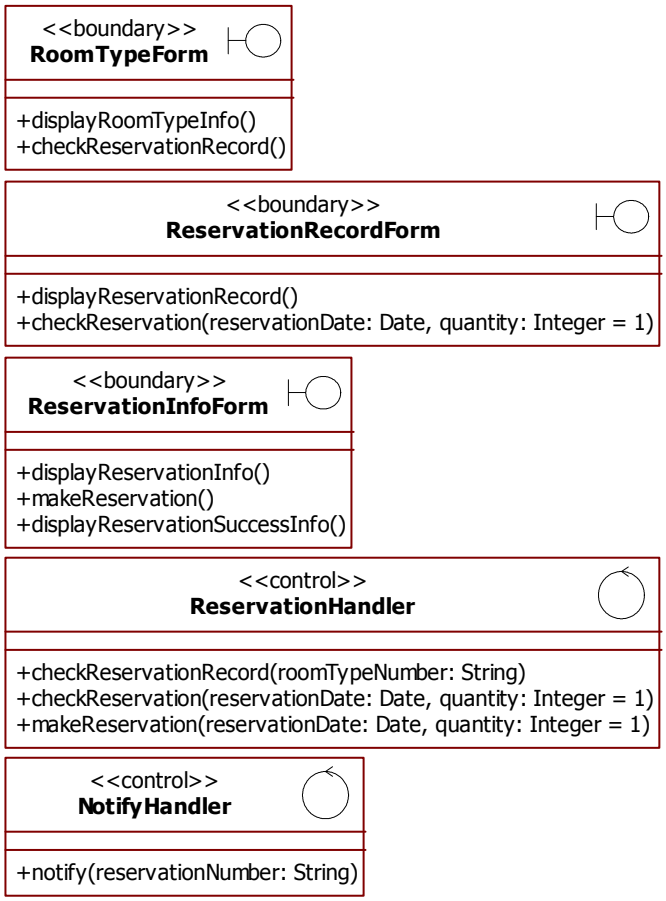

圖 6-50: 邊界類別與控制類別

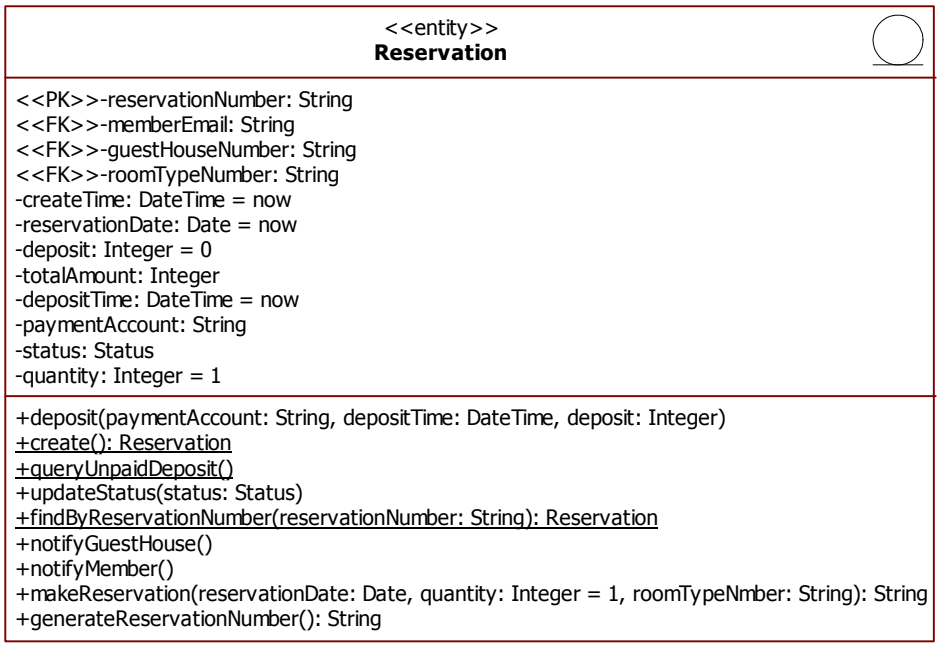
<<entity>>
Reservation
<<PK>>-reservationNumber: String
<<FK>>-memberEmail: String
<<FK>>-guestHouseNumber: String
<<FK>>-roomTypeNumber: String
-createTime: DateTime = now
-reservationDate: Date = now
-deposit: Integer = 0
-totalAmount: Integer
-depositTime: DateTime = now
-paymentAccount: String
-status: Status
-quantity: Integer = 1
+deposit(paymentAccount: String, depositTime: DateTime, deposit: Integer)
+create(): Reservation
+queryUnpaidDeposit()
+updateStatus(status: Status)
+findByReservationNumber(reservationNumber: String): Reservation
+notifyGuestHouse()
+notifyMember()
+makeReservation(reservationDate: Date, quantity: Integer = 1, roomTypeNmber: String): String
+generateReservationNumber(): String

