


軟體工程實務
Class 3: System Modeling

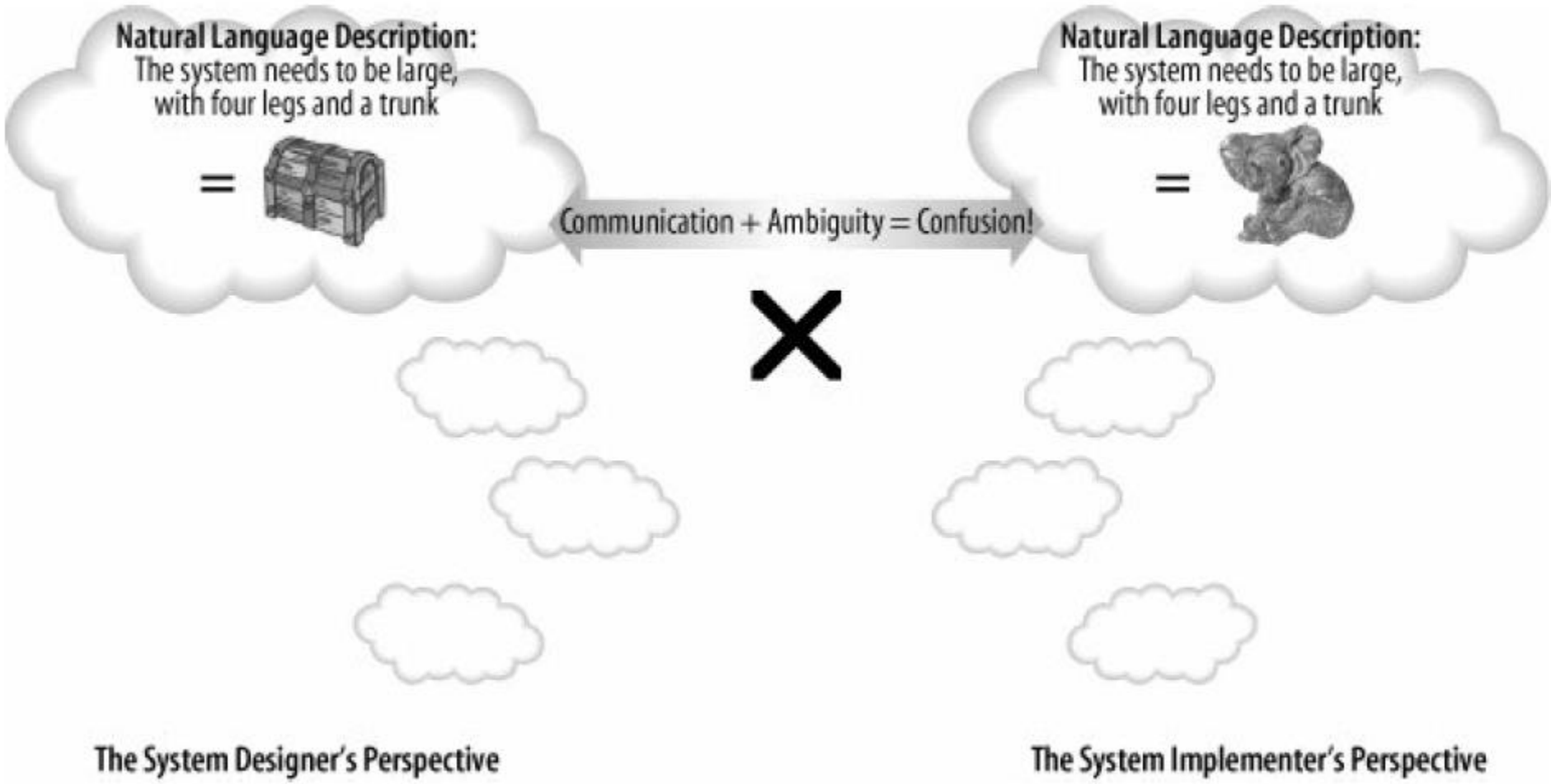
逢甲大學 人工智慧研究中心
資 訊 工 程 學 系

許懷中

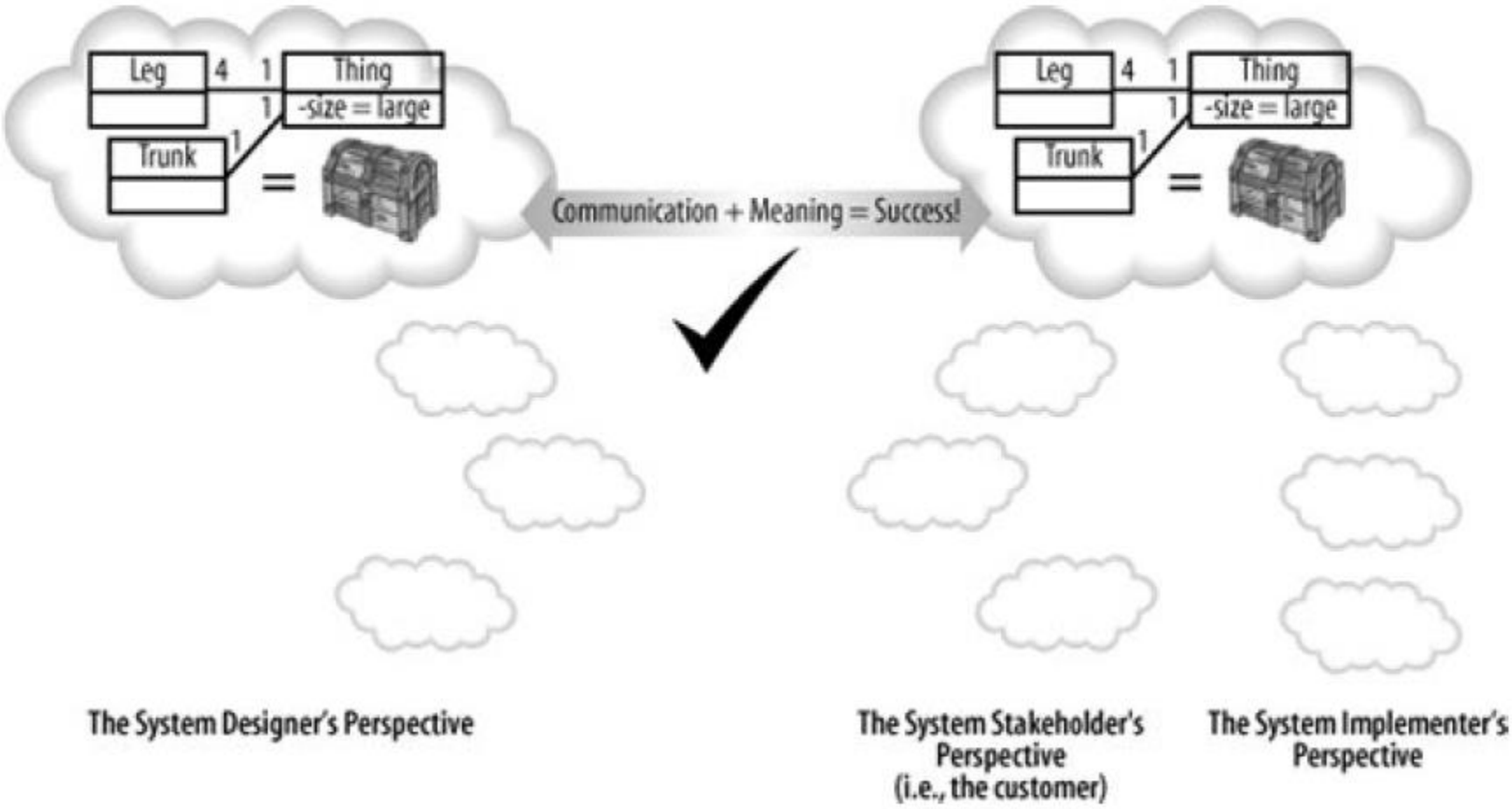
為什麼需要系統塑模 (System Modeling)

- 建構系統的抽象模型
  - 作為概念與實作(程式碼)中間的橋樑
  - 呈現系統的不同面向
  - 幫助瞭解系統的功能
  - 以圖形化的符號 (Notation) 表達

為什麼需要系統塑模 (cont.)



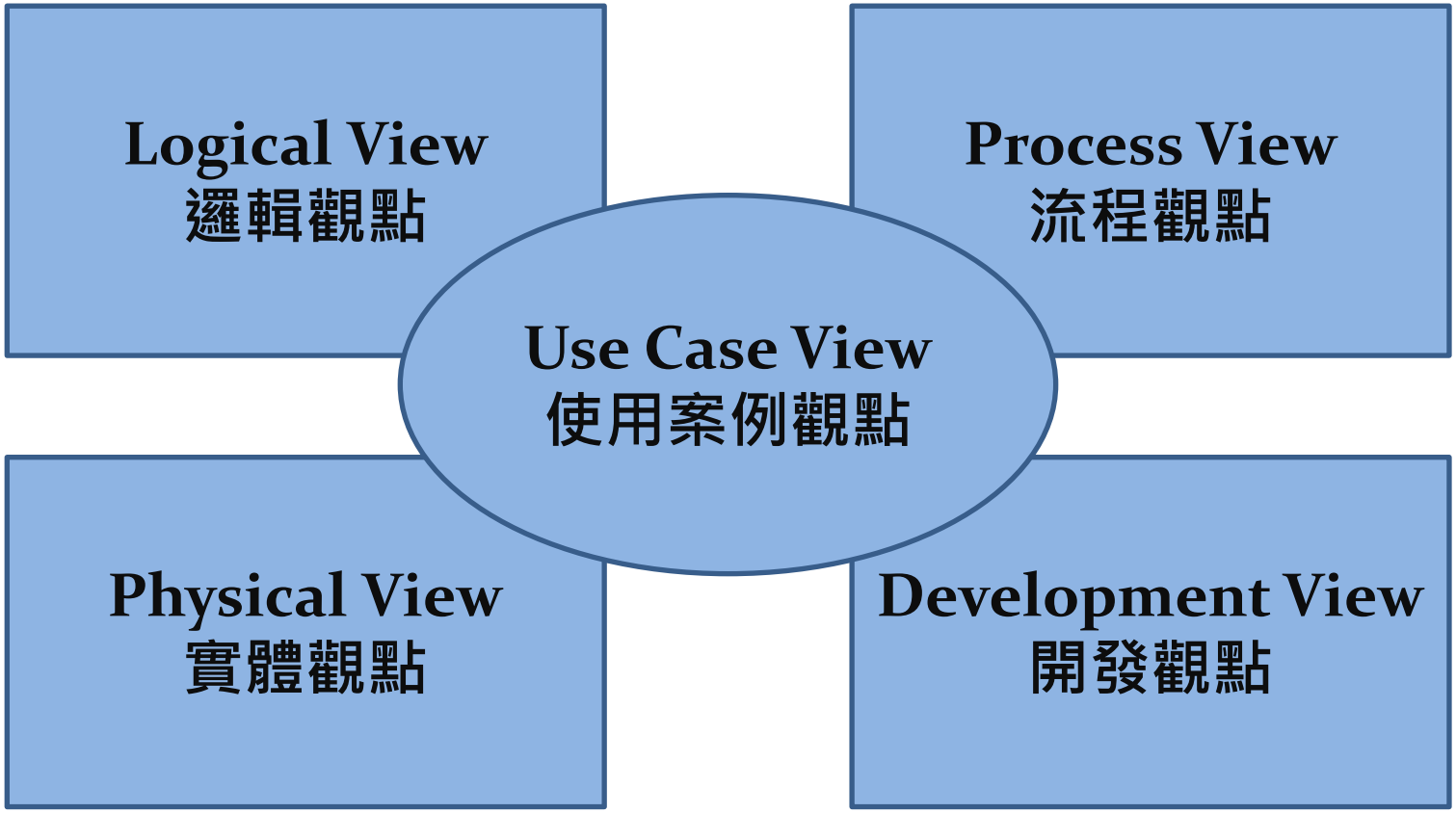
系統塑模 (cont.)


Leg
4
1
Thing
1
-size = large
Trunk
1
=
Communication + Meaning = Success!
Leg
4
1
Thing
1
-size = large
Trunk
1
=
The System Designer's Perspective
The System Stakeholder's Perspective (i.e., the customer)
The System Implementer's Perspective

圖形化塑模 (Graphical Model)

- 幫助不同的 stakeholders 瞭解系統
  - 可能不完整、或者不完全正確
- 作為系統的架構說明文件
  - 模型必須精確，但可能不完整
- 可能從詳細的塑模中直接產生實作
  - 模型必須精確而完整

系統的面向


Logical View
邏輯觀點
Process View
流程觀點
Use Case View
使用案例觀點
Physical View
實體觀點
Development View
開發觀點

Unified Modeling Language (UML)

- UML 是現今最泛用的圖形化塑模工具
- 結構性圖形 (Structure Diagram)
  - 靜態圖形 (Static Diagram)
    - **類別圖 (Class Diagram)**
    - 物件圖 (Object Diagram)
    - 套件圖 (Package Diagram)
  - 實作圖形 (Implementation Diagram)
    - 組件圖 (Component Diagram)
    - 部署圖 (Deployment Diagram)

UML (cont.)

- 結構性圖形 (cont.)
  - 剖面圖 (Profile Diagram)
  - 複合結構圖 (Composite Structure Diagram)
- 行為式圖形 (Behavioral Diagram)
  - **活動圖 (Activity Diagram)**
  - **狀態圖 (State Machine Diagram)**
  - **使用案例圖 (Use Case Diagram)**
- 交互式圖形 (Interaction Diagram)
  - **通信圖 (Communication Diagram)**
  - 交互概述圖 (Interaction Overview Diagram)
  - **時序圖 (Sequence Diagram)**
  - 時間圖 (Timing Diagram)

系統塑模

使用案例 (USE CASE)

使用案例

- 描述功能
  - 描述系統應有的行為
- 開發、設計、測試與文件的核心

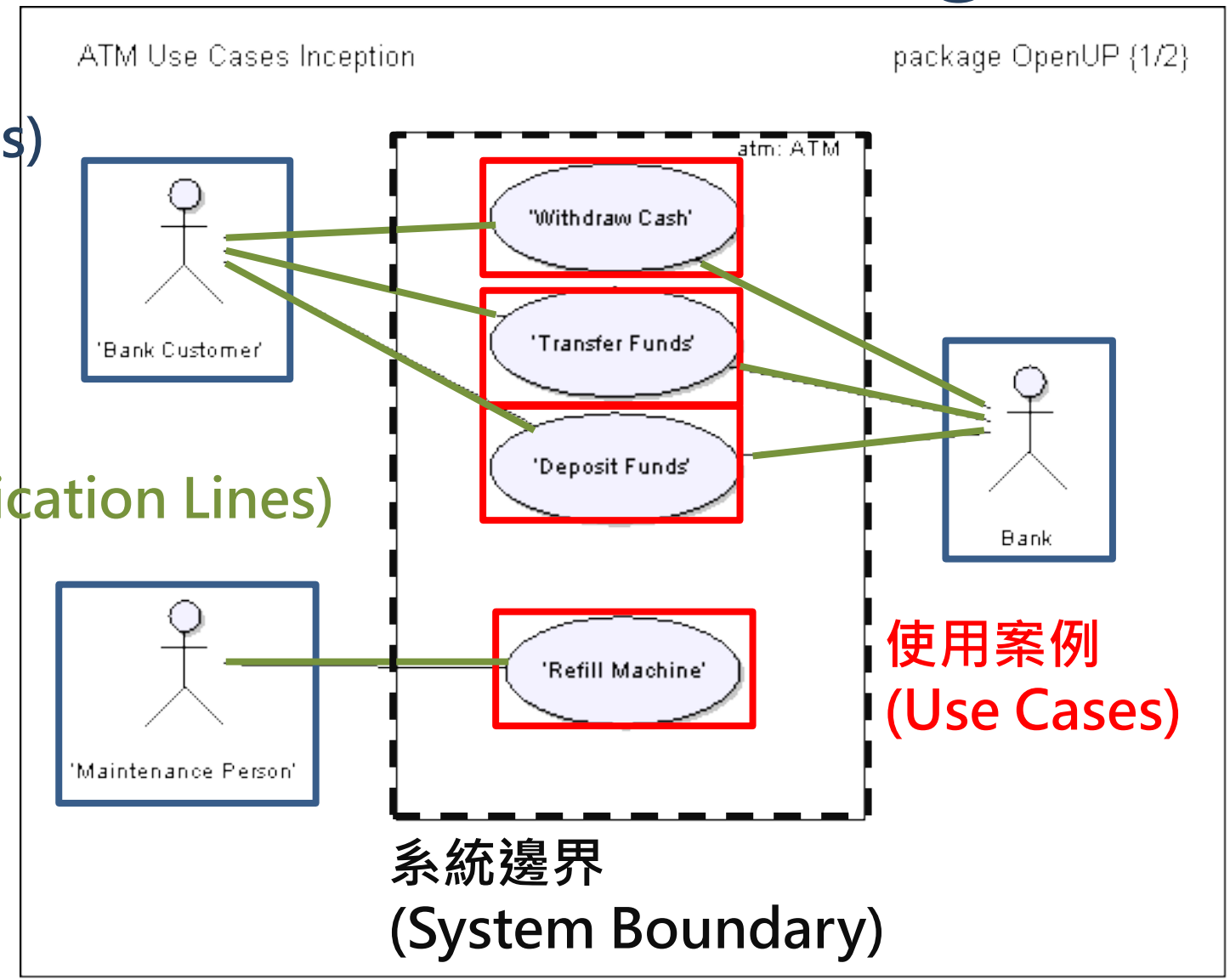
使用案例 – 定義行動者 (Actor)

- 從需求中所辨別出來的事物
  - 會與系統互動嗎？
  - 會隨系統設計而改變嗎？
- 如果第一個答案為『是』、第二個答案為『否』
- 這個事物應該是一個 行動者 (Actor)，而非系統的一部分

使用案例圖 (Use Case Diagram)

角色 (Actors)

溝通線
(Communication Lines)



使用案例
(Use Cases)

系統邊界
(System Boundary)

使用案例塑模

轉送資料使用案例

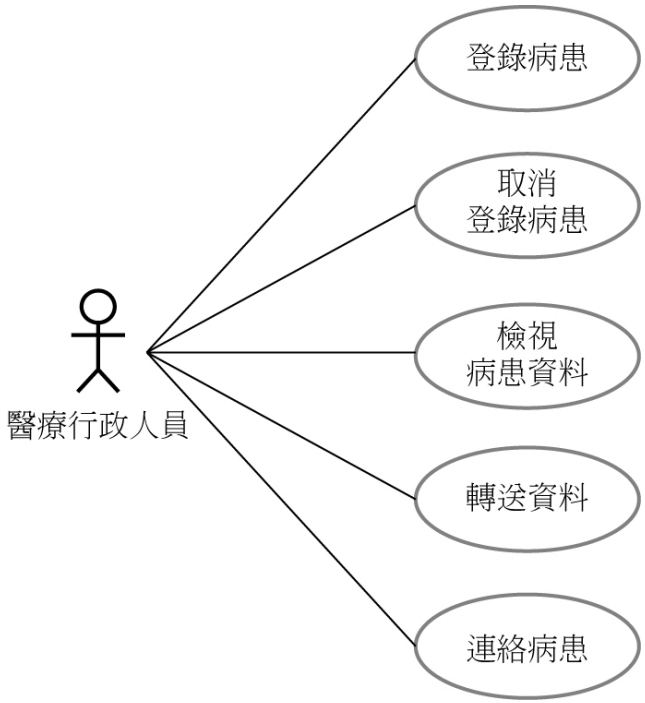
「轉送資料」使用案例的表格描述

圖5.4「轉送資料」使用案例的表格描述

MHC-PMS：轉送資料（Transfer data）	
行動者	醫療行政人員、病歷系統（PRS）
說明	醫療行政人員可能會從MHC-PMS 傳輸資料到衛生機關的總病歷資庫。被傳送的資訊可能是更正過的個人資料（住址、電話號碼等），或者是病患的診斷和治療過程摘要
資料	病患的個人資料、治療過程摘要
刺激	由醫療行政人員輸入命令
回應	確認PRS 已經更新
註解	醫療行政人員必須有適當的保全權限才能存取病患資料和PRS

從 Actor 的角度看使用案例

醫療行政人員角色的使用案例



『好』的使用案例

- 為使用者或外部系統提供可量測的結果
- 任何通過簡單的測試可以驗證的系統行為，都很適合做成使用案例
- 明確的成功或失敗的指標，使用者必須能夠清楚明白，系統是否達成了一個使用案例
- 好的使用案例 = 好的功能描述

Exercise

- 根據 Exercise 所定義的使用者需求規格，繪製選課系統的 Use Case Diagram

使用案例規格 (Use Case Specification)

- 使用案例: 提款 (Withdraw)
- 簡述
  - 這個使用案例描述銀行客戶如何利用自動櫃員機從他/她的帳戶提款
- 參與行動者
  - 銀行顧客 (Bank Customer)
  - 銀行 (Bank)

使用案例規格 (cont.)

- 基本流程 (Basic Flow)
  1. 使用案例開始於當銀行顧客將金融卡插入自動櫃員機的讀卡機
  2. 執行使用案例: 檢證使用者身份
  3. 自動櫃員機顯示服務選項
  4. 使用者選擇提款
  5. 自動櫃員機顯示要求客戶輸入提款金額
  6. 顧客輸入提款金額，提款金額為 1,000 的倍數
  7. 自動櫃員機將顧客的帳號、密碼、以及欲提款金額傳往銀行，開始一個交易(Transaction)
  8. 銀行回傳交易核准訊息，並且從客戶帳戶中扣除提領金額
  9. 吐鈔機吐出與顧客輸入金額相當之鈔票
  10. 顧客取走鈔票
  11. 讀卡機退出顧客的金融卡
  12. 顧客取走金融卡
  13. 自動櫃員機印出收據
  14. 顧客取走收據
  15. 使用案例結束，交易成功

使用案例規格 (cont.)

- 替代流程 (Alternative Flow)

2.1 於基本流程第二步，假如檢證使用者，發生無效卡片錯誤 (Invalid Card)

1. 螢幕顯示卡片無效
2. 讀卡機退出顧客的金融卡
3. 顧客取走金融卡
4. 使用案例結束，交易失敗

2.2 於基本流程第二步，假如檢證使用者，發生密碼錯誤 (Incorrect Password)，且密碼錯誤次數 < 3

1. 密碼輸入錯誤次數 + 1
2. 螢幕顯示請顧客重新輸入密碼
3. 顧客重新輸入密碼
4. 進行基本流程第二步

2.3於基本流程第二步，假如檢證使用者，發生密碼錯誤，且密碼錯誤次數 >= 3

1. 螢幕顯示密碼輸入錯誤次數過多
2. 使用案例結束，交易失敗

使用案例規格 (cont.)

- 替代流程 (cont.)

  6.1 於基本流程第六步，顧客輸入的金額並非1,000的倍數

  1. 螢幕顯示，本機僅提供千元鈔票
  2. 重複執行基本流程第六步

  7.1 在基本流程第七步前，顧客按下取消鍵

  1. 營幕顯示交易取消
  2. 讀卡機退出顧客的金融卡
  3. 顧客取走金融卡
  4. 使用案例結束，交易失敗

使用案例規格 (cont.)

- 替代流程 (cont.)
  - 8.1 於基本流程第八步，銀行回傳欲提領金額超出今日提領上限
    1. 營幕顯示欲提領金額超出今日可提領金額上限，請重新操作
    2. 讀卡機退出顧客的金融卡
    3. 顧客取走金融卡
    4. 使用案例結束，交易失敗
  - 8.2 於基本流程第八步，銀行回傳帳戶餘額不足
    1. 營幕顯示帳戶餘額不足，與客戶目前帳戶餘額
    2. 讀卡機退出顧客的金融卡
    3. 顧客取走金融卡
    4. 使用案例結束，交易失敗
  - 8.3 於基本流程第八步，銀行未在時限內回應
    1. 營幕顯示連線出現問題，請稍後再進行操作
    2. 讀卡機退出顧客的金融卡
    3. 顧客取走金融卡
    4. 使用案例結束，交易失敗

使用案例規格 (cont.)

- 替代流程 (cont.)

  10.1 於基本流程第十步，顧客未取走鈔票

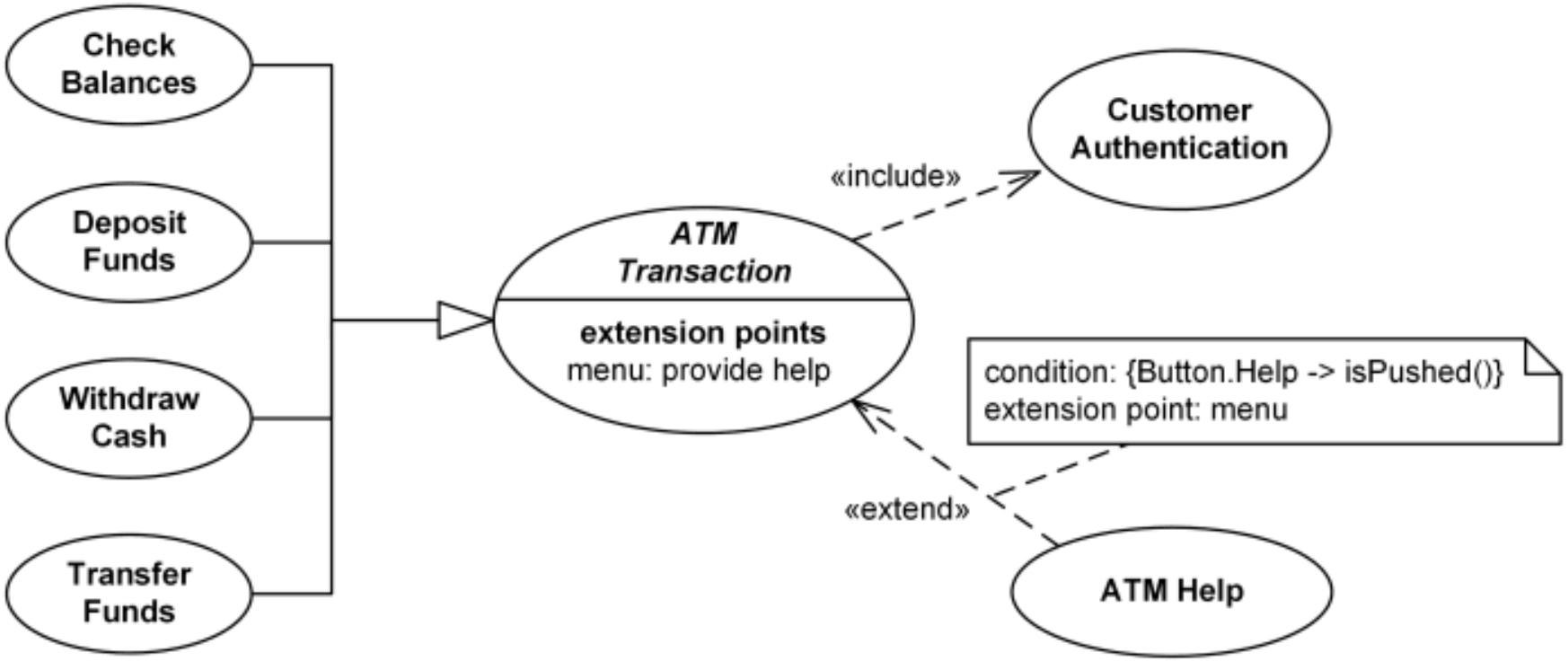
  1. 自動櫃員機發出警示聲，並在螢幕上顯示，鈔票將在 30 秒後收回
  2. 30秒後，顧客仍未取走鈔票，自動櫃員機將鈔票收回，向銀行啟動滾回程序，回復客戶帳戶金額
  3. 使用案例結束，交易失敗

- 後置條件
  - 若使用案例結束，交易成功，則顧客獲得相當於其輸入金額之現金，款項亦同時從其帳戶扣除
  - 若使用案例結束，交易失敗，則自動櫃員機內鈔票需與使用案例開始前相同，顧客帳戶之金額需與使用案例開始前相同

使用案例之間的關係

- 引用 (Include)
- 普遍化 (Generalization)
- 延伸 (Extend)
  - Optional 或 conditional 的 include

使用案例之間的關係 (cont.)



Exercise

- 從選課系統的 Use Case Diagram 中選擇一項非登入/登出的 Use Case，描寫其 Use Case Specification

系統塑模

穩健圖 (ROBUSTNESS DIAGRAM)

穩健圖 (Robustness Diagram)

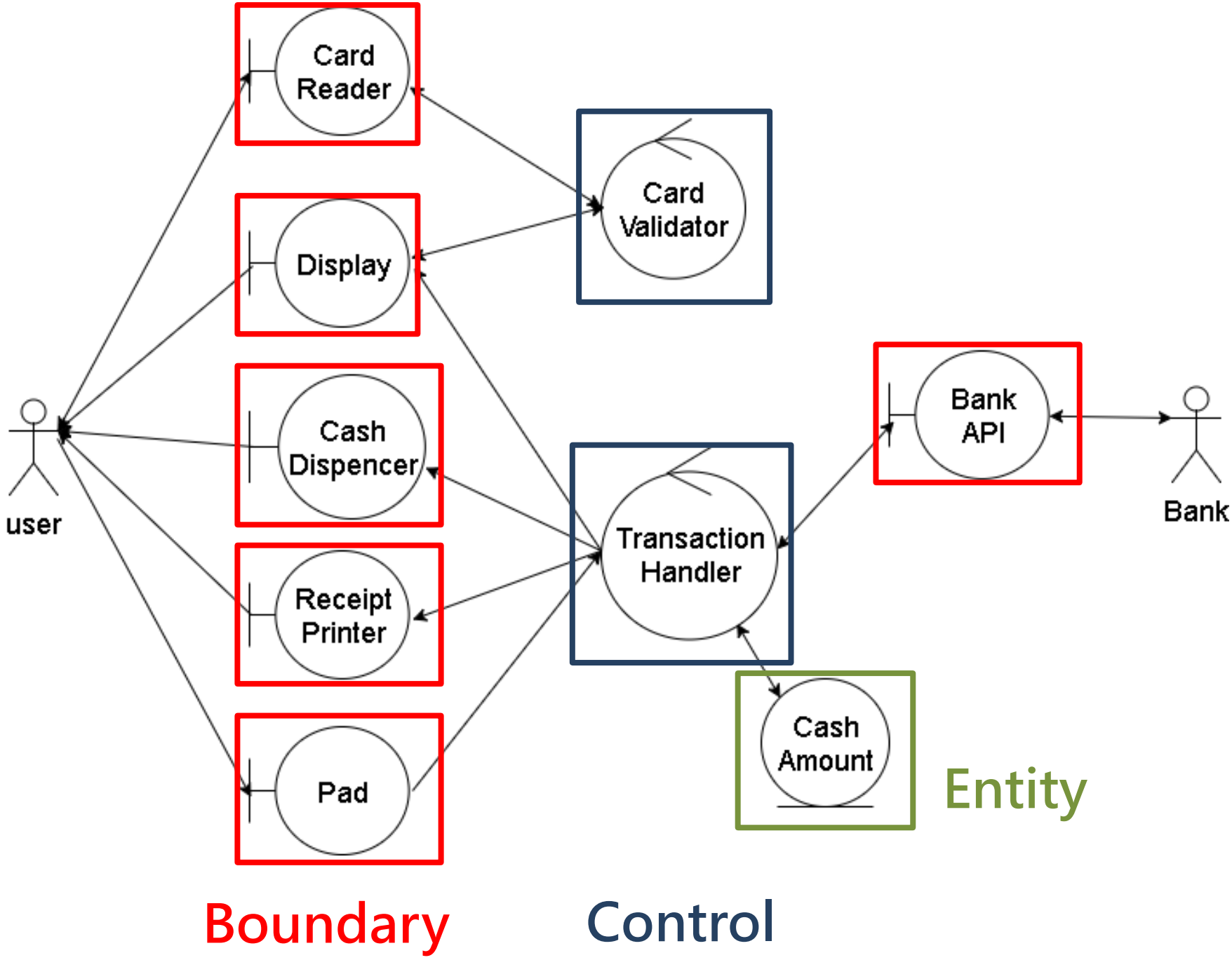
- 邊界物件 (Boundary)
  - 與 Actor 互動
  - 代表外部元素與系統互動的關係
- 控制物件 (Control)
  - 代表系統的動態行為
  - Use Case 中，系統應有的規則與邏輯
- 實體物件 (Entity)
  - 泛指系統會存取的資料

穩健圖 (cont.)

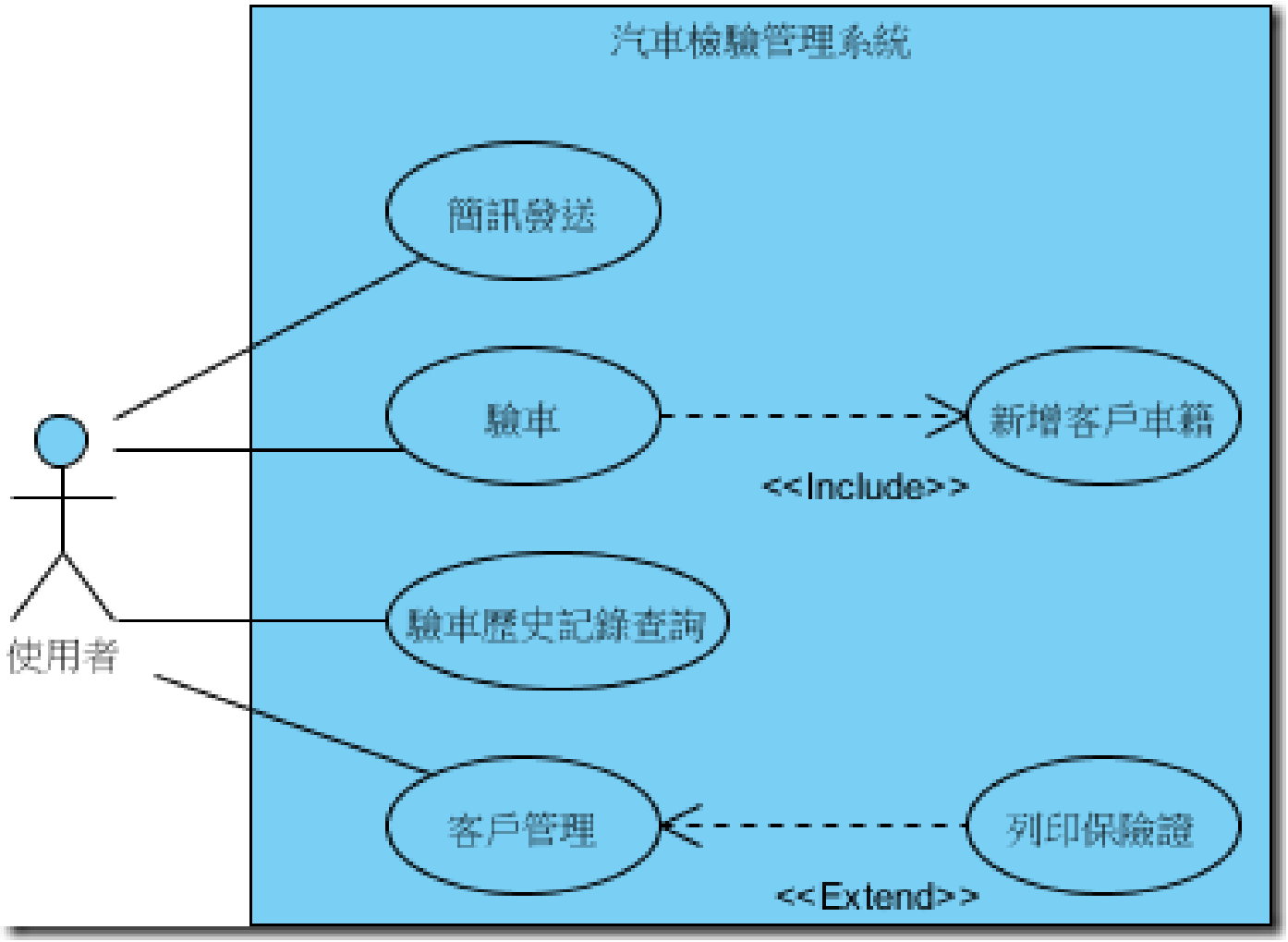
- 基本限制
  - Boundary 與 Entity 只能透過 Control 交談 (Interaction)
  - Entity 與 Entity 或 Boundary 與 Boundary 間，也得透過 Control
  - Actor 只能與 Boundary 互動

範例：從使用案例到穩健圖

- 基本流程 (Basic Flow)
  1. 使用案例開始於當銀行顧客將金融卡插入自動櫃員機的讀卡機
  2. 執行使用案例: 檢證使用者身份
  3. 自動櫃員機顯示服務選項
  4. 使用者選擇提款
  5. 自動櫃員機顯示要求客戶輸入提款金額
  6. 顧客輸入提款金額，提款金額為 1,000 的倍數
  7. 自動櫃員機將顧客的帳號、密碼、以及欲提款金額傳網銀行，開始一個交易(Transaction)
  8. 銀行回傳交易核准訊息，並且從客戶帳戶中扣除提領金額
  9. 吐鈔機吐出與顧客輸入金額相當之鈔票
  10. 顧客取走鈔票
  11. 讀卡機退出顧客的金融卡
  12. 顧客取走金融卡
  13. 自動櫃員機印出收據
  14. 顧客取走收據
  15. 使用案例結束，交易成功

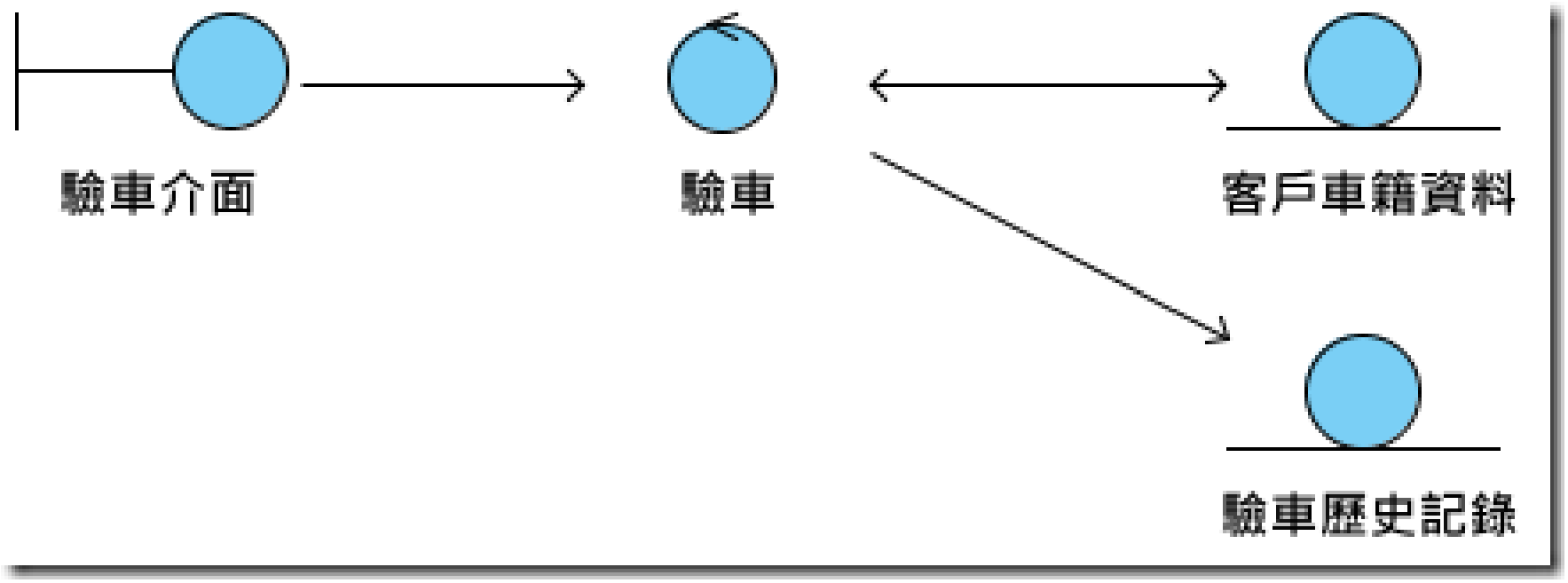

Card Reader
Display
Cash Dispencer
Receipt Printer
Pad
Card Validator
Transaction Handler
Cash Amount
Bank API
user
Bank
Boundary
Control
Entity

穩健圖分析範例

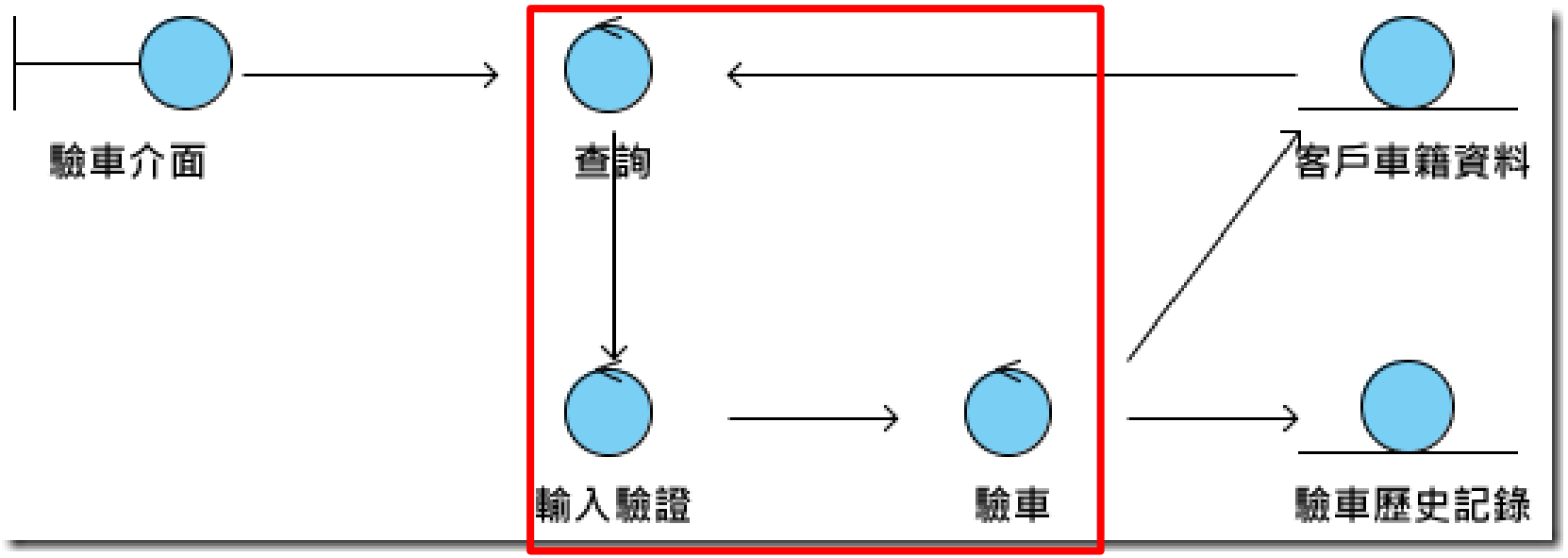


試著畫畫看驗車系統的穩健圖

穩健圖分析範例 (cont.)

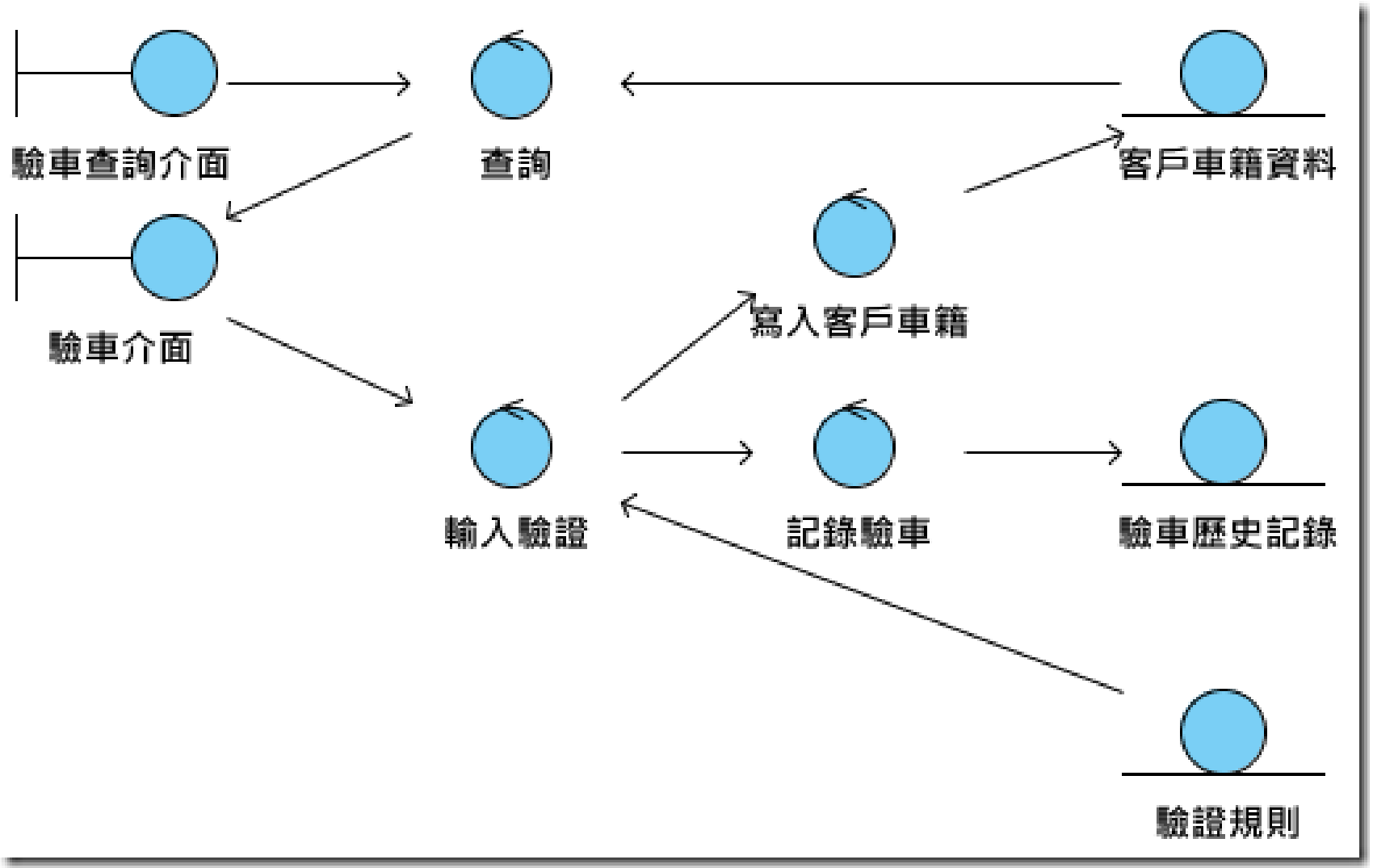


穩健圖分析範例 (cont.)



為何拆成三個物件？

穩健圖分析範例 (cont.)

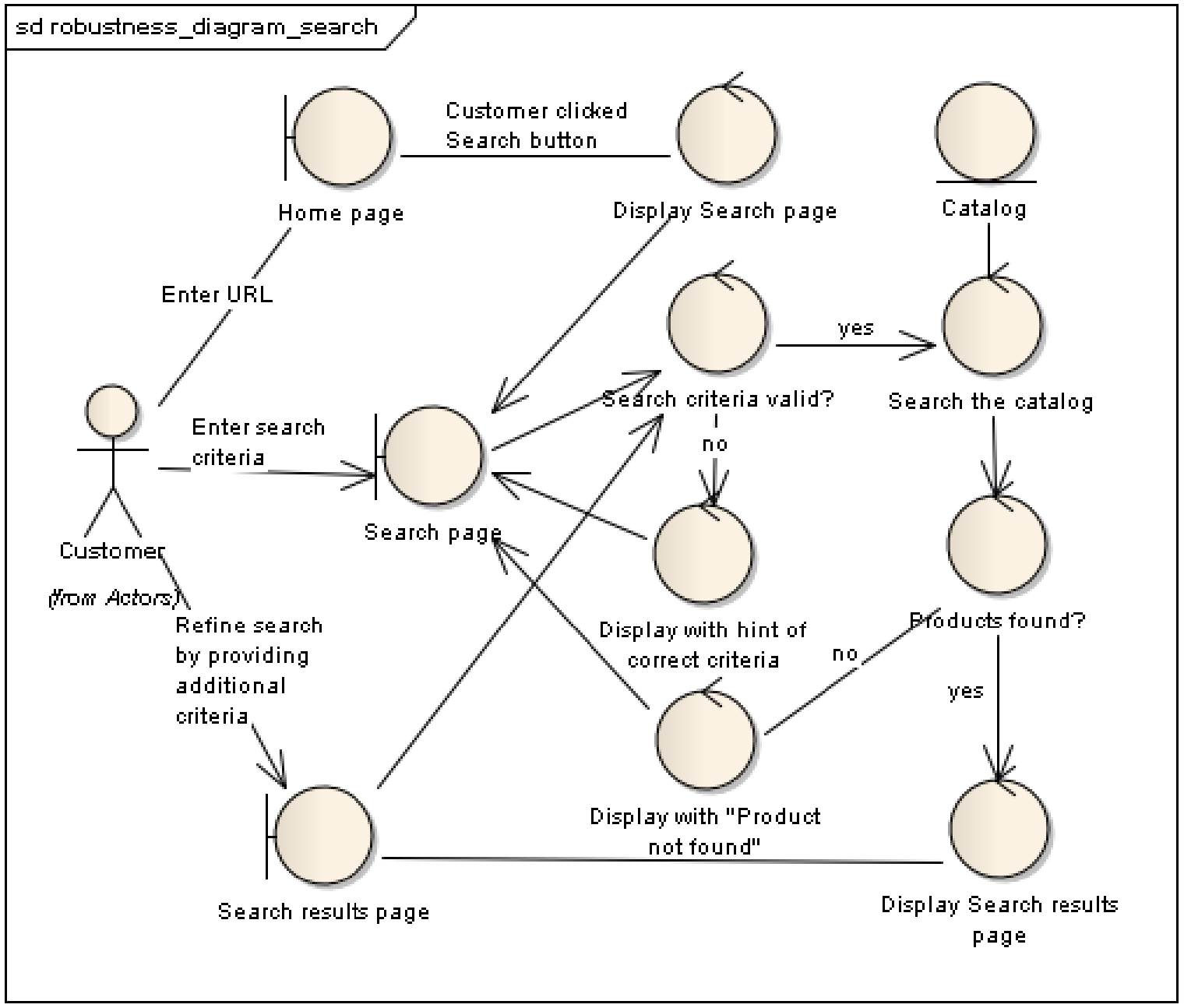


穩健圖與使用案例

Use case name	Search Products
Actors	Customer
Description	Search for products based on some criteria.
Trigger	The customer wants to browse among products or the customer would like to search for certain products.
Pre-condition	Customer starts a web browser.
Post-condition	Search results meeting the criteria are displayed.