圖 6-51: 訂房實體類別

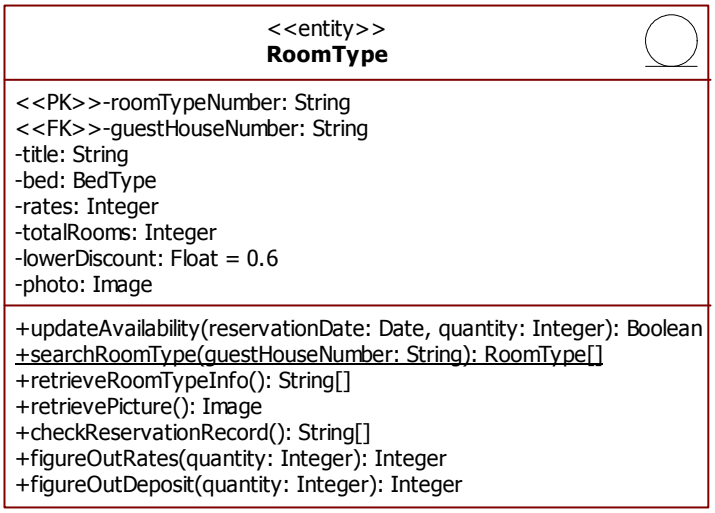

圖 6-52: 房型實體類別

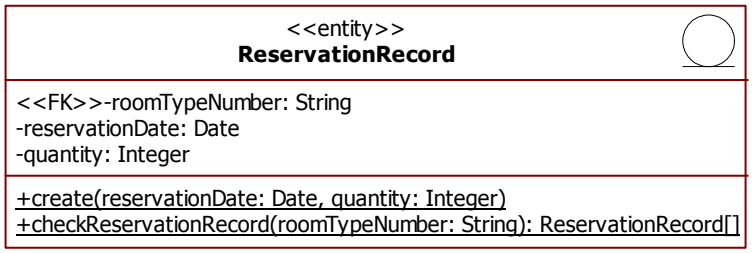

圖 6-53: 預訂紀錄實體類別

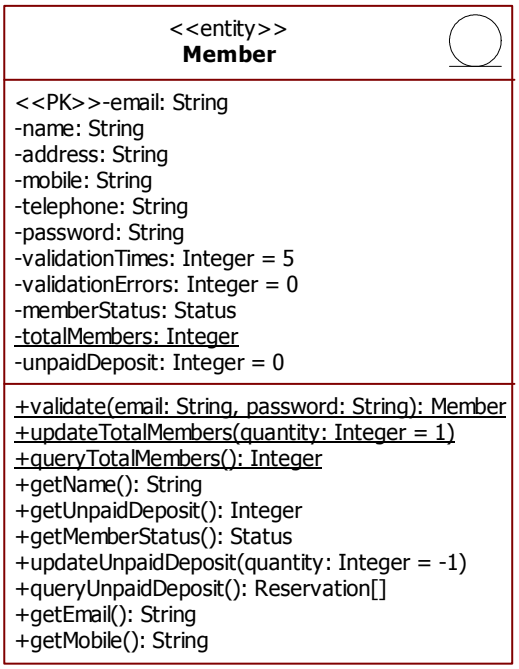

圖 6-54: 會員實體類別

最後，我們還是一併列出之前撰寫的「訂房」用例的替代流程及其他，如表 6-8 所示。

用例	訂房		
啓動者	會員	**支援者**	
替代流程 SR1：畫面上必須標記必須填寫的欄位，並且在客戶端先檢驗欄位，並且提醒使用者填寫完整資料，直到必填欄位完整之後，才會回送到			