穩健圖與使用案例 (cont.)

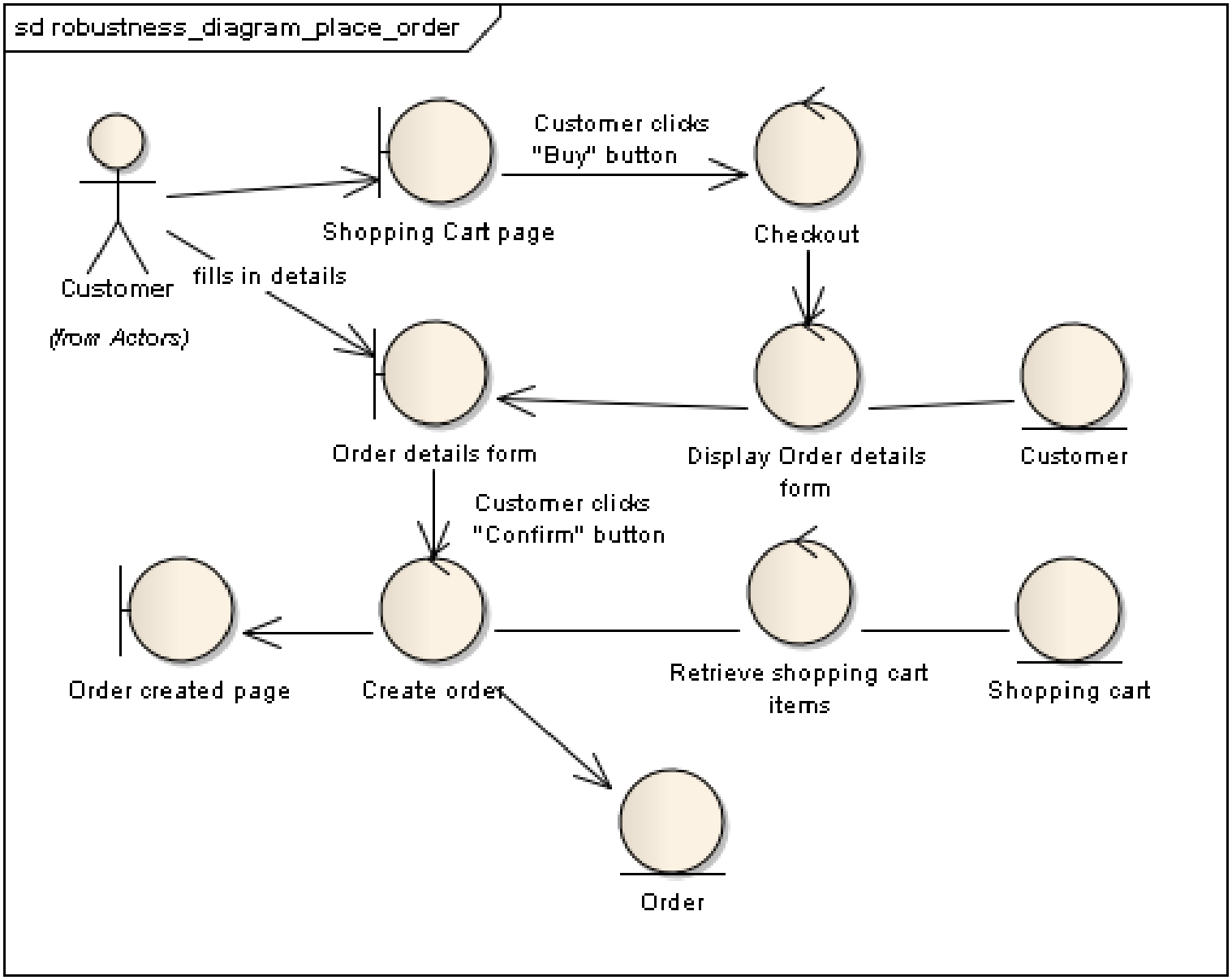
Normal flow	1.Customer visits application home page. 2.Customer clicks Search button. 3.Search page is displayed by the system. 4.Customer enters search criteria. 5.The system validates the criteria provided. 6.The system looks up the catalog to find the products that meet the criteria. 7.Search results page is displayed with the products fulfilling the criteria.
Alternative flows	Refine Search Results The following steps are added: 8.Customer refines search results by providing additional criteria. 9.Steps 5–7 are re-executed.
Exceptions	In Step 5, if search criteria is invalid or even missing then Step 3 (display search page) will be executed along with some hints on valid criteria. In Step 6, if no products meet the criteria then Step 3 (display search page) will be executed along with providing the error message "Product not found".


sd robustness_diagram_search
Customer clicked
Search button
Home page
Display Search page
Catalog
Enter URL
yes
Search criteria valid?
Search the catalog
Enter search
criteria
no
Search page
Customer
(from Actors)
Products found?
Display with hint of
correct criteria
Refine search
by providing
additional
criteria
no
yes
Display with "Product
not found"
Search results page
Display Search results
page

穩健圖與使用案例

Use case name	Place Order
Actors	Customer
Description	Customer places an order to purchase certain products from the shop.
Trigger	Customer wants to order certain products from the shop.
Precondition	Customer is logged in.
Postcondition	The system saves the new order.

Normal flow	1.Customer visits the Shopping cart page. 2.Customer clicks the Buy button. 3.The system displays the Order details form where the required data to complete the order (name, phone number, email, shipping and billing addresses, payment method) needs to be provided. 3.a The system will retrieve information of the customer in order to populate a list of default values 4. Customer fills in data and submits Order details form by clicking the Confirm button. 5. The system records the order. 5.a. It retrieves shopping cart elements to add them to the order. 6. The system displays the Order created page.
Alternative flows	None
Exceptions	1. The system cannot save the order due to a database failure. 2. The customer can cancel the order at any time before confirming it.
Note	1. The customer cannot move items back into the shopping cart. 2. If the order is cancelled, its items will also be lost. 3. The system calculates and re-calculates the total price, discounts include, for the order according to the amount changes.


sd robustness_diagram_place_order
Customer
(from Actors)
Shopping Cart page
Customer clicks
"Buy" button
Checkout
fills in details
Order details form
Display Order details
form
Customer
Customer clicks
"Confirm" button
Order created page
Create order
Retrieve shopping cart
items
Shopping cart
Order

Exercise

- 從選課系統的 Use Case Diagram 中選擇一項非登入/登出的 Use Case，根據其 Use Case Specification 繪製 Robustness Diagram

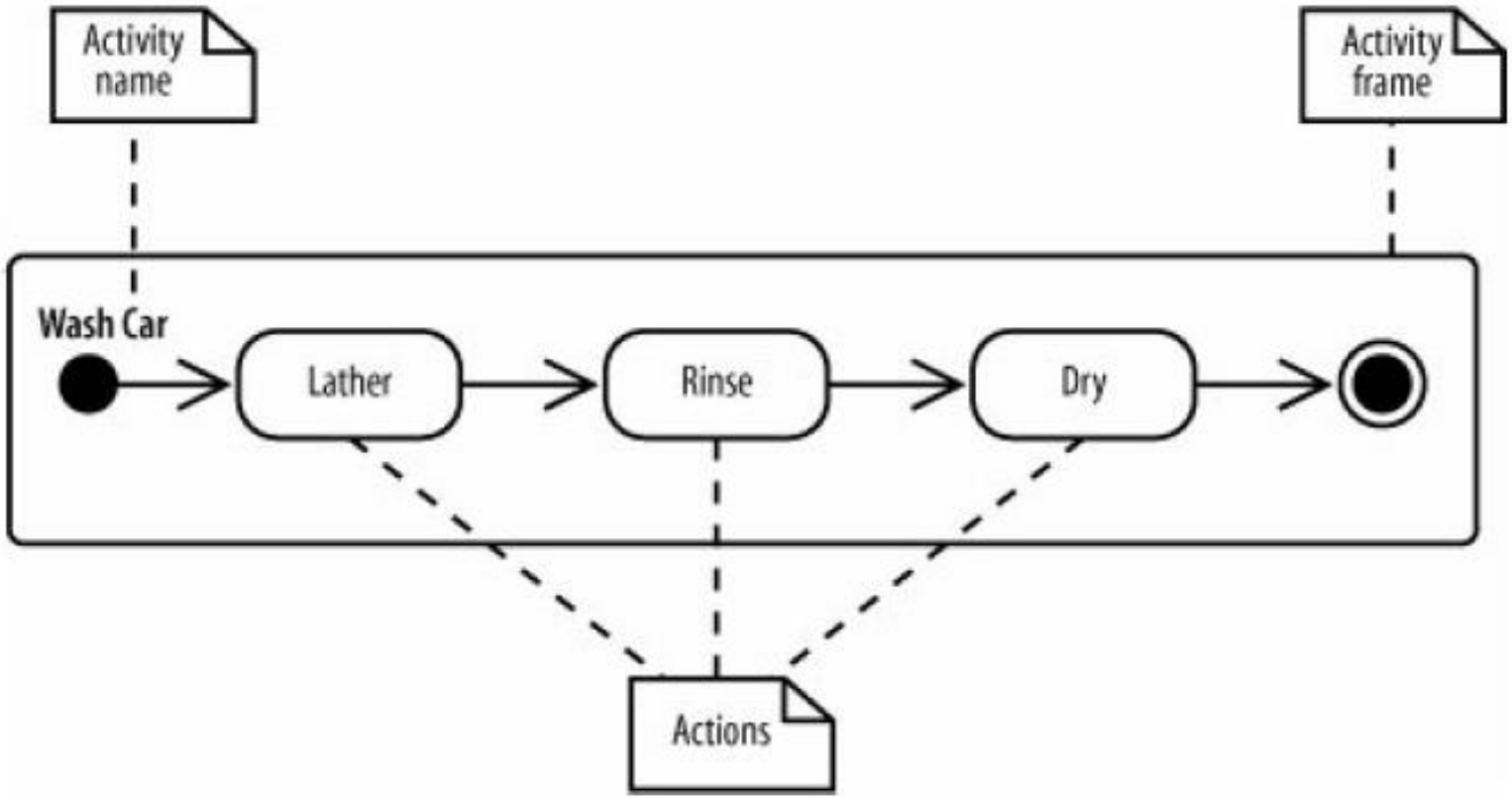
系統塑模

活動圖 (ACTIVITY DIAGRAM)

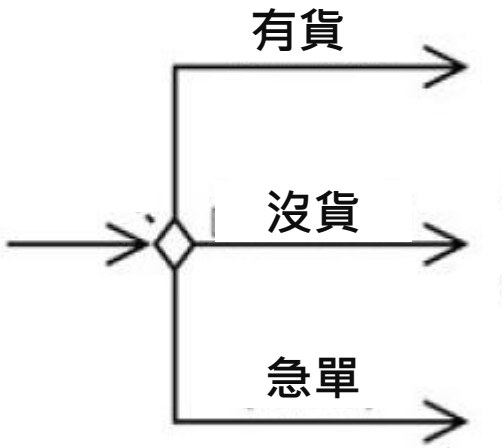
活動圖 (Activity Diagram)

- Use Case: 系統該做什麼 (What)
- Activity Diagram: 系統如何達成目標 (How)
- 描述業務流程 (Business Process)
  - 完成一連串的工作 (Task, Activity)以達成業務目標 (Business Goal)

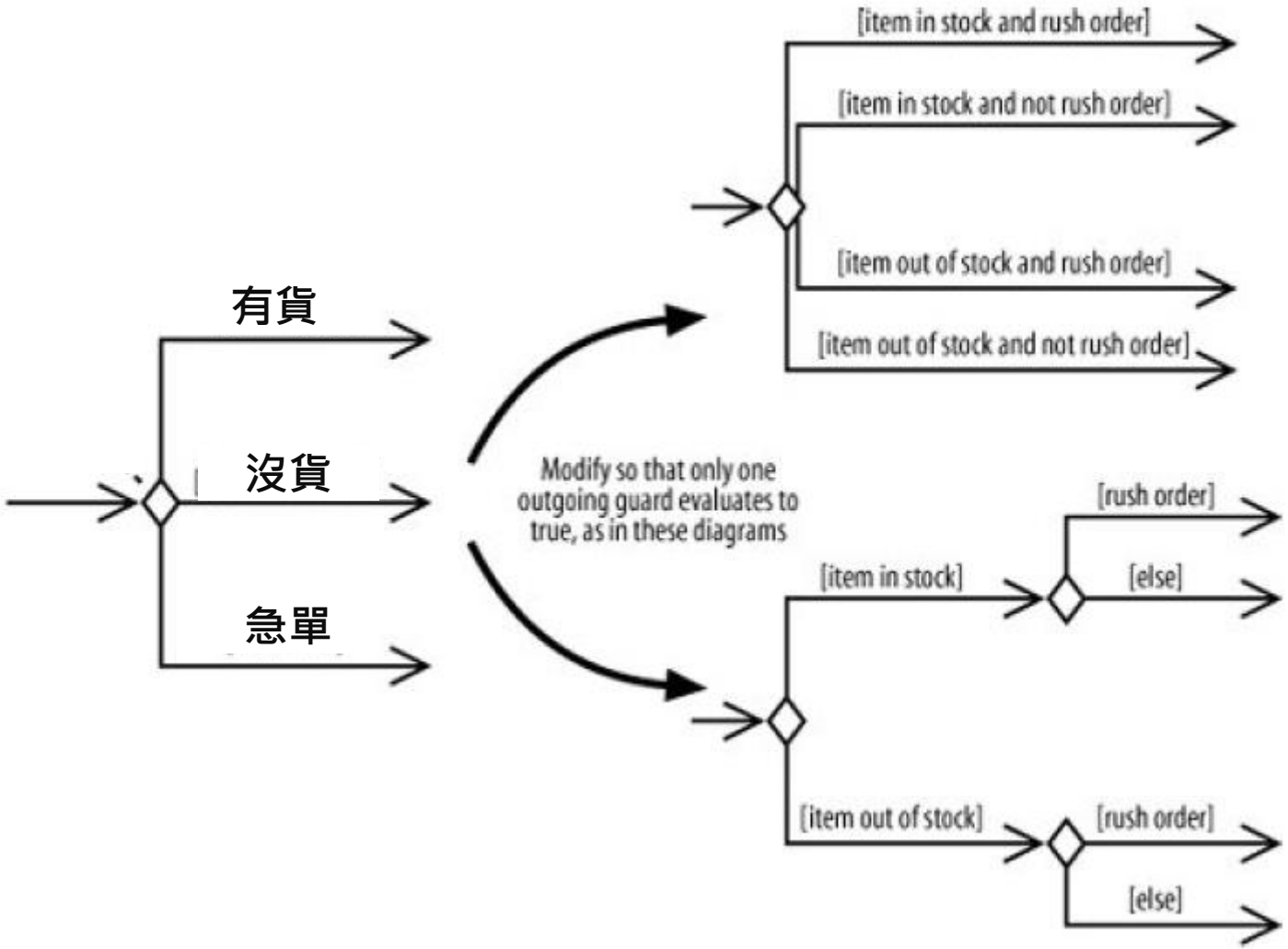
活動圖 (cont.)


Activity name
Activity frame
Wash Car
Lather
Rinse
Dry
Actions

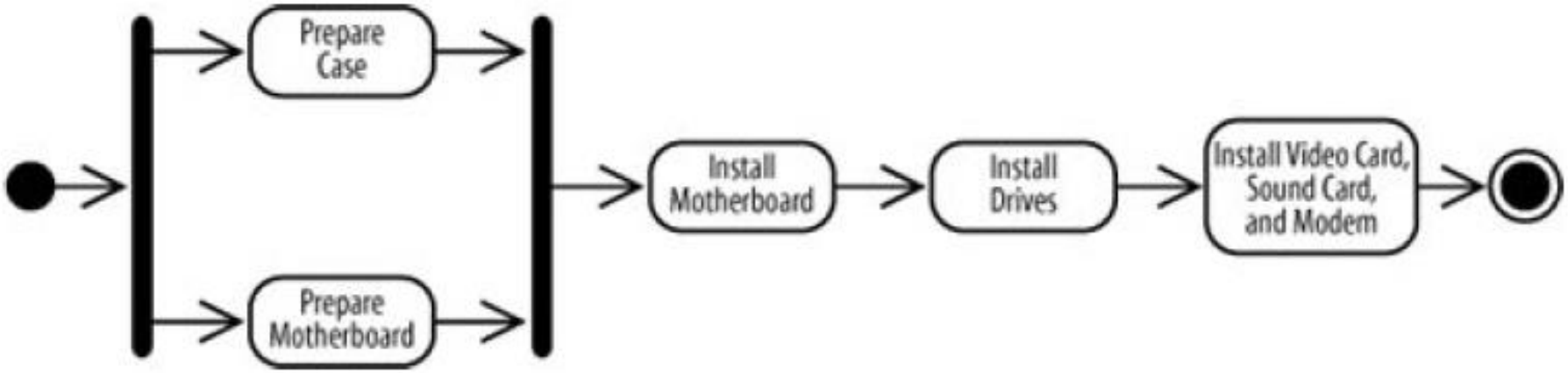
活動圖 - branch



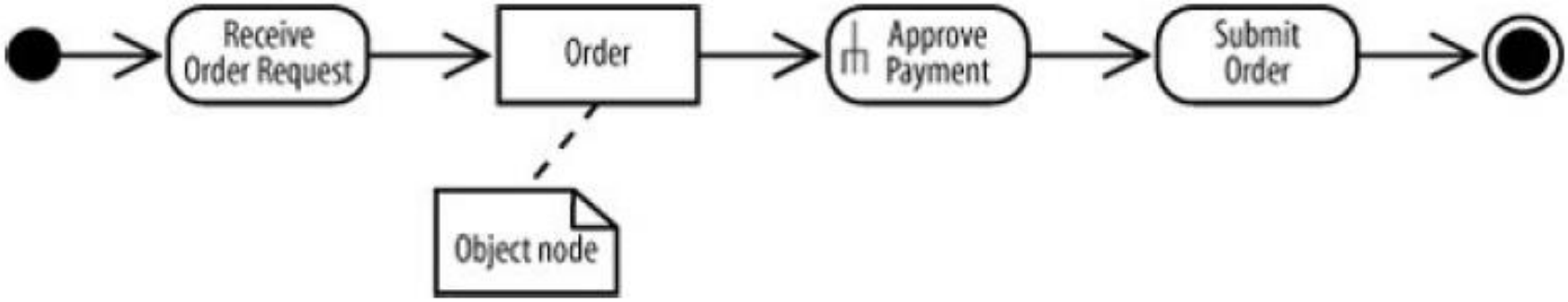
活動圖 – branch (cont.)

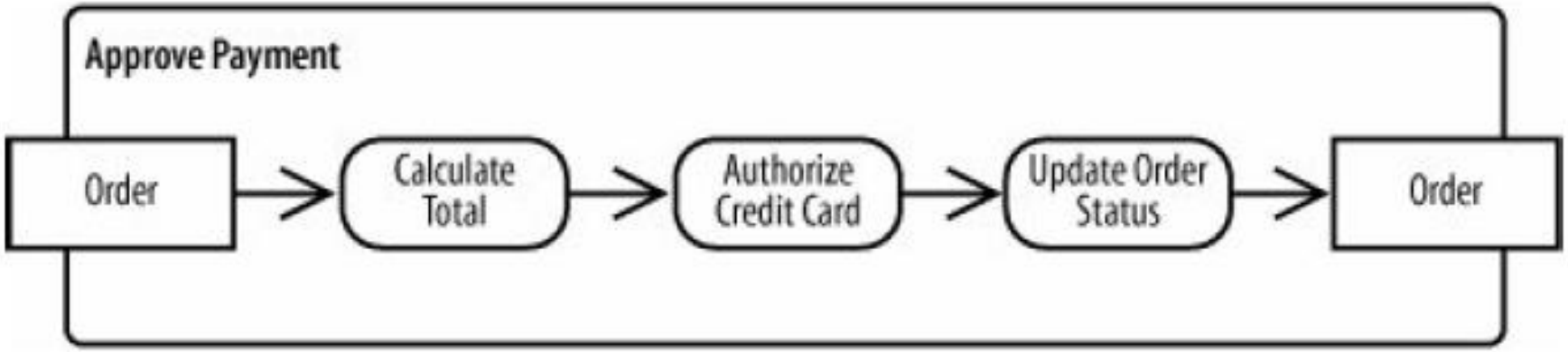


活動圖 fork, join (cont.)



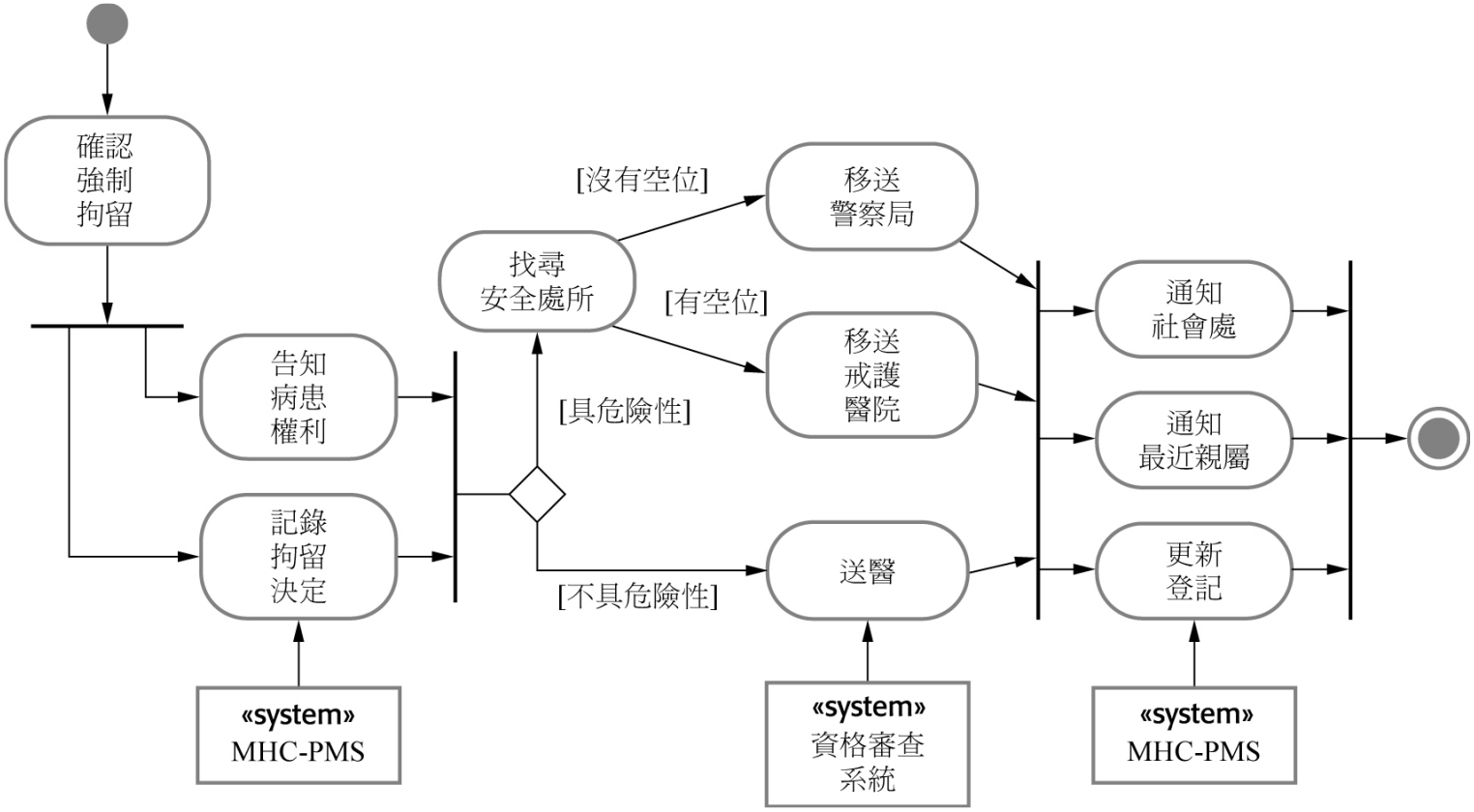
活動圖 (cont.)


Receive Order Request
Order
Approve Payment
Submit Order
Object node


Approve Payment
Order
Calculate Total
Authorize Credit Card
Update Order Status
Order

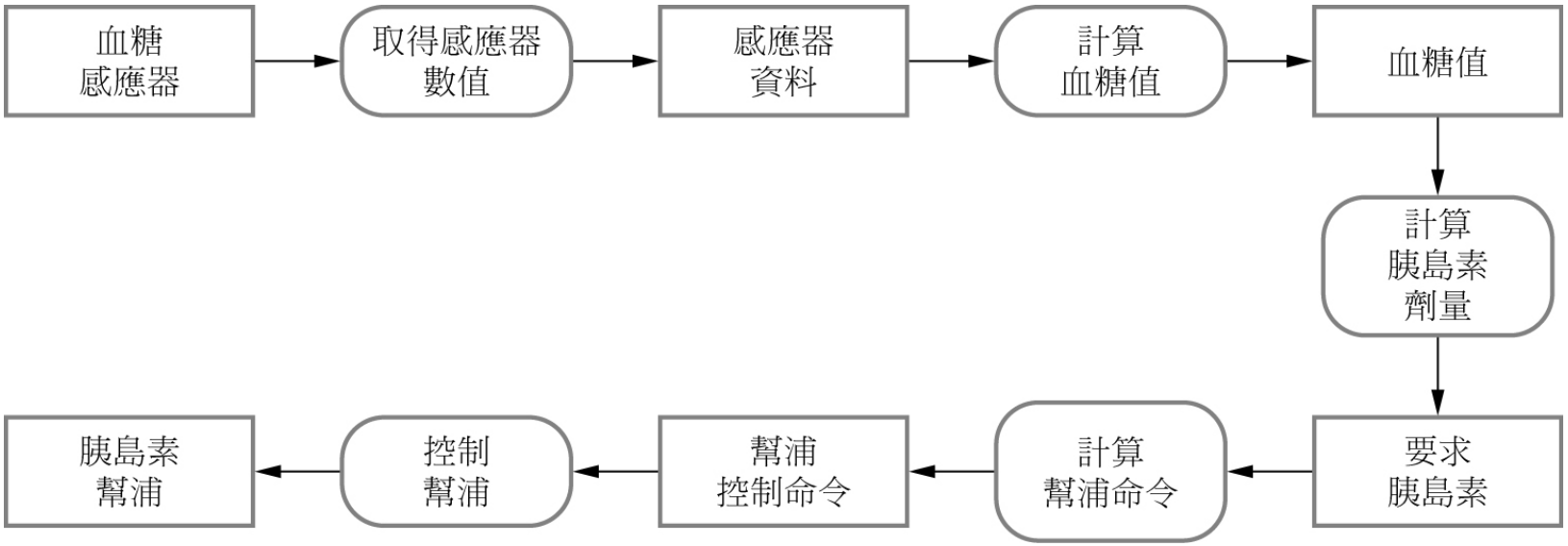
程序模型 (Process Model)

強制拘留的活動圖 => 程序模型 => Control Flow



程序模型 (cont.)

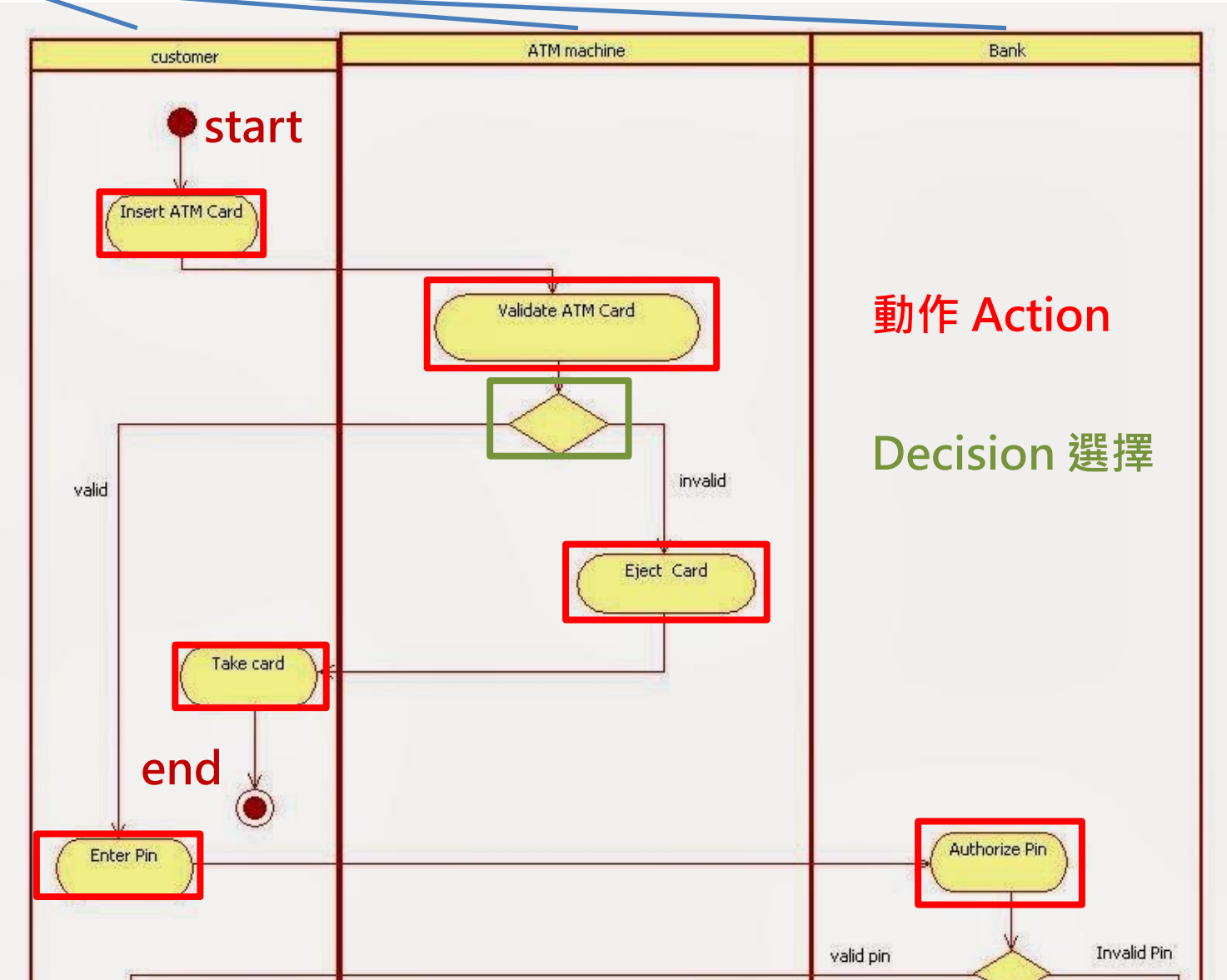
胰島素給藥的活動圖 => 程序模型 => Data Flow

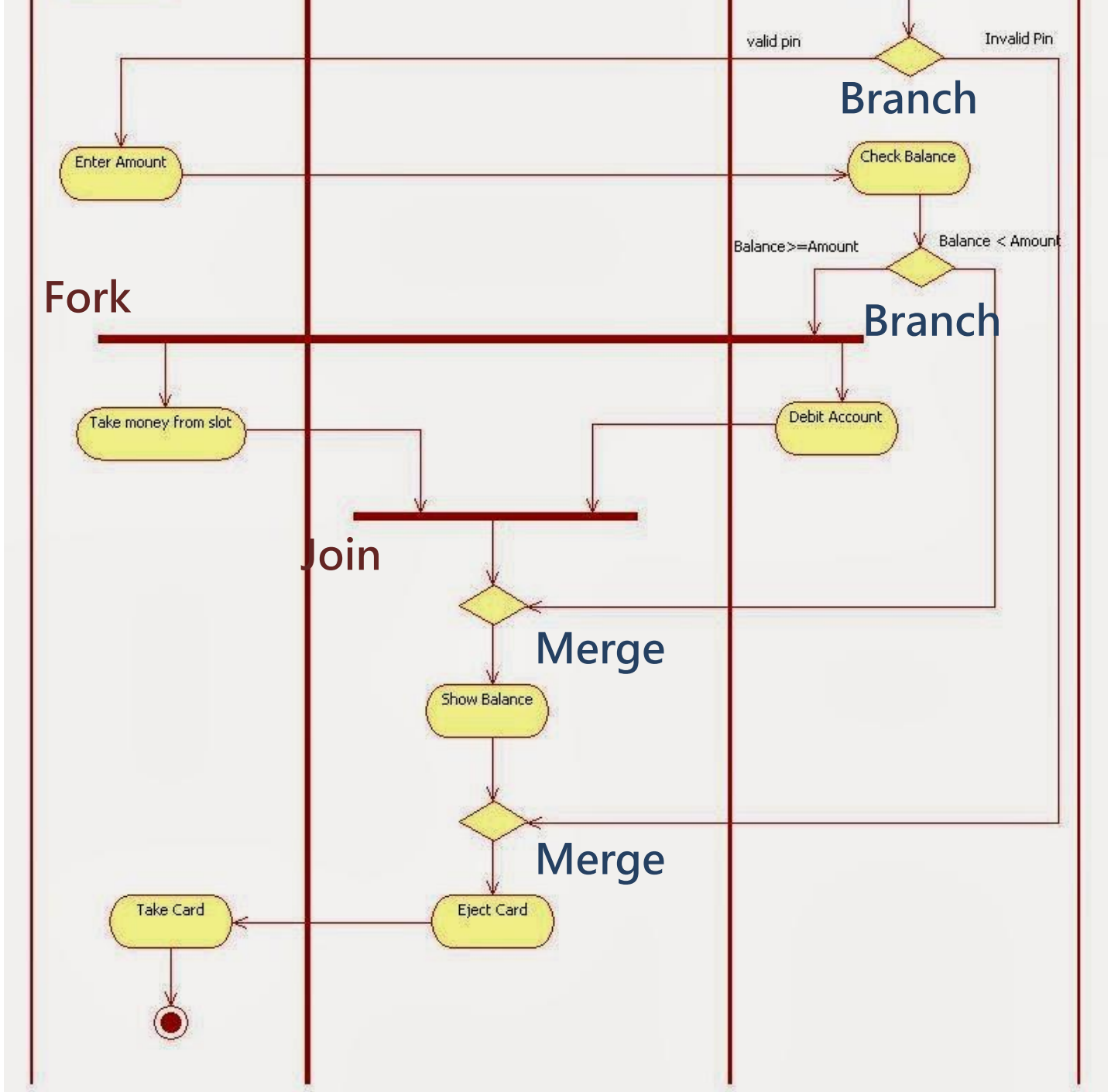


試著畫出 ATM 提款流程的活動圖

活動圖 (cont.)

水道 Swimlane




valid pin
Invalid Pin
Branch
Enter Amount
Check Balance
Balance>=Amount
Balance < Amount
Fork
Branch
Take money from slot
Debit Account
Join
Merge
Show Balance
Merge
Take Card
Eject Card

Exercise

- 挑選選課系統其中一項 Use Case、根據其 Use Case Specification，繪製活動圖

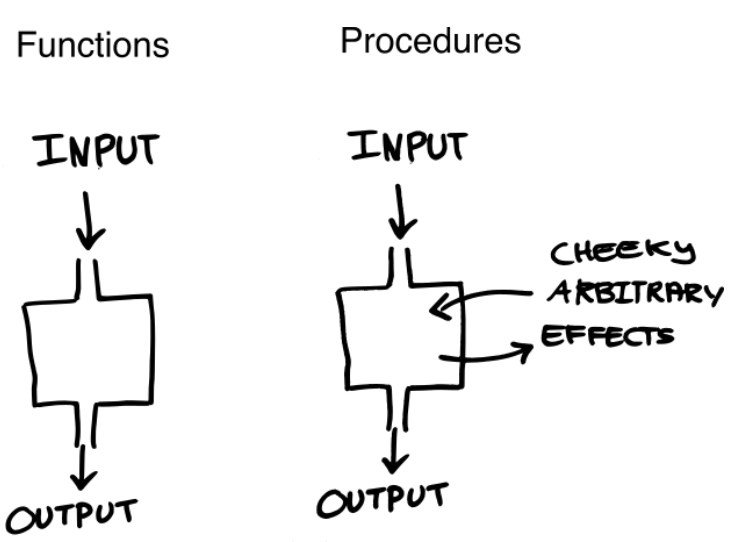
系統塑模

物件導向概念簡介

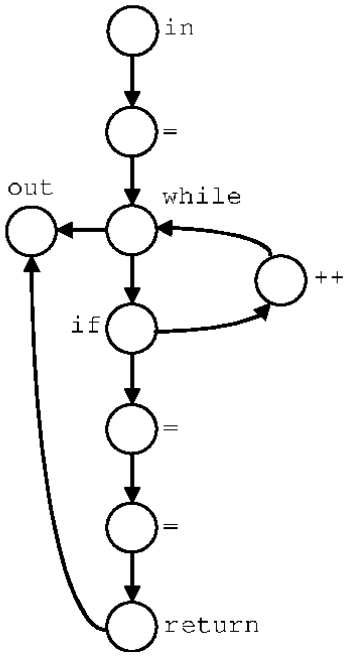
(INTRODUCTION TO OBJECT-ORIENTED)

傳統 Programming

Procedural Programming & Control Flow

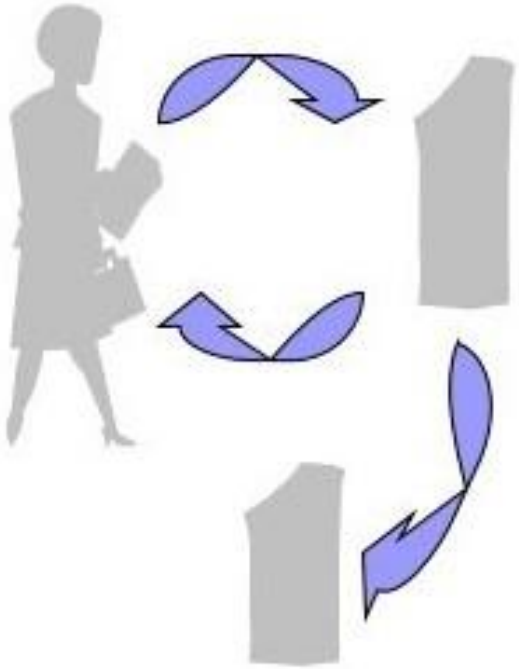



```
void Search(int arr[], int key,
            int *found, int *index)
{
  int i = 0;
  int b;

  *found = 0;

  while (i < N)
  {
    if (b = isabsequal(arr[i], key))
    {
      *found = b;
      *index = i;
      return;
    }

    i++;
  }
}
```



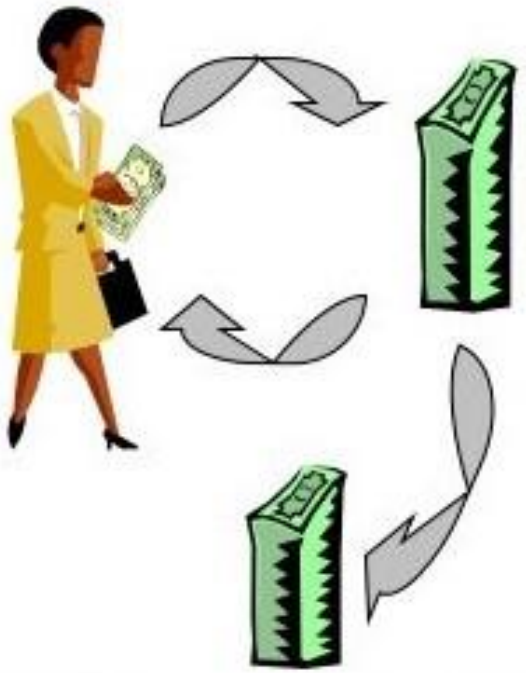
Procedural vs. Object-Oriented

- Procedural



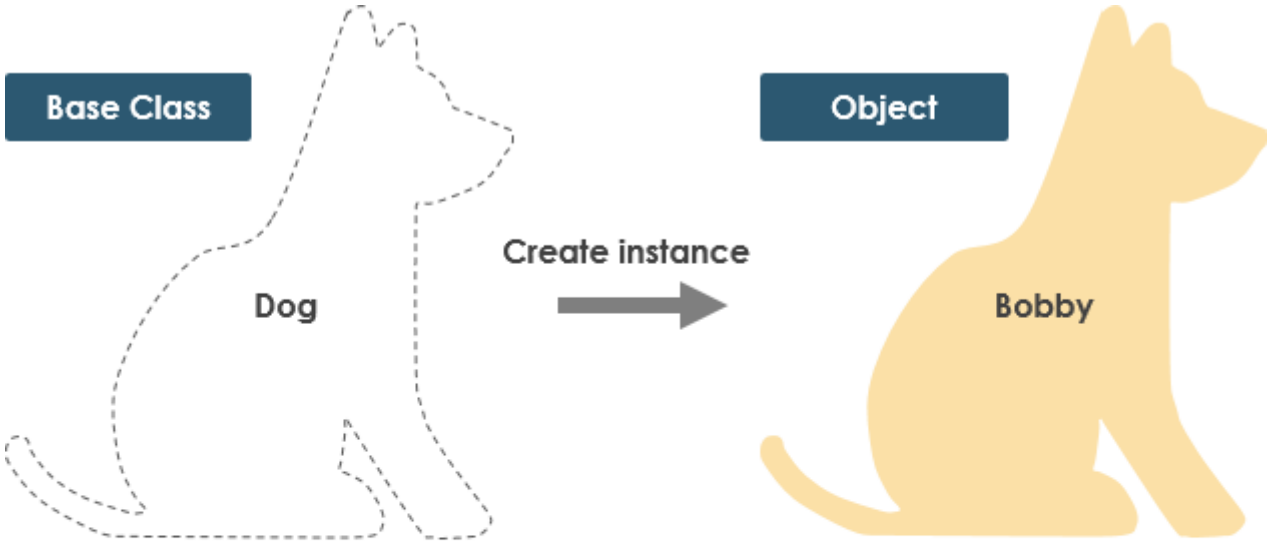
Withdraw, deposit, transfer

- Object Oriented



Customer, money, account

Class & Object



Properties	Methods
Color	Sit
Eye Color	Lay Down
Height	Shake
Length	Come
Weight	

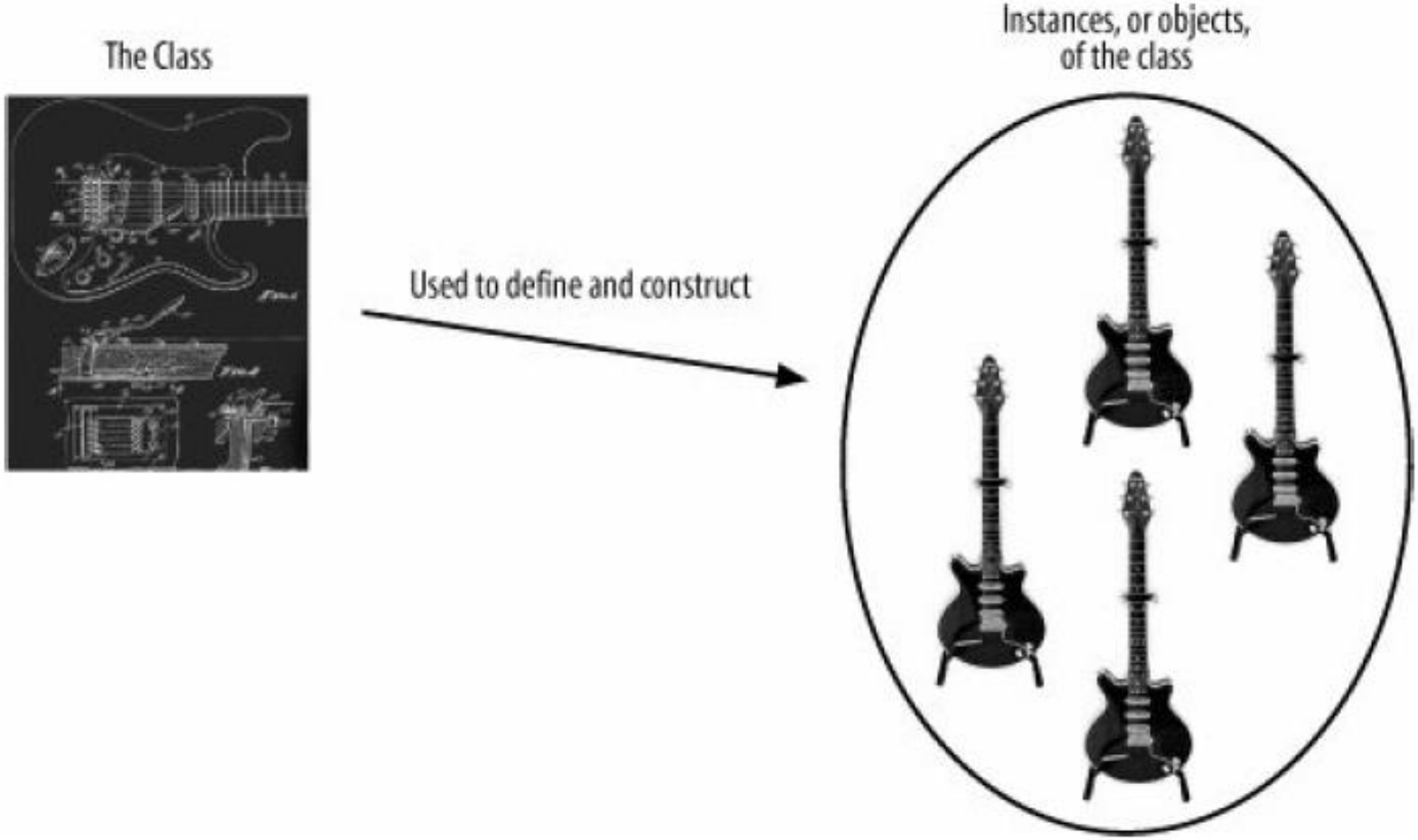
Property Values	Methods
Color: Yellow	Sit
Eye Color: Brown	Lay Down
Height: 17 in	Shake
Length: 35 in	Come
Weight: 24 pounds	

Encapsulation (封裝)

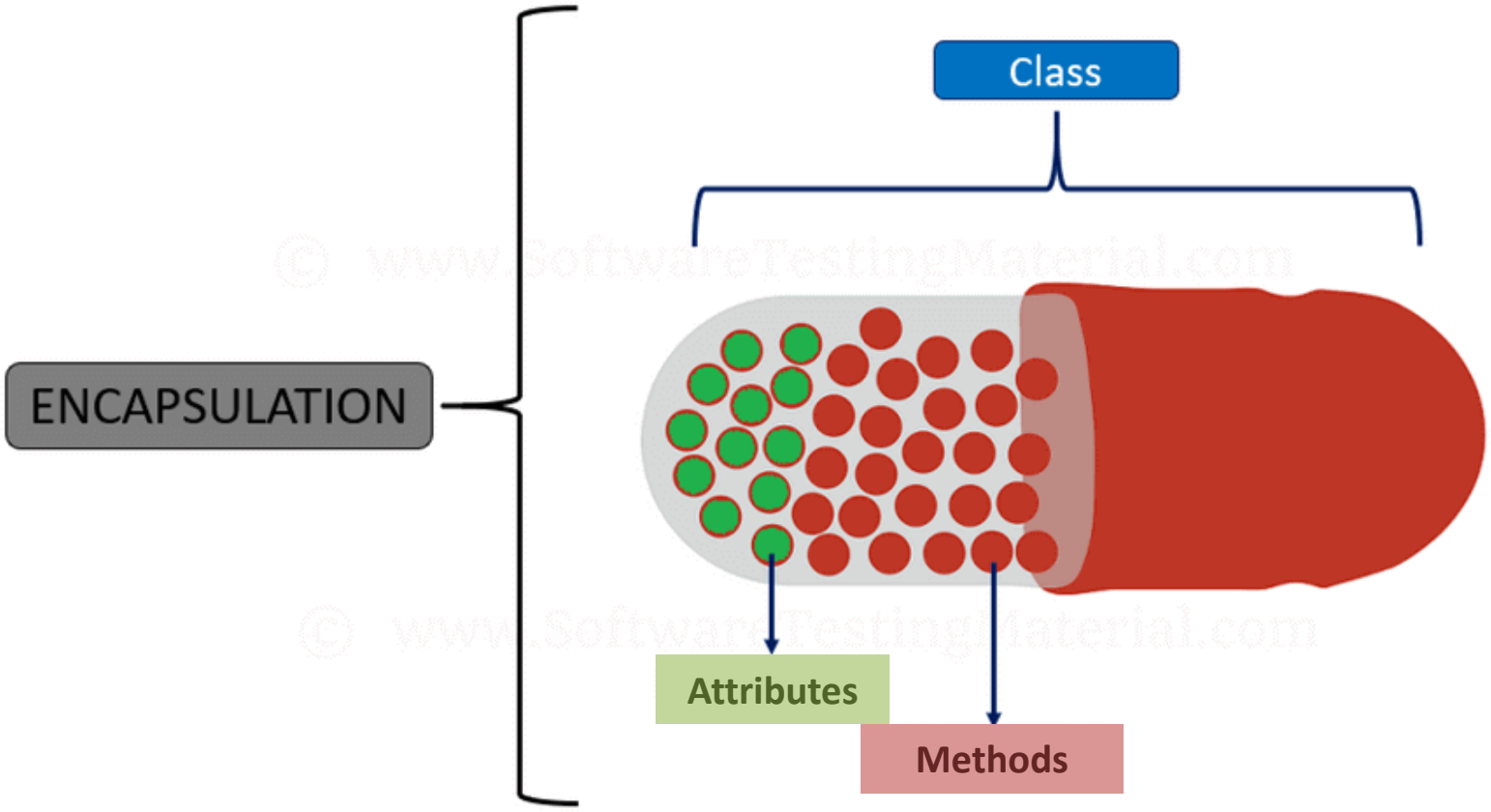
Class & Object (cont.)