伺服端。
企業規則 BR4：訂房交易序號的編碼規則爲「預訂 yyyyMMdd0001」，每日以流水號 0001 起始，每日流水號最大到 9999。 BR5：訂金=總價×0.1。 BR6：會員需在交易成立後，48 小時內付訂金。
議題與其他 1. 編定「系統規則」(System Rule,SR)，做爲整個訂房系統皆須遵守的規則。 2. 企業規則和系統規則將集中管理，用例敘述中僅片面記錄規則初次出現的時刻。

表 6-8: 「訂房」用例敘述-替代流程及其他

6.3.7 其他

此外，還有一些共用的列舉型別或公用類別，我們就一併列在圖 6-55 中了。

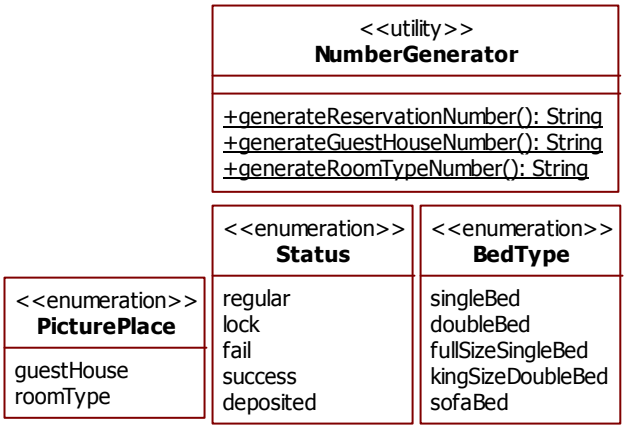

圖 6-55: 會員實體類別

6.4 UML 囈語

跟上一章最後一小節的「後話」一樣，這些年來累積了一些對 UML 的感受，不容易以系統性的方式安排在書的正文中。原先我把這些感受條列在我的 UML Blog(http://www.umltw.com)中，現在一併在此處列出，與各位讀者分享，也以此做爲這本實用書的抽象結尾，引發我們探究更多關於 UML/OOAD 一切的熱情。

1. UML 是輔助，不是限制。
2. 腦海中的想像，永遠比 UML 的表達，來得重要。
3. UML 的表達是拋磚引玉，別眷戀眼前的圖，因爲更佳的設計將破圖而生。
4. 製圖期間的三不政策：不管 UML 語法是否正確、不管自己的觀

點是否合理、不管專家說些什麼，一切等畫完圖再說。

5. 充分且完整地表達所思，然後反思，即使不合乎 UML 語法也絕不罷休。
6. 創意通常在超越 UML 語法而現身，但是爲了讓眾人分享，事後還是得勉爲其難地用 UML 來表達。
7. 學 UML 像練雙節棍，耍的好，可以唬人；耍不好，會 K 得自己滿頭包。
8. UML 圖示少用一點，你就可以看到更多。
9. 六祖慧能以手指月；UML 是指，不是月啊！
10. 如果 UML 不去重視本身過於繁雜的難題，UML 很快就會成爲難題的一部分，或者現在就已經是如此了。
11. 對於我們無緣參與的專案而言，唯一的替代品就是 UML 圖與原始程式碼。
12. 發現自己陷在 UML 圖坑裡時，千萬不要再往下掘洞。
13. 陷溺於過多的 UML 圖示中，恐怕會一事無成，並且對結果感到失望。
14. 多少次我們興高采烈地買了 UML 書回去，翻開之後才發現，艱深抽象的文字毫無樂趣又令人挫折。
15. UML 的妥善組織力，能使複雜的系統顯得比較簡單。
16. 爲了解釋複雜、理解複雜、處理複雜，學習 UML，就有絕對的必要。
17. 採用 UML 必然有風險和成本，但遠低於不採用所帶來的長程風險和成本。