- Class (類別)
  - 物件的定義，內容包含了動作 (operation) 與資料 (data)
- Object (物件)
  - 描述軟體行為的基本單位，軟體藉由物件本身的動作以及物件間的互動而運作
- Instance (實體)
  - 由一個類別所產生的物件，稱為該類別的實體

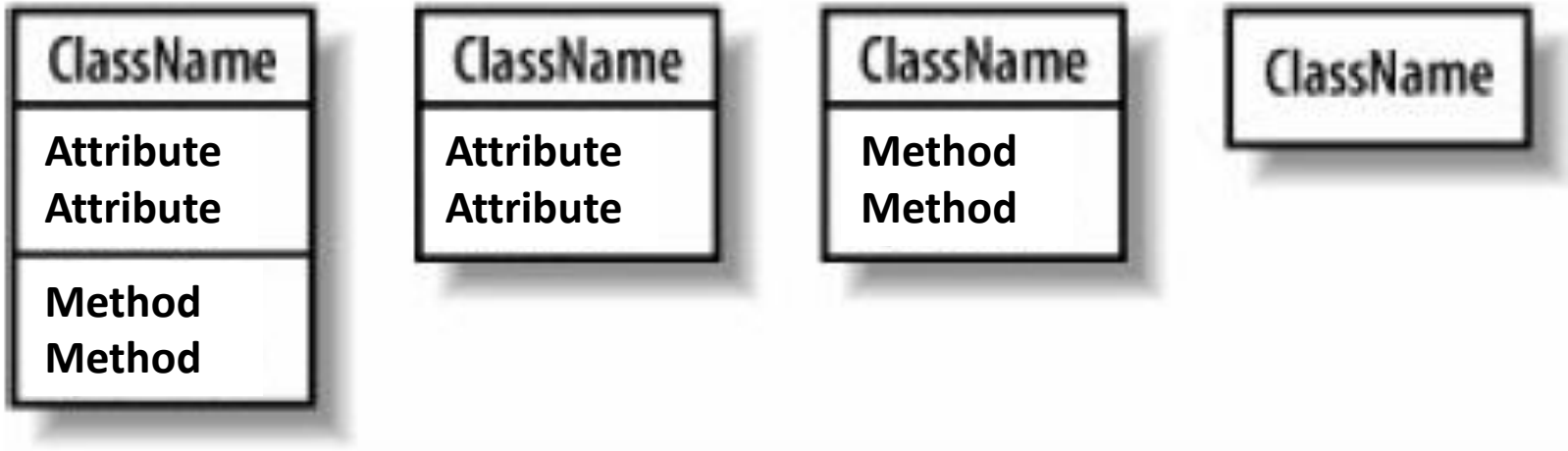
Class & Object (cont.)



Encapsulation



Encapsulation (cont.)



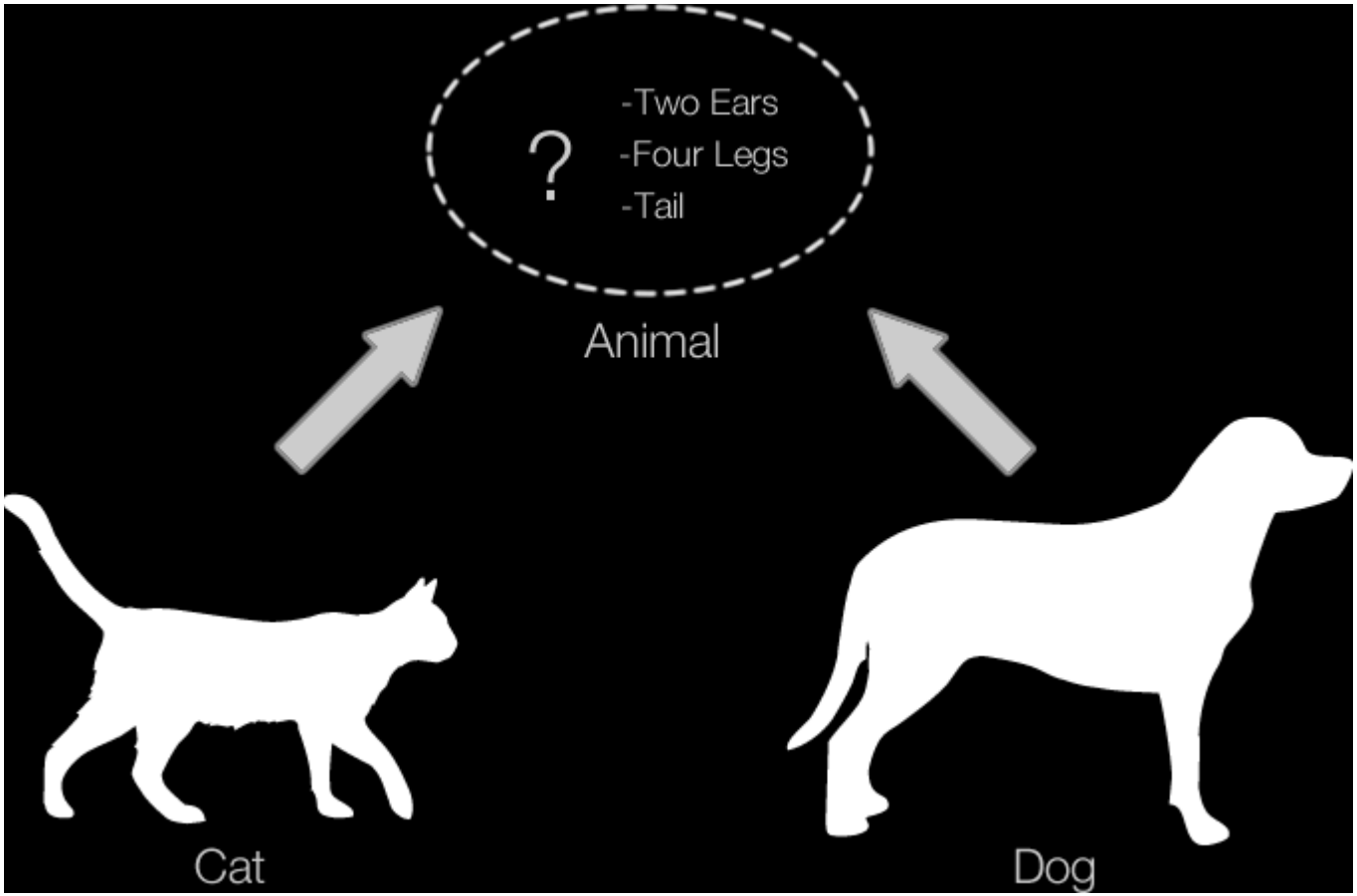
Hero 類別封裝

Attributes

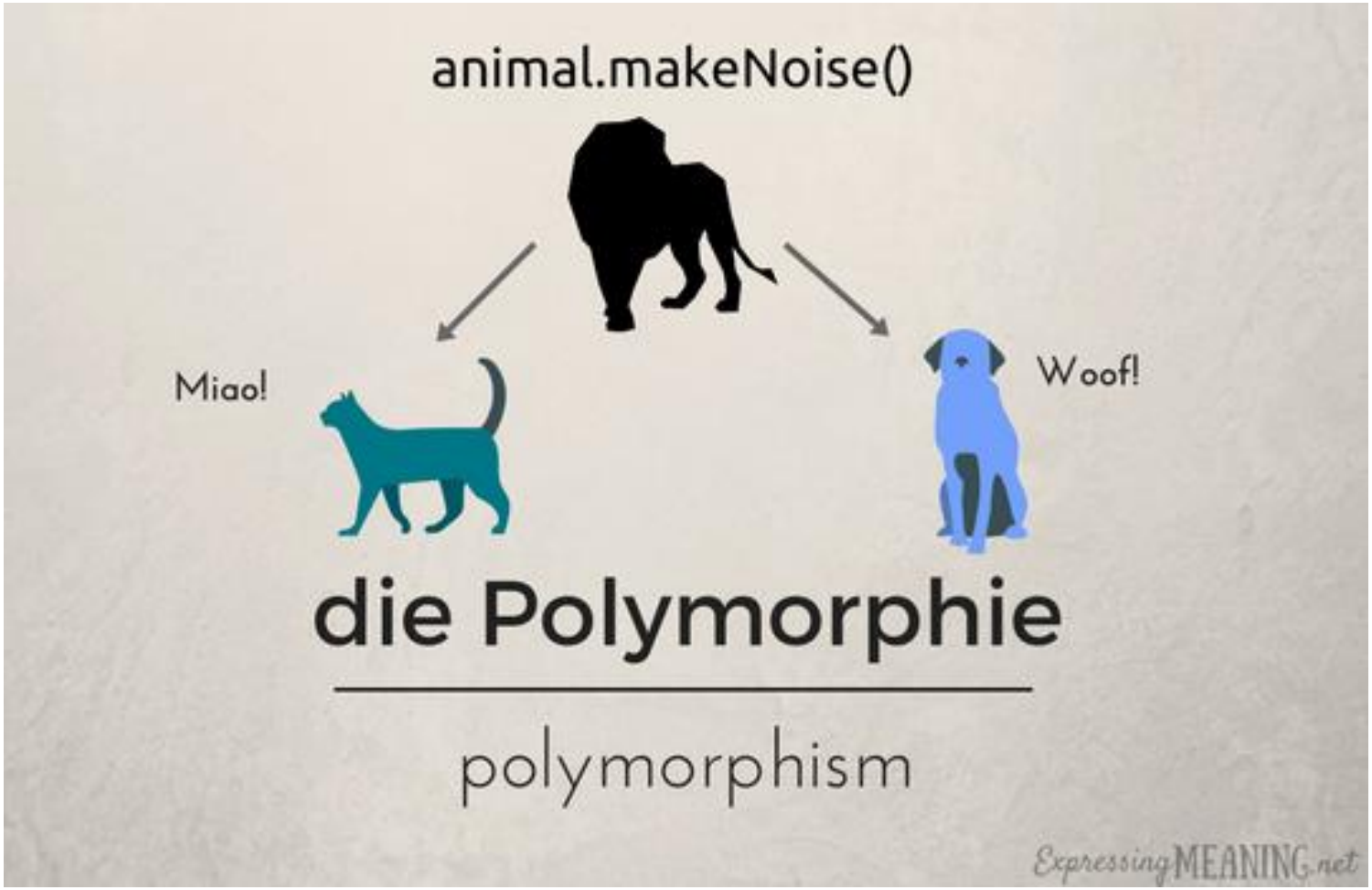
Methods

Class Hero
- name: String - hp: int - str: int - exp: int
+ Hero(name: String) + getName(): String + setName(): String + getHP(): int + getStr(): int + getExp: int + attack(target: Hero): void + beHit(enemyStr: int): void

Inheritance (繼承)



Polymorphism (多形)



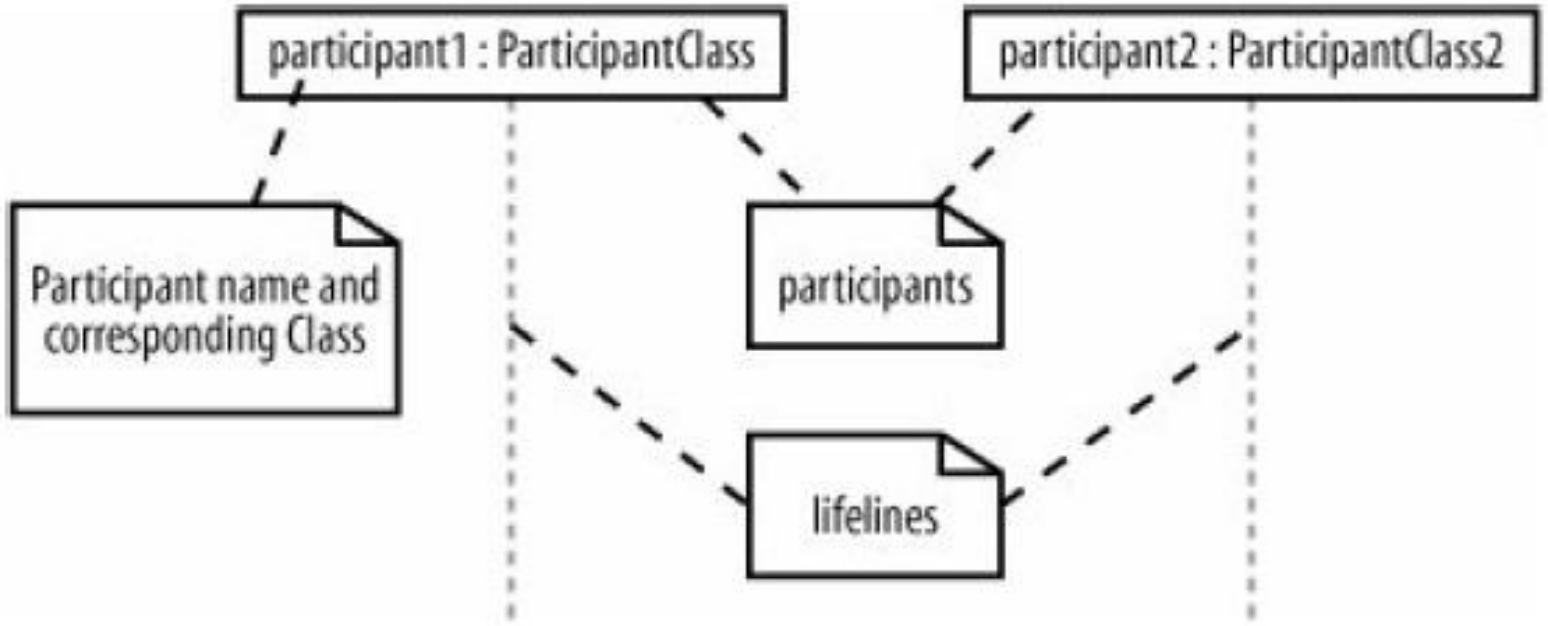
系統塑模

序列圖 (SEQUENCE DIAGRAM)

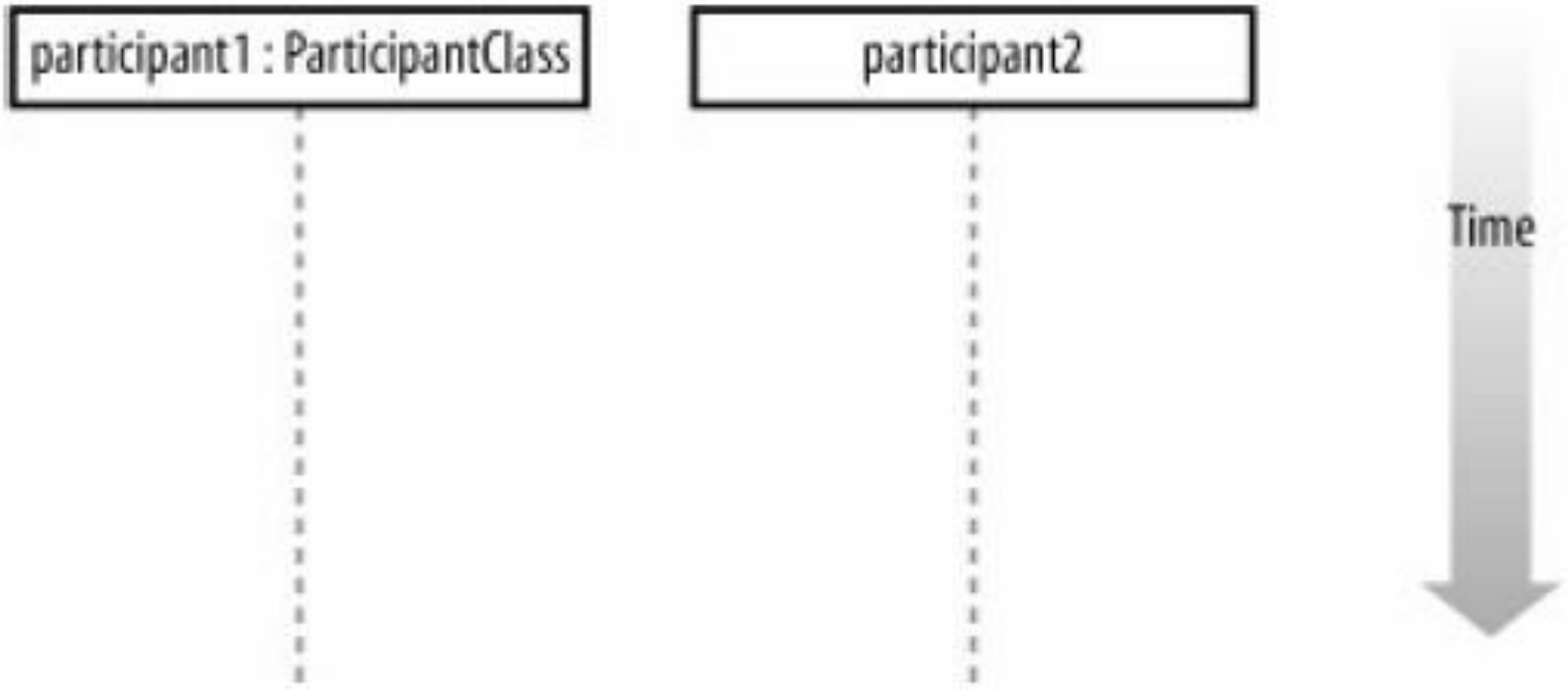
序列圖 (Sequence Diagram)

- 將結構上的描述與行為上的描述相結合
- 執行期 (Runtime) 的交談模型 (Interaction Model)

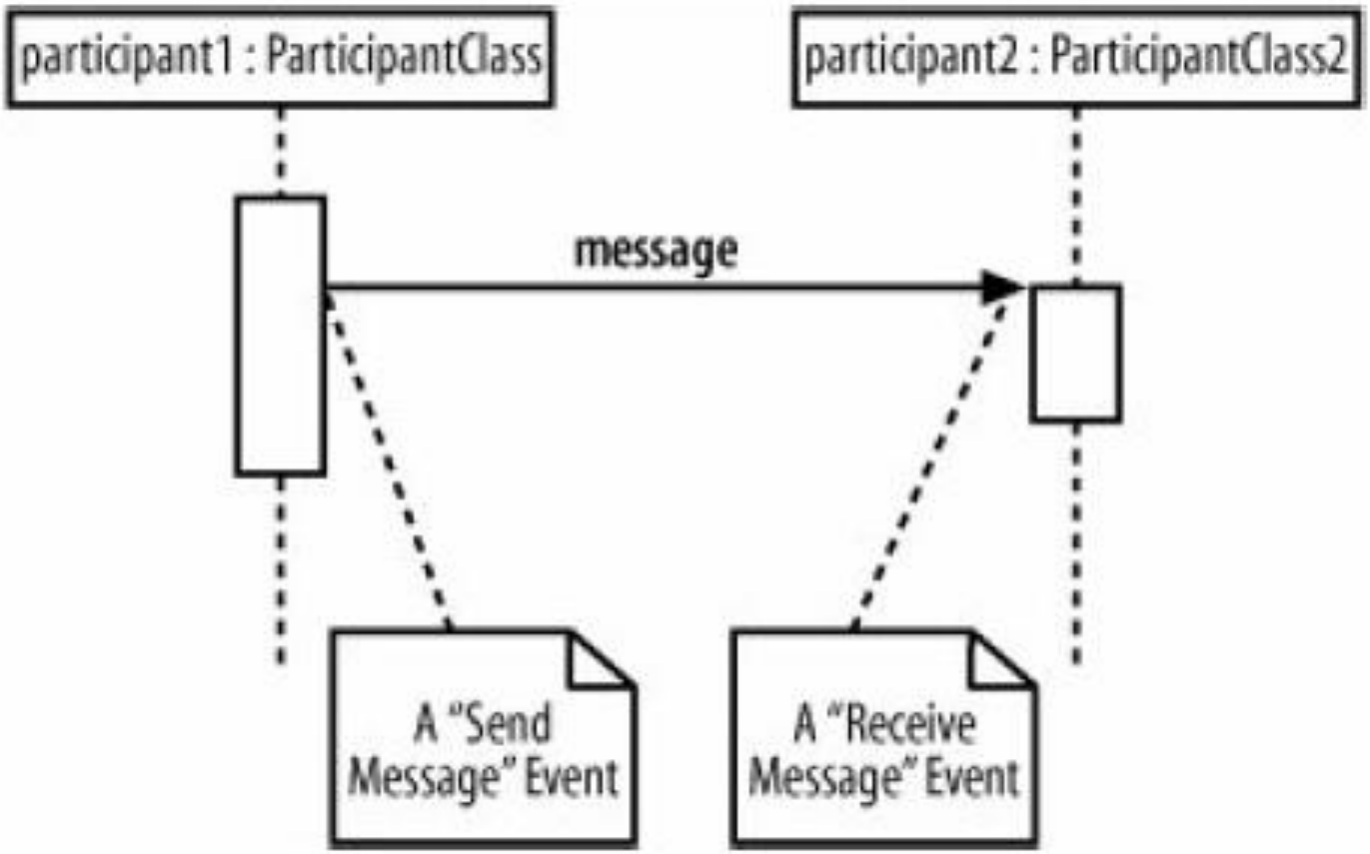
序列圖 (Sequence Diagram)



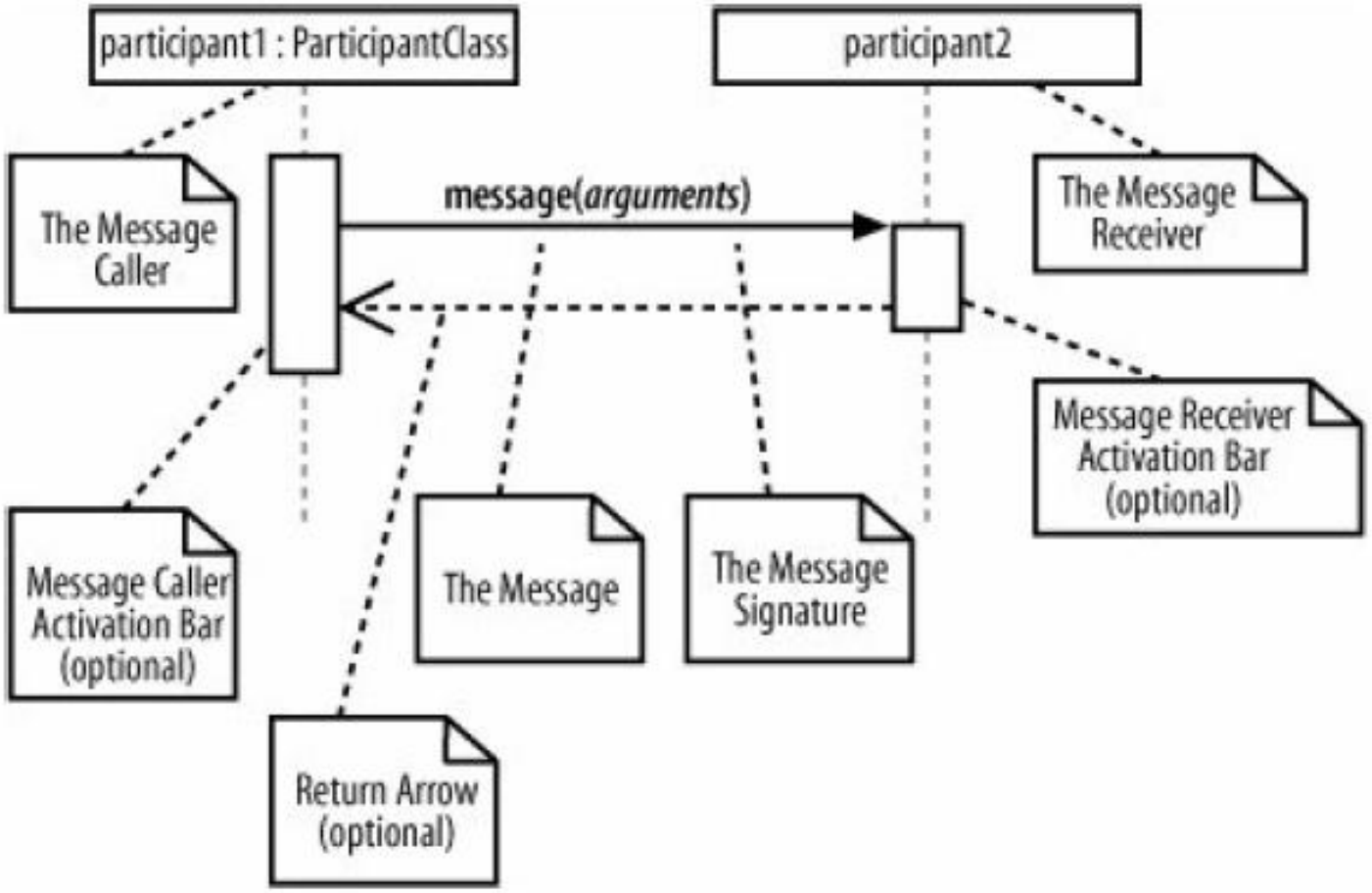
序列圖 (cont.)