18. 以可預測性來替代變動性，可以讓 UML 從小卒變英雄。
19. 趕快畫出下一張圖，用來麻醉腦袋。畫出下一張圖時，其實並沒有真的改變任何事。不過，當注意力轉向可視見的圖形時，已經增加了正面的能量和動力。
20. 製圖只有兩個問題：一、知道自己在想什麼，但不知道如何表達；二、根本不知道自己在想什麼。
21. 開始動手製圖，體內的某一部分就會不由自主地想著：「我沒有全部的資料。」此時，我們就會感覺好像遺漏了什麼，但又不確定是什麼，所以就名正言順的停擺了。
22. 把畫不下去的圖找出來，什麼也不修，全部砍了，這種感覺很不錯。面對令你腦殘的方法之一是，勇敢刪除。
23. 在我畫了什麼之前，怎麼會知道自己在想什麼？
24. 如果沒有工具把圖畫下來，甚至可以感覺到腦袋根本不想深思。
25. 有思考 UML 更清明的時刻，也有根本不該思考 UML 的時候。
26. 頭腦並不擅長記憶、提醒等工作，UML 能夠擺脫這些「低階」任務，讓我們專注於思考。
27. UML 要用的好，必須學會見樹又見林，且了解其中之關聯。
28. 得出好圖的方法是，先畫出一堆圖。
29. 有太多人愛畫圖而不愛改圖，這樣就等於放棄了吸收更多新思想和可能性的機會。
30. 不要求去學太多的 UML 圖示，因爲學到時往往已經過時了。
31. 繪製 UML 圖時，先要知道自己在找尋什麼，然後再去找尋它。
32. 建構 UML 模式不光是設計的藝術，不光是說服別人認同就夠

了，而是創造出一種讓參與者想說出自己想法的情境。

33. 達到效率，卻抹殺了創意，不要以效率來衡量建構 UML 模式期間的所有活動。
34. 決定 UML 價值如何，是需求的假象，而非 UML 實際的價值。
35. 當我們能從 UML 圖中的錯誤之處學習，錯誤才有價值可言。
36. UML 圖可以在一開始創造出讓我們的觀眾(使用者、夥伴、上司、顧客)了解我們，與我們建立理性溝通及達成共識的方式。
37. UML 圖比一連串的事實或資料更能在觀眾心中停留更久。
38. 在 UML 專業技能上非常精通，既是優勢也是弱點。
39. 對許多 UML 圖示，你只需要知道一點點就足夠了。
40. 繪製 UML 圖不是在產出結論，而是一個收集資訊和創意的過程。
41. 阻礙我們繪圖的障礙很少是由於缺乏技術性資訊，更多的是由於缺乏自信與勇氣。
42. 透過 UML 圖打開自己的心扉，而不是封閉自己的頭腦。
43. 學會讓 UML 圖跟隨你的思想，而不是讓思想跟著你的 UML 圖。
44. 捍衛 UML 圖的表像是沒有意義的，應該捍衛的是你的構想。
45. UML 雖然是讓我們能做很多事情的工具，卻也可能使我們失去很多能力。
46. 簡化 UML 圖是減少明顯的，增加有意義的。
47. 面對複雜的 UML 圖示，只能簡單的選用。
48. UML 可以讓人跟人之間的溝通更明確、更聚焦，但是不可能取代人跟人之間的溝通。

49. 不能帶來價值的 UML 元素，不值得花時間和成本去用它。
50. 用得好，UML 就像能夠改善體質的維他命；用不好，UML 可能變成只會帶來熱量的垃圾食物。