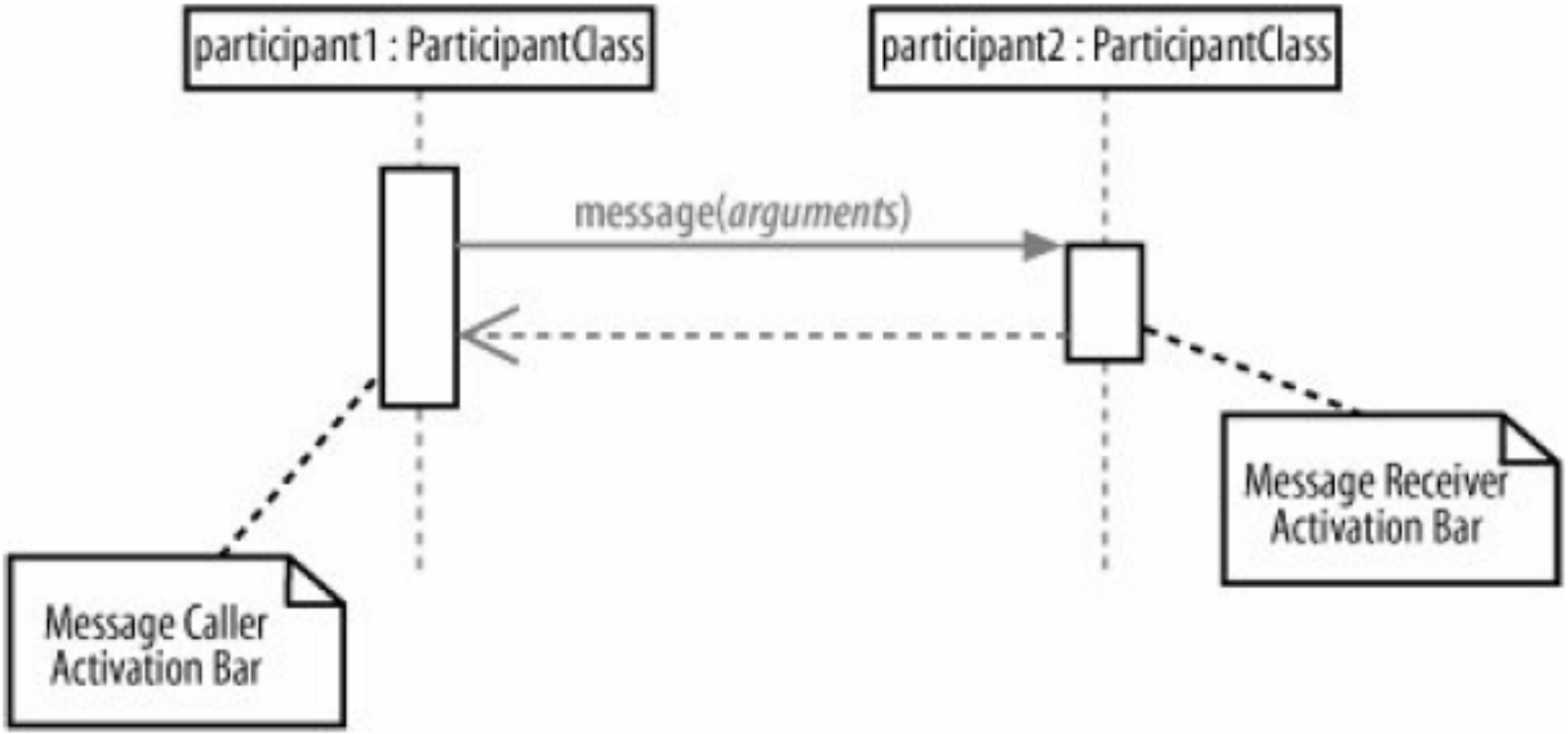
序列圖 (cont.)


participant1 : ParticipantClass
participant2 : ParticipantClass2
message
A "Send Message" Event
A "Receive Message" Event

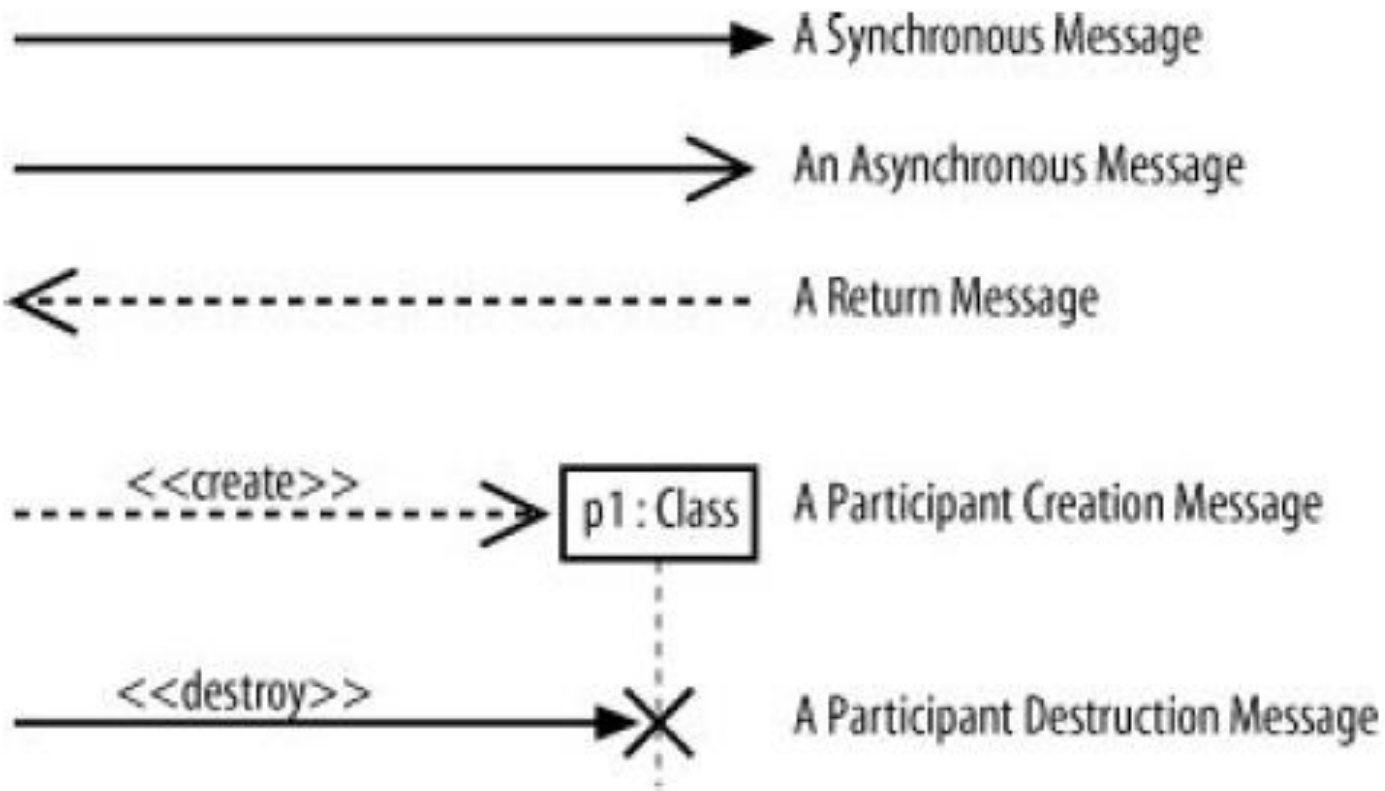
序列圖 (cont.)


participant1 : ParticipantClass
participant2
The Message Caller
message(arguments)
The Message Receiver
Message Receiver Activation Bar (optional)
Message Caller Activation Bar (optional)
The Message
The Message Signature
Return Arrow (optional)

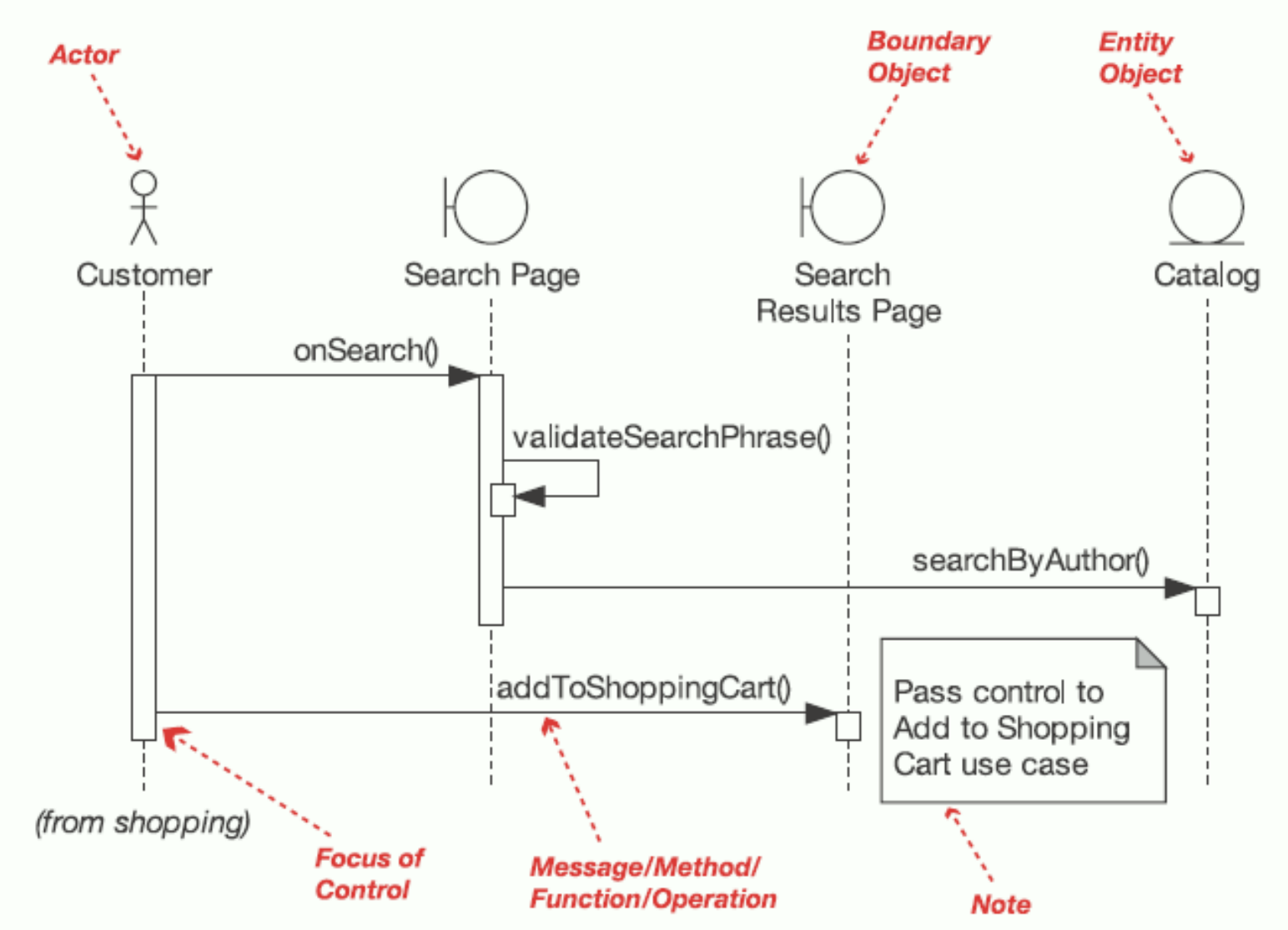
序列圖 (cont.)


participant1 : ParticipantClass
participant2 : ParticipantClass
message(arguments)
Message Caller
Activation Bar
Message Receiver
Activation Bar

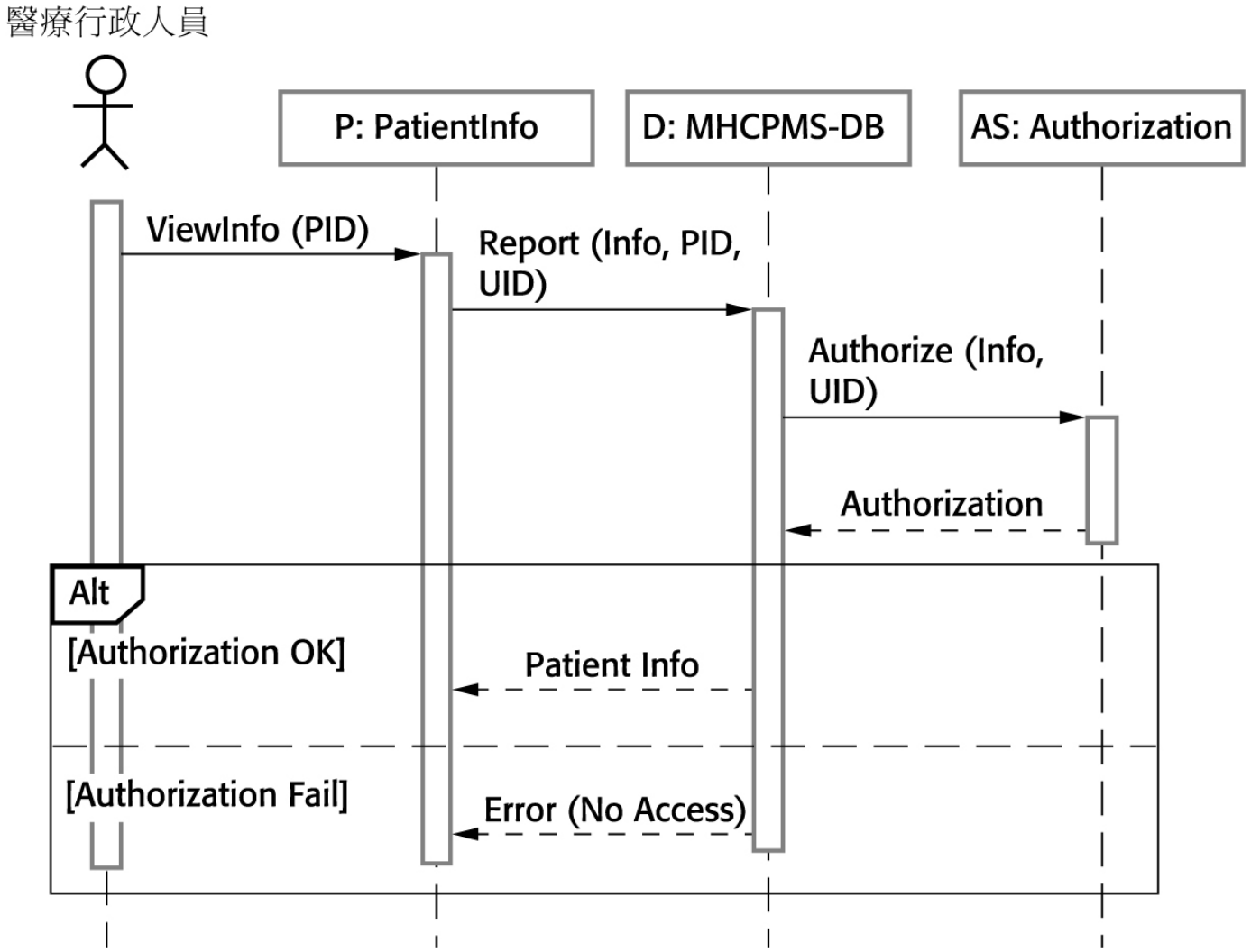
序列圖 (cont.)

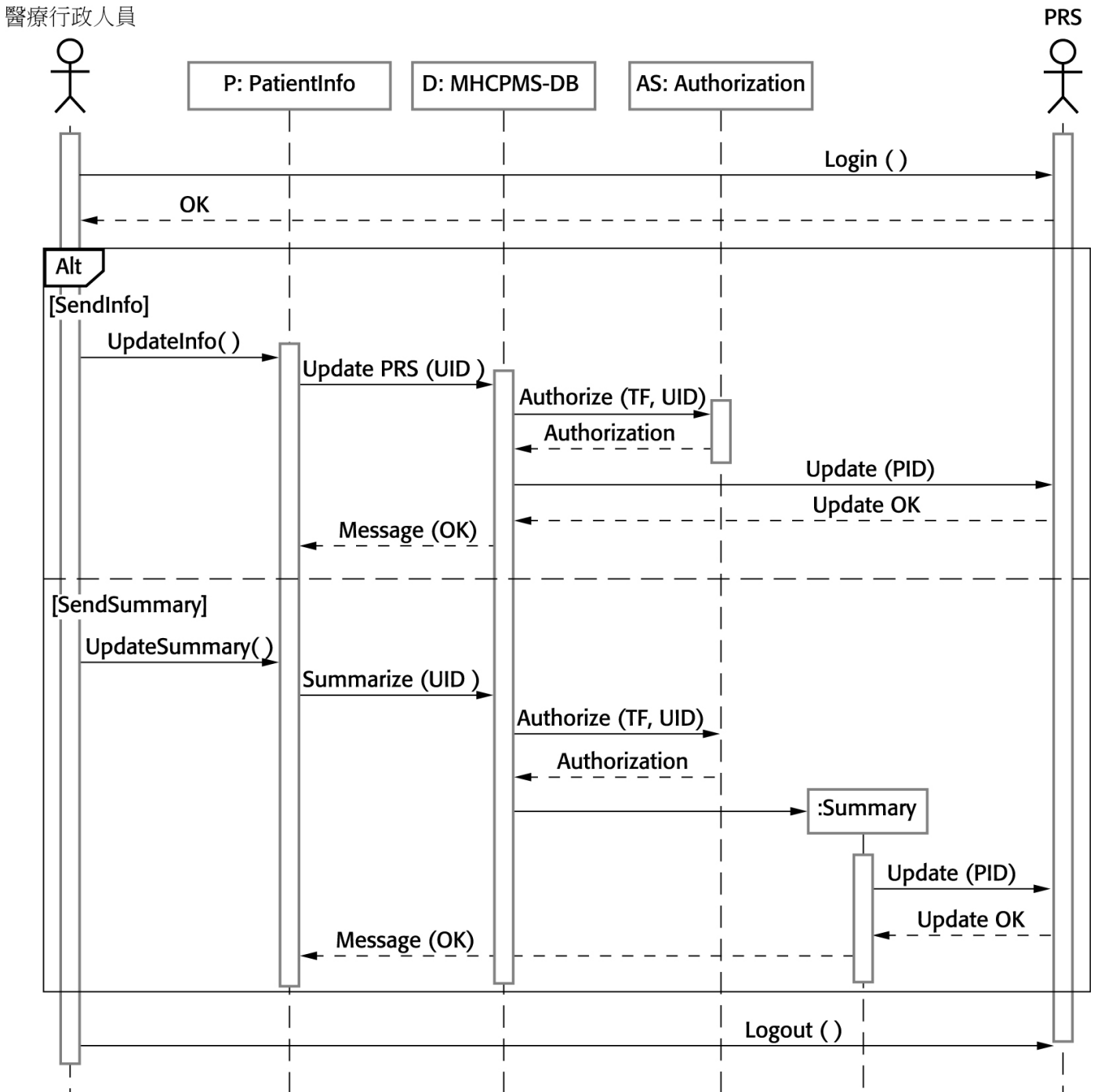

A Synchronous Message
An Asynchronous Message
A Return Message
<<create>>
p1 : Class
A Participant Creation Message
<<destroy>>
A Participant Destruction Message

序列圖範例 (cont.)


Actor
Boundary Object
Entity Object
Customer
Search Page
Search Results Page
Catalog
onSearch()
validateSearchPhrase()
searchByAuthor()
addToShoppingCart()
Pass control to Add to Shopping Cart use case
(from shopping)
Focus of Control
Message/Method/ Function/Operation
Note

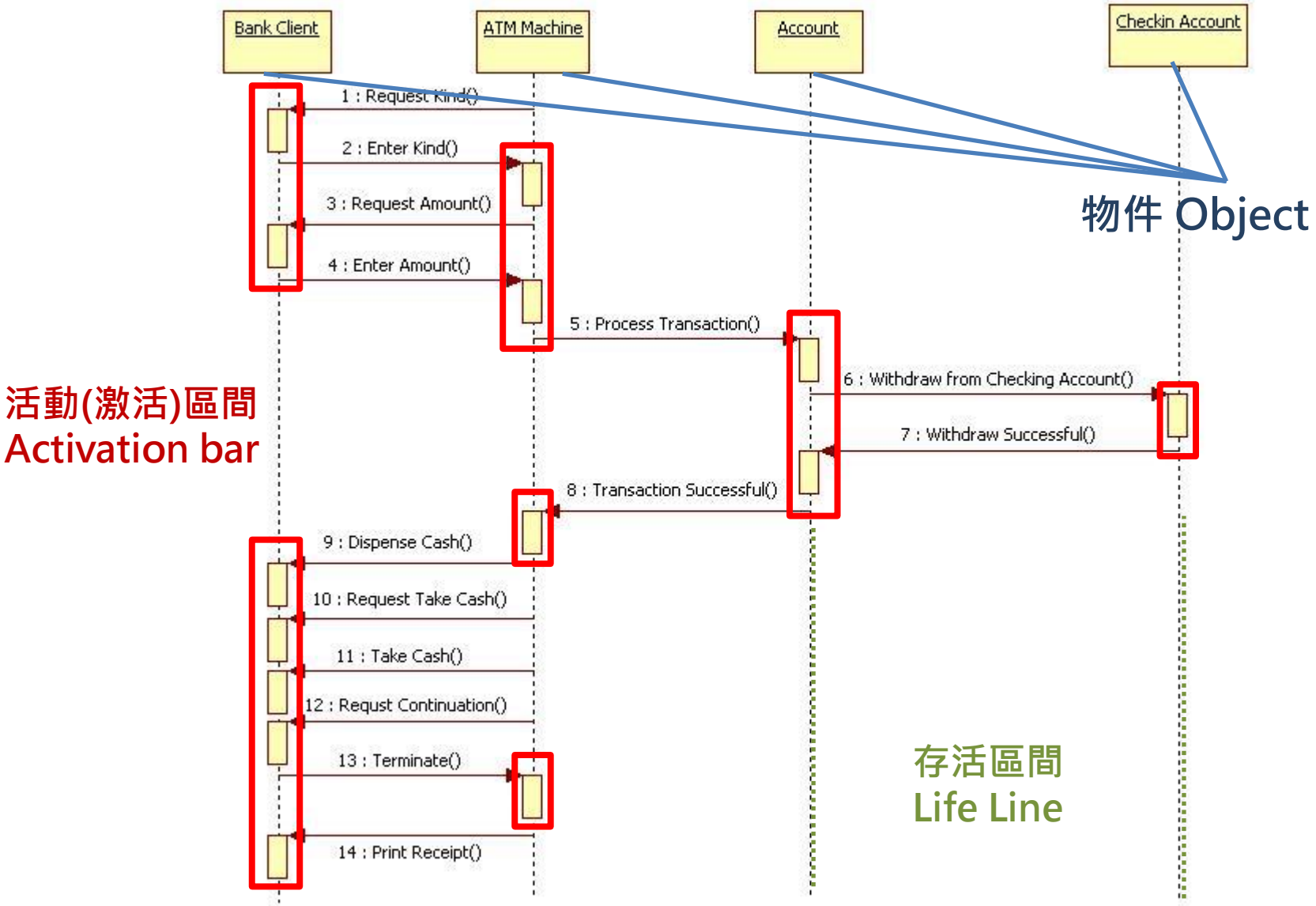
序列圖範例 (cont.)


醫療行政人員
P: PatientInfo
D: MHCPMS-DB
AS: Authorization
ViewInfo (PID)
Report (Info, PID, UID)
Authorize (Info, UID)
Authorization
Alt
[Authorization OK]
Patient Info
[Authorization Fail]
Error (No Access)


醫療行政人員
PRS
P: PatientInfo
D: MHCPMS-DB
AS: Authorization
Login ( )
OK
Alt
[SendInfo]
UpdateInfo( )
Update PRS (UID)
Authorize (TF, UID)
Authorization
Update (PID)
Update OK
Message (OK)
[SendSummary]
UpdateSummary( )
Summarize (UID)
Authorize (TF, UID)
Authorization
:Summary
Update (PID)
Update OK
Message (OK)
Logout ( )

試著畫出 ATM 提款流程的序列圖

序列圖 (Sequence Diagram)



Exercise

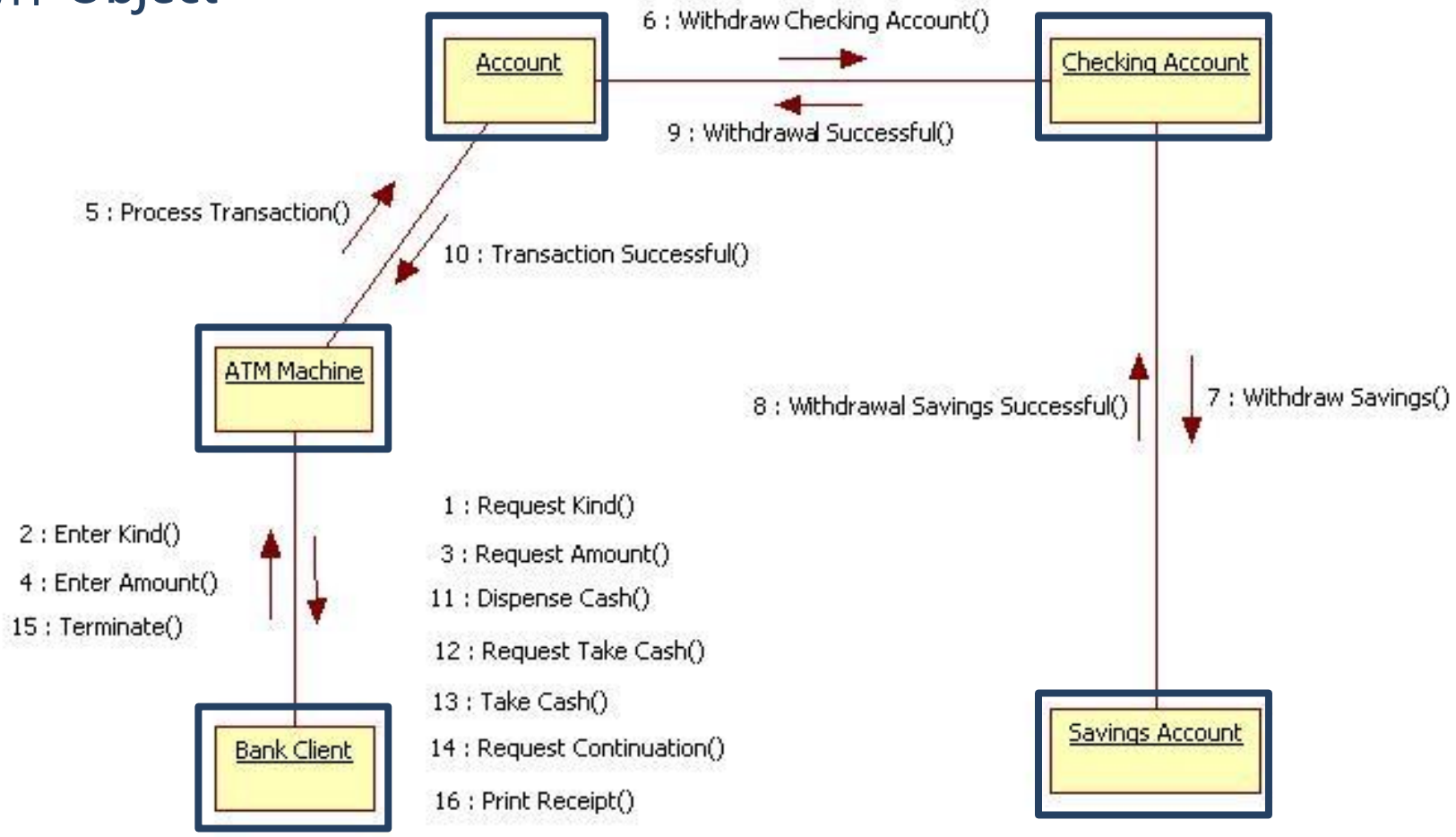
- 挑選選課系統其中一項 Use Case、根據其 Use Case Specification 以及 Robustness Diagram 中相關的部分，繪製序列圖

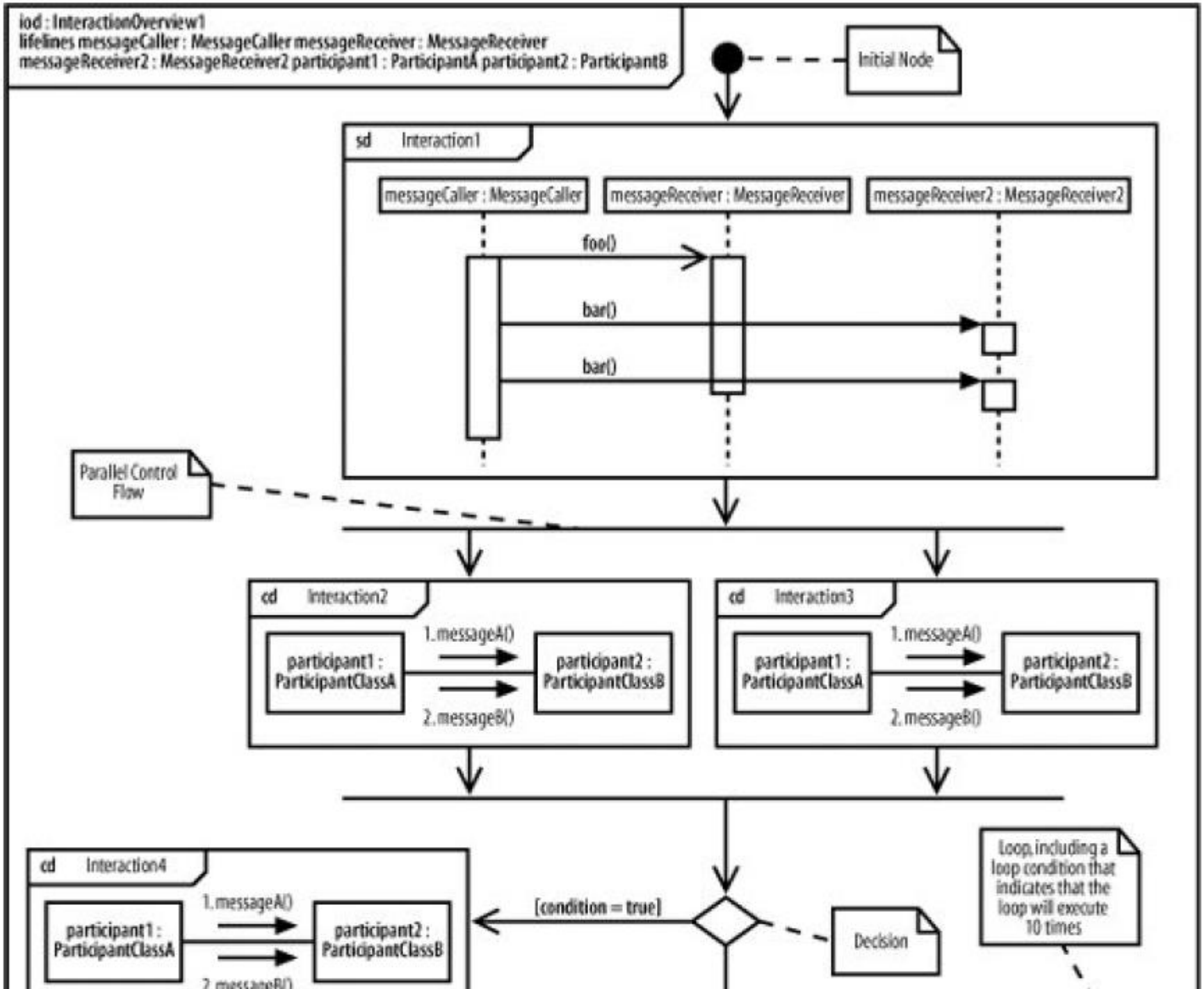
系統塑模

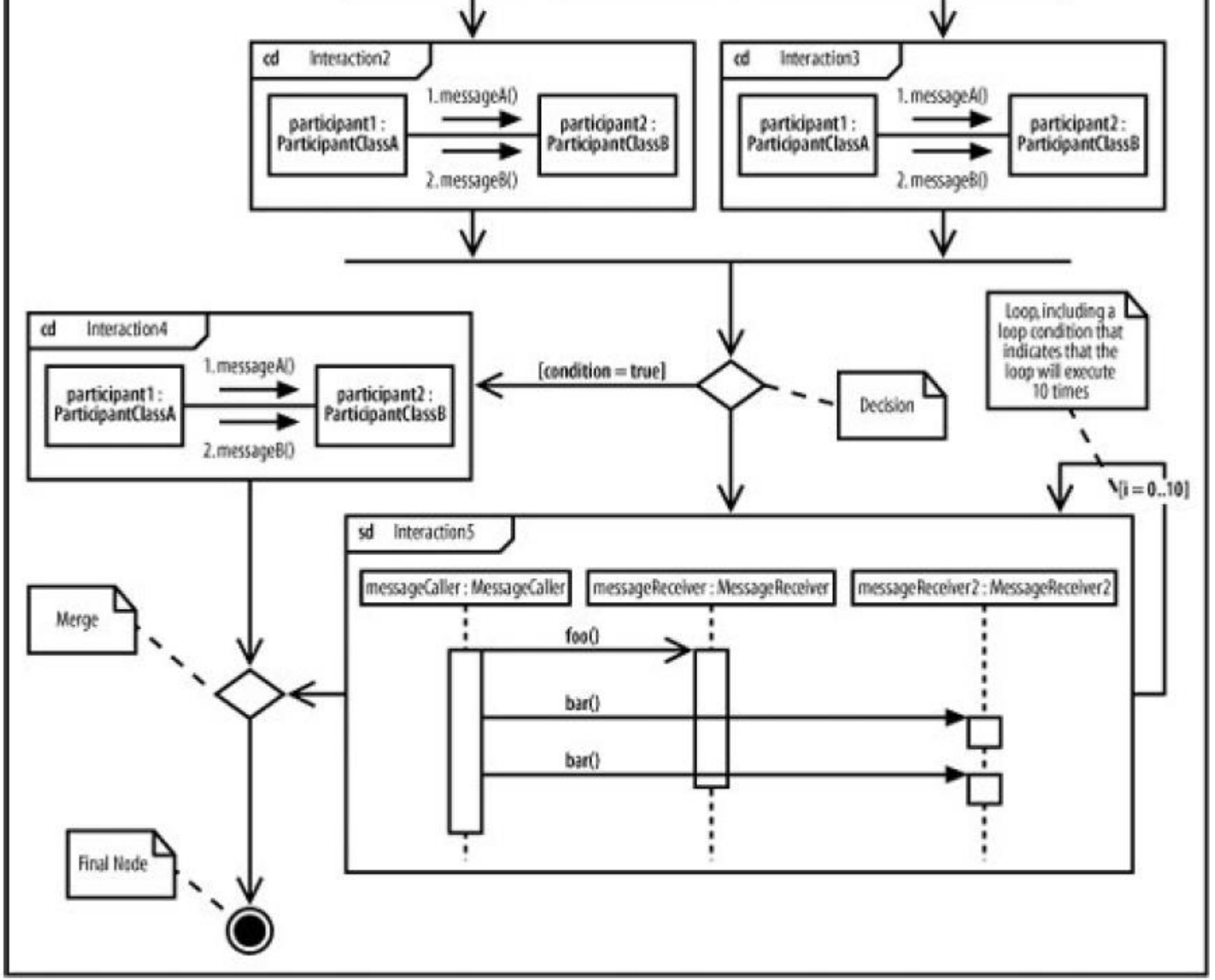
通信圖 (COMMUNICATION DIAGRAM)

通信圖 (Communication Diagram)

物件 Object




iod : InteractionOverview1
lifelines messageCaller : MessageCaller messageReceiver : MessageReceiver
messageReceiver2 : MessageReceiver2 participant1 : ParticipantA participant2 : ParticipantB
Initial Node
sd Interaction1
messageCaller : MessageCaller
messageReceiver : MessageReceiver
messageReceiver2 : MessageReceiver2
foo()
bar()
bar()
Parallel Control Flow
cd Interaction2
participant1 : ParticipantClassA
1. messageA()
2. messageB()
participant2 : ParticipantClassB
cd Interaction3
participant1 : ParticipantClassA
1. messageA()
2. messageB()
participant2 : ParticipantClassB
cd Interaction4
participant1 : ParticipantClassA
1. messageA()
2. messageB()
participant2 : ParticipantClassB
[condition = true]
Decision
Loop, including a loop condition that indicates that the loop will execute 10 times


cd Interaction2
participant1 : ParticipantClassA
1.messageA()
2.messageB()
participant2 : ParticipantClassB
cd Interaction3
participant1 : ParticipantClassA
1.messageA()
2.messageB()
participant2 : ParticipantClassB
cd Interaction4
participant1 : ParticipantClassA
1.messageA()
2.messageB()
participant2 : ParticipantClassB
[condition = true]
Decision
Loop, including a loop condition that indicates that the loop will execute 10 times
[i = 0..10]
sd Interaction5
messageCaller : MessageCaller
messageReceiver : MessageReceiver
messageReceiver2 : MessageReceiver2
foo()
bar()
bar()
Merge
Final Node

Exercise

- 將前一個 Exercise 的序列圖改畫為通信圖

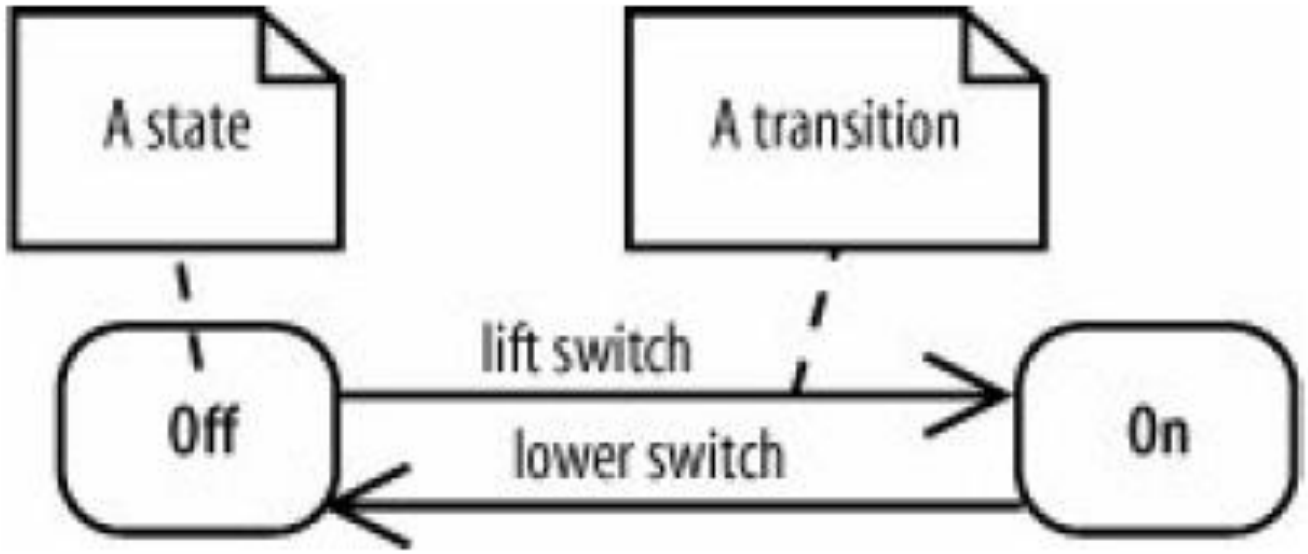
系統塑模

狀態圖
(STATE CHART DIAGRAM)

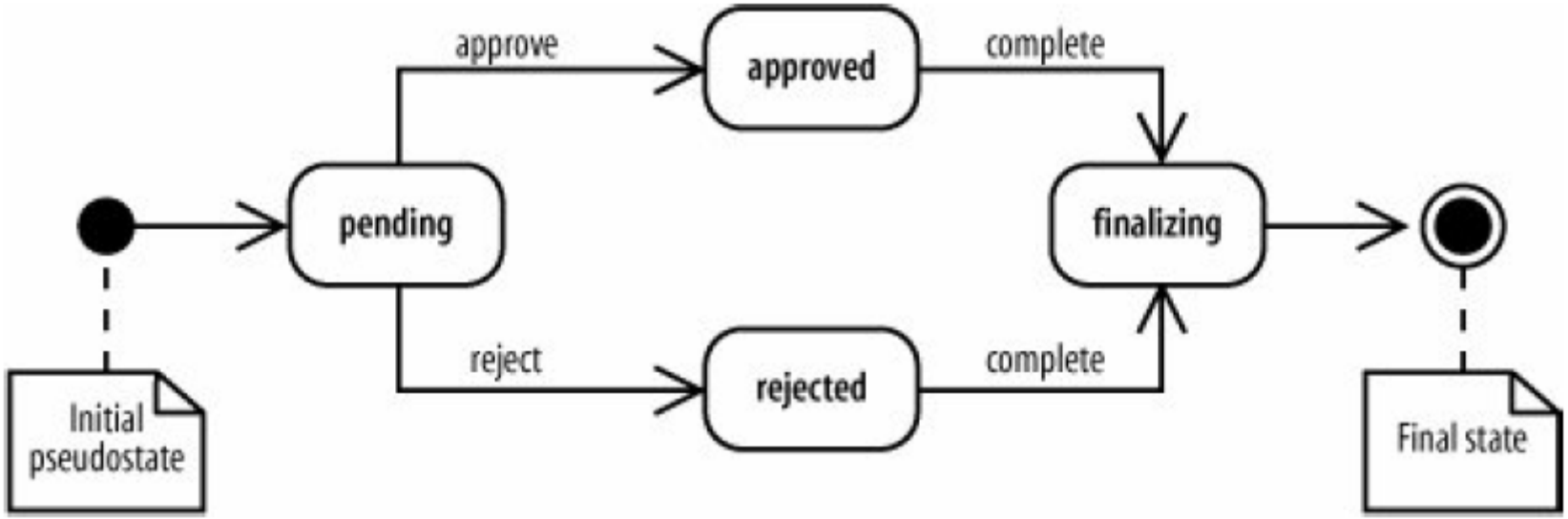
狀態圖 (State Chart Diagram)

- 表示物件的狀態

狀態圖 (cont.)



狀態圖 (cont.)


approve
approved
complete
pending
finalizing
reject
rejected
complete
Initial pseudostate
Final state

微波爐的狀態與刺激類型

圖5.17 微波爐的狀態與刺激類型

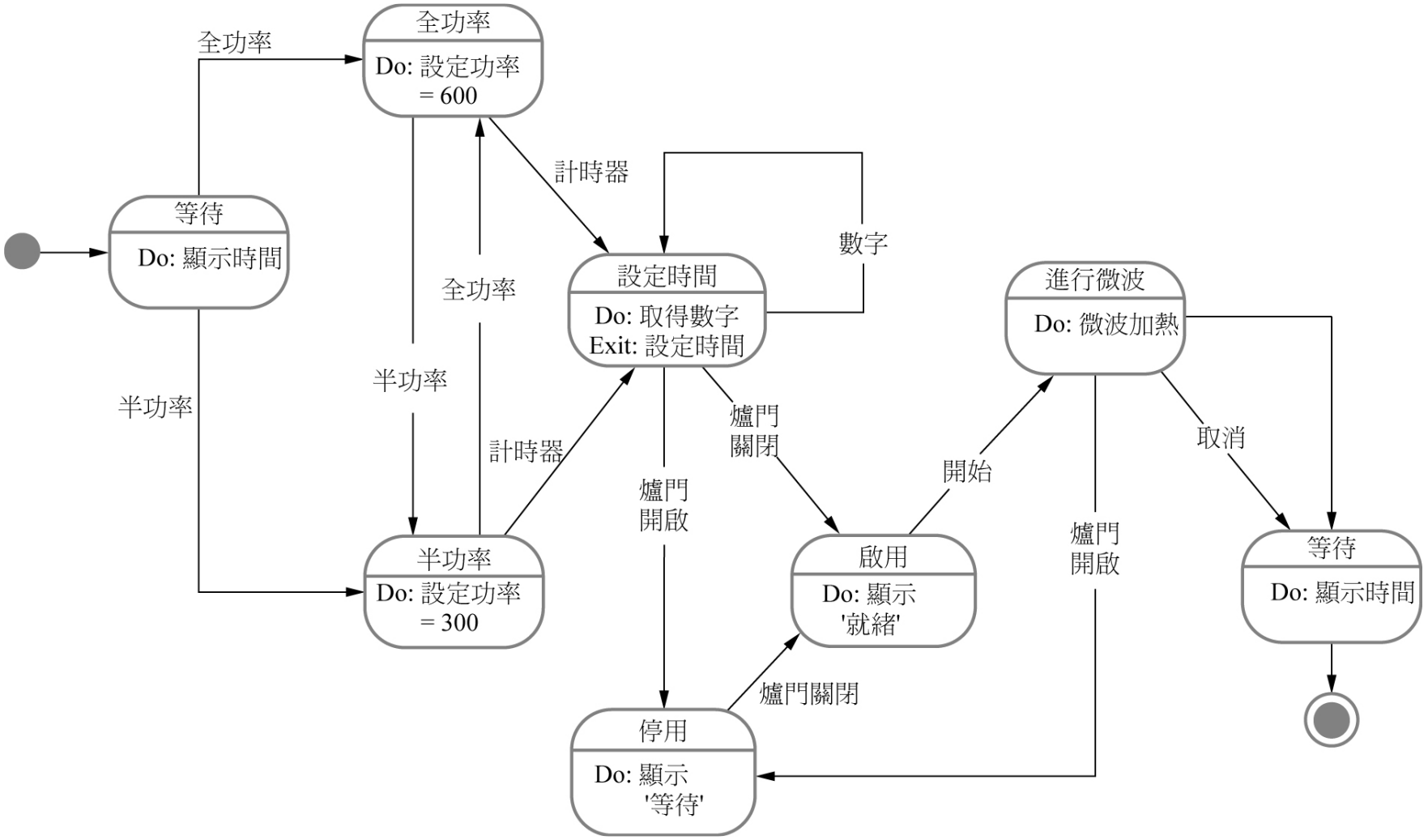
狀態	說明
等待	微波爐等待輸入，螢幕顯示目前時間
半功率	微波爐的功率設為300 瓦，螢幕顯示「半功率」
全功率	微波爐的功率設為600 瓦，螢幕顯示「全功率」
設定時間	烹煮時間設為使用者輸入值。螢幕顯示所選的烹煮時間，並且在時間設定後會更新
停用	基於安全理由微波爐被停用，微波爐內部的燈會亮，螢幕顯示「準備中」
啟用	微波爐被啟用，內部的燈關閉，螢幕顯示「準備烹煮」
進行微波	微波爐運作中，內部的燈會亮，螢幕顯示倒數時間。烹煮完成後，發出5 秒鐘嗶聲，微波爐燈號亮起，發出聲響時螢幕顯示「烹煮完成」訊息

微波爐的狀態與刺激類型 (續)