A
成本估算

※※

A.1 成本估算

成本估算一直都是件難事！記得曾經有個朋友跟我說，如果專案成本可以估算的準，那就可以立於不敗之地了。因爲，這樣就可以選擇賺得多的專案來接，賠錢的專案不接。但是，立於不敗之地的公司鮮少，可見得成本估算有多難搞定。

那麼，該如何估算 UML/OOAD 專案的成本呢？

知名的 UML/OOAD 大師—Ivar Jacobson，他同時是用例技術與 UML 的主要創辦人之一。Ivar Jacobson 在 1992 年所出版的巨擘《Object-Oriented Software Engineering: A Use Case Driven Approach》一書中，提出了用例技術。

而隔年 1993 年，與 Ivar Jacobson 同公司的 Gustav Karner，提出的一篇論文《Resource Estimation for Objectory Projects》中，正式提到使用「用例點」(Use Case Point,UCP)來估算成本。Gustav Karner 在論文中提到，自己是參考 Allan Albrecht 在 1979 年所提出的「功能點」(Function Point,FP)概念，提出用例點的估算方法。

從這一段小小的歷史來看，用例點有下列兩個特點，或許值得我們採用：

1. 系出名門—與用例技術出自於同一家公司 Objective Systems AB，相當於有 Ivar Jacobson 的背書。
2. 歷史悠久—提出於 1993 年，於用例技術提出的隔年，至今已有十多年歷史了，可以假設它是個成熟的技術了。更何況 Gustav Karner 參考的功能點技術，來自於 1979 年，更爲成熟。

A.2 用例點

用例點的計算公式很簡單，如表 A-1 所示。實務上，困難之處在於，用例的顆粒度、用例敘述的細膩度、技術和環境的評價，都會導致估算不準確。所以，在我們懂得用例點技術之後，還需要一段時間的實務磨鍊，才可能立於不敗之地。

用例點＝未調整用例點x技術複雜因子x環境因子
未調整用例點＝參與者總權重＋用例總權重
技術複雜因子＝0.6＋(0.01x技術總權重)
環境因子＝1.4＋(-0.03x環境總權重)
一個用例點大約需要耗費 20~28 人時(Man Hours)

表 A-1: 用例點公式

A.3 參考資料

如果，您對用例點有興趣的話，網路上可以查到許多資料。或者，下列文獻您也可以找來念念：

1. 我在 2009 年出版的《寫給 SA 的 UML/UseCase 實務手冊》一書中，花了一整章說明並示範，如何使用「用例點」估算工時。
2. 《Resource Estimation for Objectory Projects》，就是這篇論文提出用例點技術，由 Gustav Karner 於 1993 年提出的，網路上可以找到這篇論文，免費的。不過，既然是論文，所以不容易看懂，要有心理準備。

3. 《Applying Use Cases: A Practical Guide》2nd ed.，這本由 Ivar Jacobson 幫忙寫序的用例書籍，也花了一些篇幅在談用例點，寫得很簡明。好書一本，雖然有點舊，這本書是 2001 年出版的。
4. 《More About Software Requirements: Thorny Issues and Practical Advice》，這本書比較新，2006 年出版的。不過，在用例點概念上，沒有新的見解，跟《Applying Use Cases: A Practical Guide》一樣，簡明易懂。
5. 《Software Cost Estimation Using Use Case Points: Getting Use Case Transactions Straight》，這篇文章最新，2009 年中出自於 IBM Rational。這篇文章同樣系出名門，有一段小歷史。Rational 軟體公司 1995 年收購了 Ivar Jacobson 創辦的 Objective Systems AB 公司，然後在 2003 年 IBM 收購了 Rational。這篇文章免費，值得一看，主要提到了如何評估用例的業務量，有助於評估用例權重。
6. 《 Object-Oriented Software Engineering: A Use Case Driven Approach》，這本是不可不讀的用例技術聖經，由 Ivar Jacobson 於 1992 年出版。既然，用例點以用例技術爲主，至少得讀讀這本談用例技術的經典書籍吧！