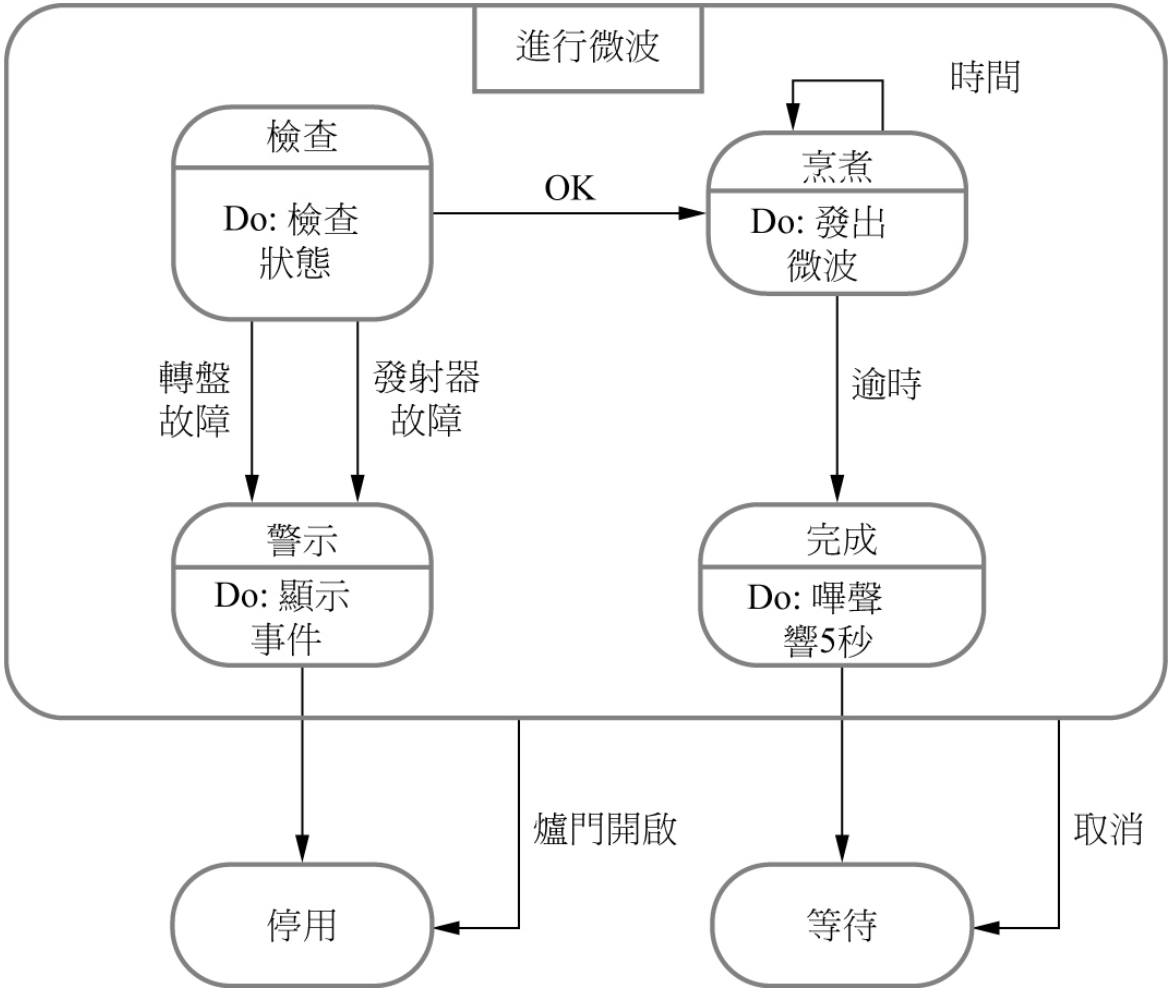
圖5.17 微波爐的狀態與刺激類型

刺激	說明
半功率	使用者按下半功率按鈕
全功率	使用者按下全功率按鈕
計時器	使用者按下計時器的某個按鈕
數字	使用者按下某個數字鍵
爐門開啟	微波爐的門開關未關上
爐門關閉	微波爐的門開關已關上
開始	使用者按下開始按鈕
取消	使用者按下取消按鈕

微波爐的狀態圖

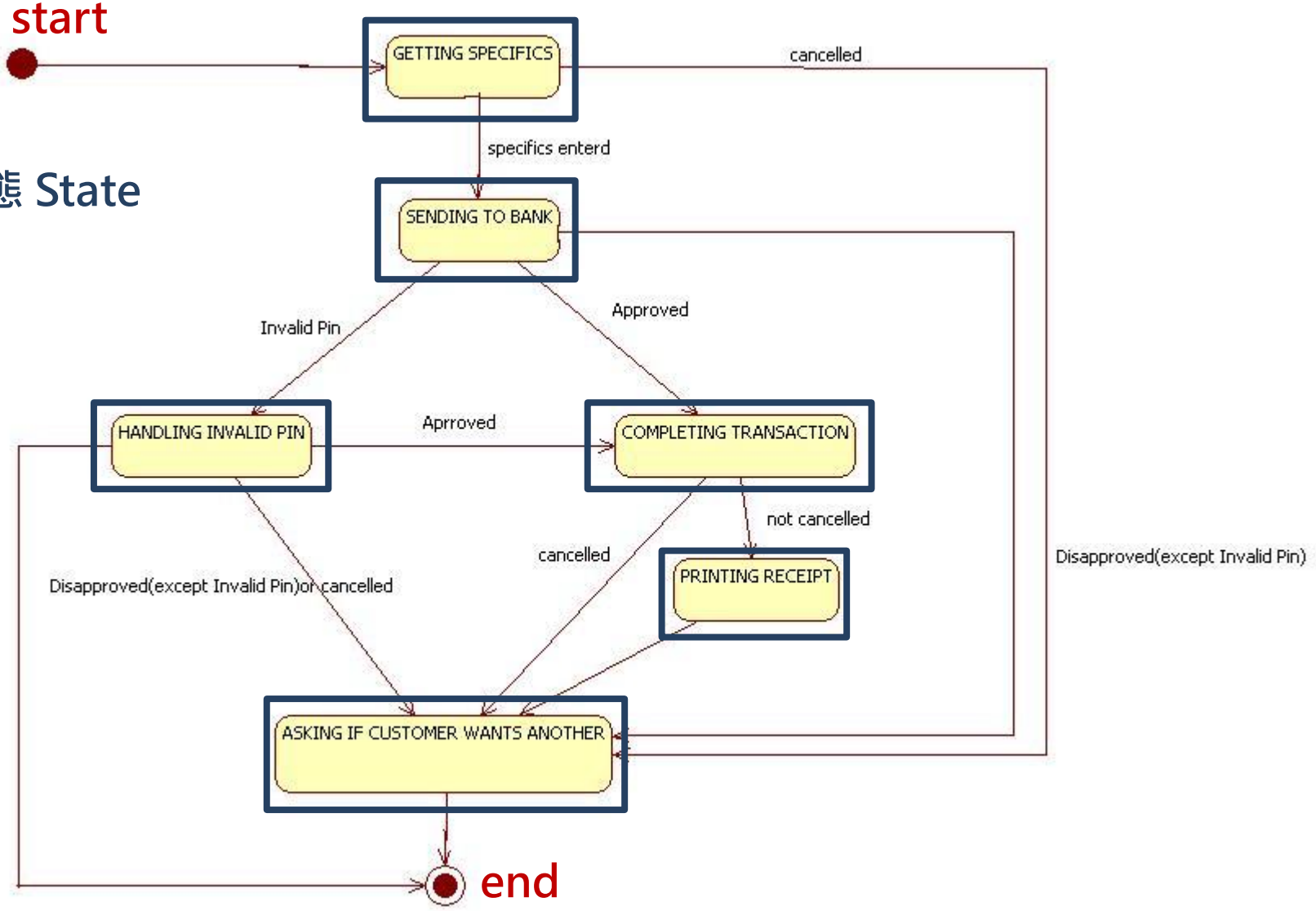

全功率
全功率
Do: 設定功率
= 600
計時器
等待
Do: 顯示時間
數字
設定時間
Do: 取得數字
Exit: 設定時間
進行微波
Do: 微波加熱
全功率
半功率
半功率
計時器
爐門
關閉
開始
取消
爐門
開啟
爐門
開啟
半功率
Do: 設定功率
= 300
啟用
Do: 顯示
'就緒'
等待
Do: 顯示時間
爐門關閉
停用
Do: 顯示
'等待'

微波爐加熱 活動圖 + 狀態圖



ATM 的狀態圖

狀態 State



系統塑模

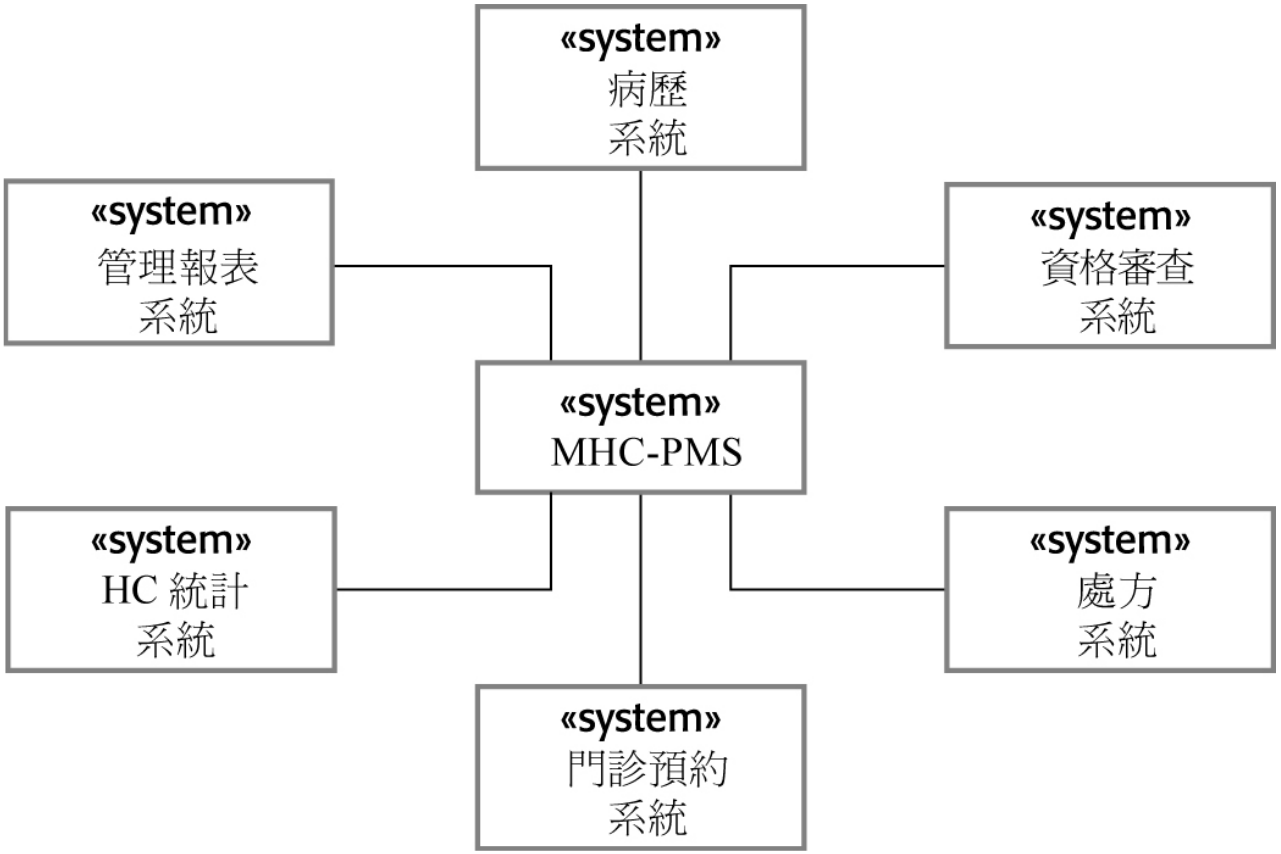
模型之應用與其他塑模用圖

環境模型 (Context Model)

- 描述系統以及系統的執行環境
- 描述系統的邊界
  - 用以限制瞭解系統需求和設計所需的成本和時間
  - 由於社會和組織的考量，使系統邊界的定位可能會決定於某些非技術性因素
- 無法顯示系統與環境中其他系統之間的關係
- 通常會需要其他模型來補充

MHC-PMS 系統的環境

MHC-PMS 系統的環境 – 元件圖



程序模型

- 描述系統執行的程序
  - Control flow
  - Data flow
- 活動圖

互動模型

- 使用案例塑模
  - 大多用來建立系統與外界行動者 (使用者或其他系統) 之間互動的模型。
- 序列圖、通信圖
  - 用來建立系統元件之間互動的模型，有時外界機構也會包括進來

結構模型 (Structural Model)

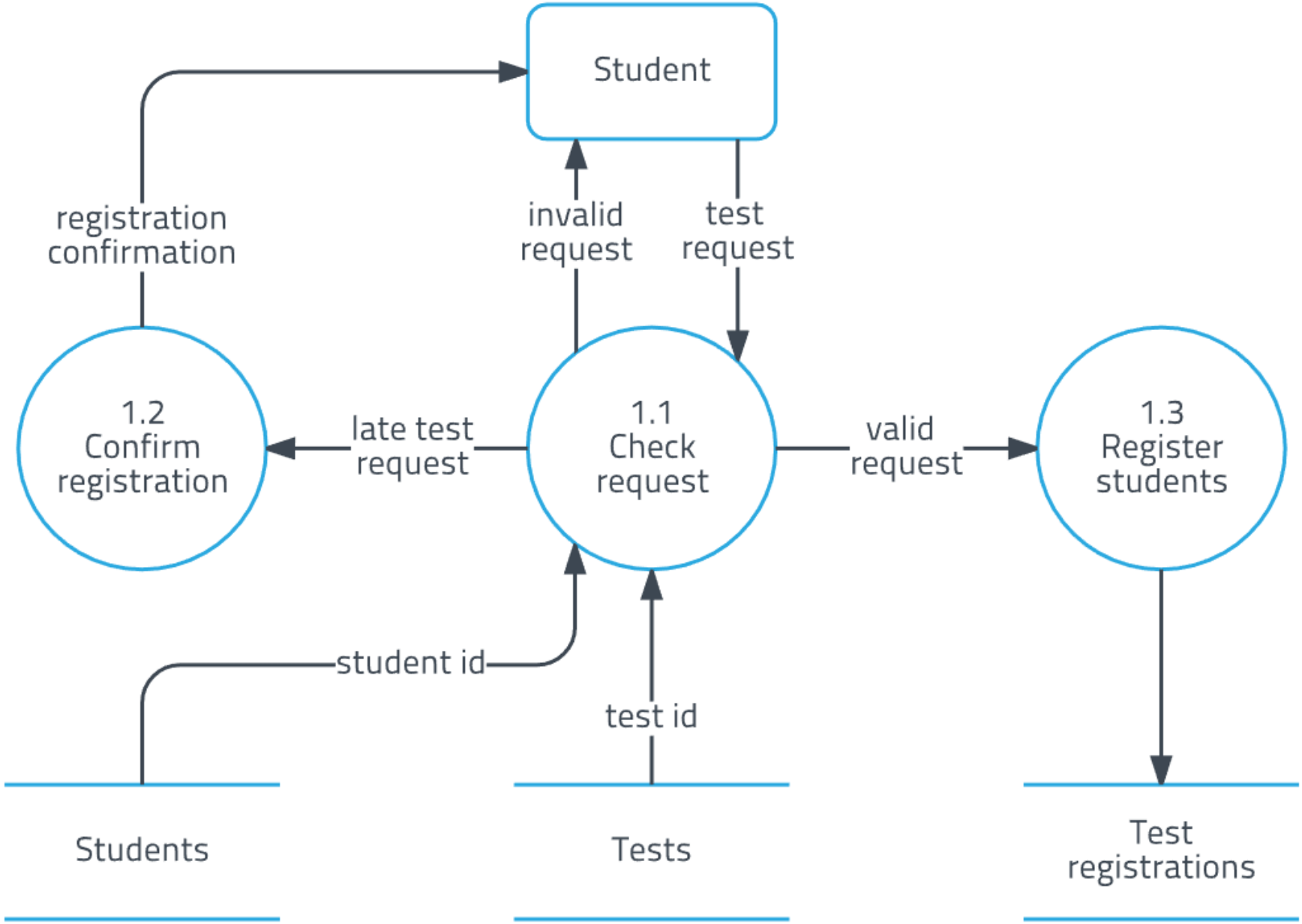
- 以組成系統的元件和它們之間的關係來顯示系統的架構
  - 可顯示系統設計結構的靜態模型
  - 也可顯示系統執行中架構的動態模型
- 在討論和設計系統架構時會建立系統的結構模型
- 穩健圖、類別圖

行為模型 (Behavioral Model)

- 描述系統執行中動態行為的模型
- 展示系統對來自環境刺激的回應
- 刺激的類型
  - 資料：有些必須由系統處理的資料剛抵達
  - 事件：發生某些會觸發系統處理的事件

資料驅動塑模 (Data-Driven Modeling)

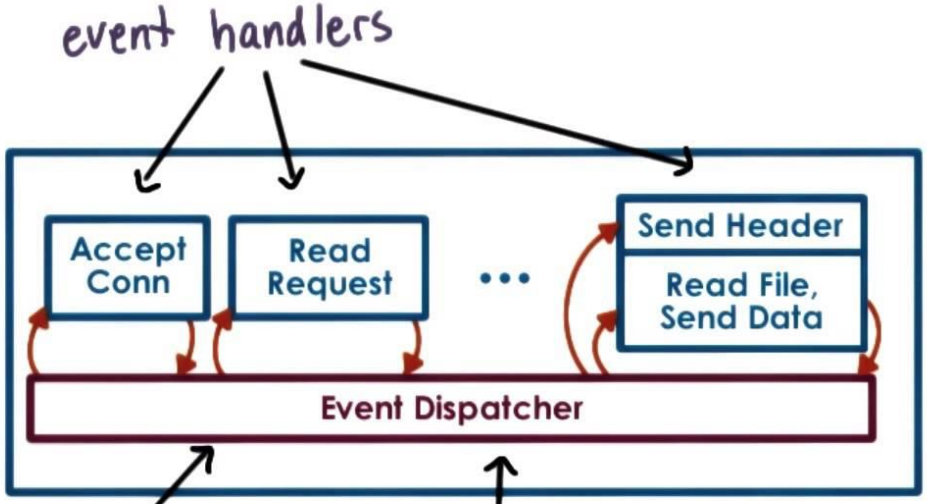
- 顯示參與處理輸入資料和產生相關輸出的一連串行動
- 資料驅動系統在商業上很常見
- 在需求分析期間特別有用，可顯示出系統從這端到那端之間的處理過程


Student
registration confirmation
invalid request
test request
1.2 Confirm registration
late test request
1.1 Check request
valid request
1.3 Register students
student id
test id
Students
Tests
Test registrations

事件驅動塑模
(Event-Driven Modeling)

- 描述系統如何回應外在和內部事件
- 特別合適即時系統
- 根據的假設是此系統的狀態個數是有限的

Event-Driven Model



single address space

single process!

single thread of control!

模型驅動工程 (Model-Driven Engineering)

- 開發程序的主要輸出是模型而非程式
- 從模型自動產生出能在硬體 / 軟體平台上執行的程式

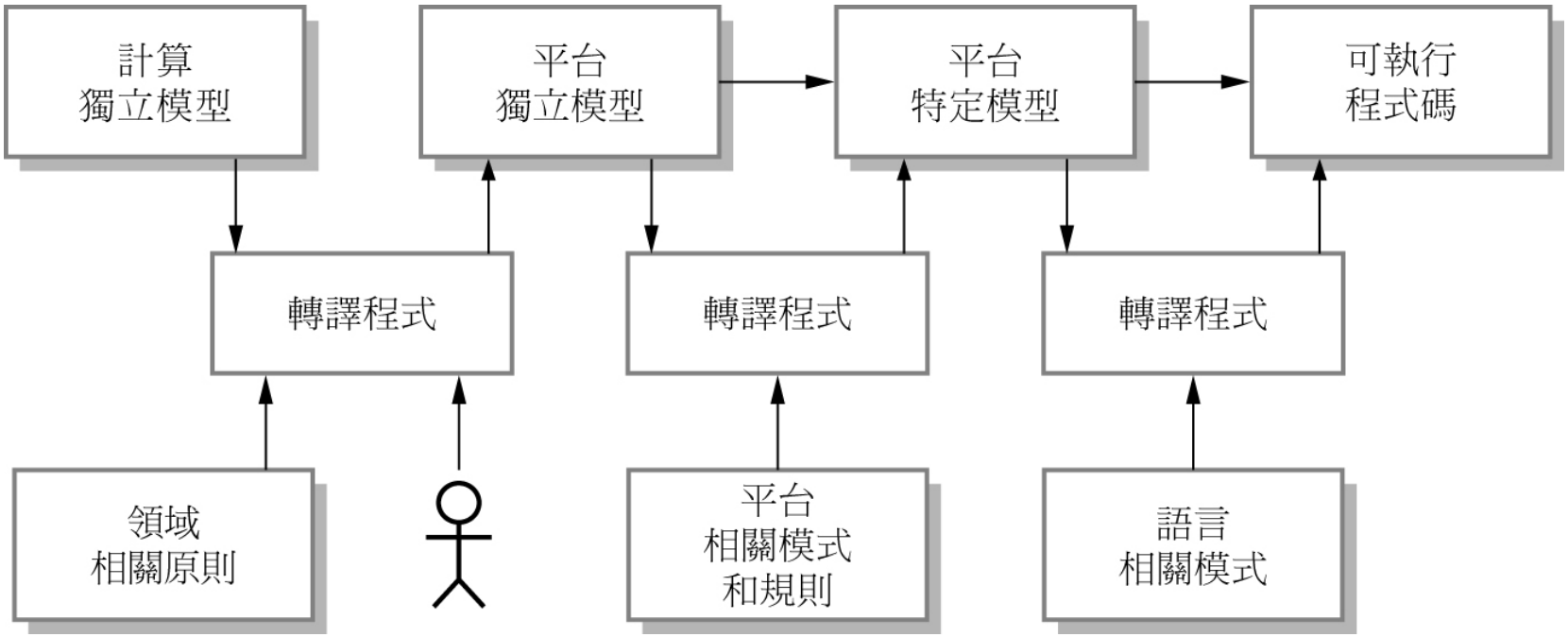
模型驅動架構 (Model-Driven Architecture)

- 針對模型的軟體設計和實作方法
- 使用UML 模型的子集來描述系統
- 從高階的與平台無關的模型，有可能在無需人為介入下產生一個可運作的程式

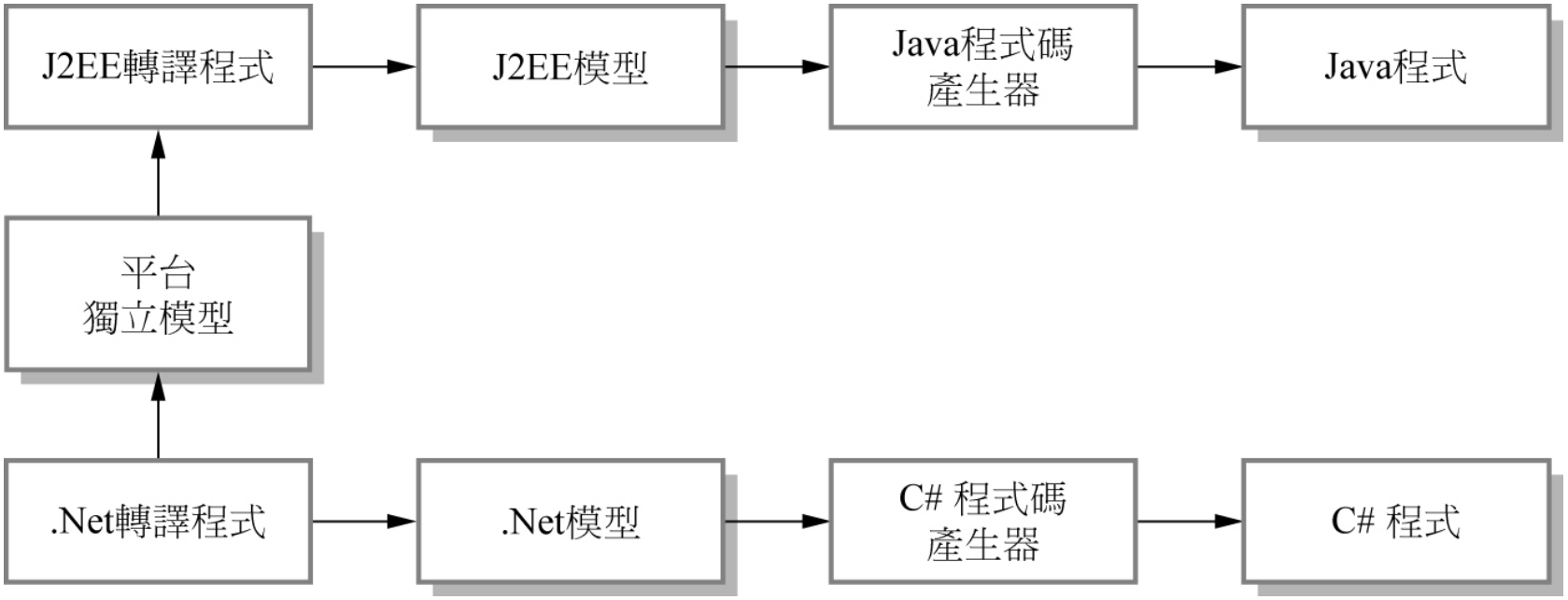
模型的類型

- 計算獨立模型
  - 描述系統中重要的領域抽象資訊的模型
- 平台獨立模型
  - 描述系統的運作但不會參考到它的實作
- 平台特定模型
  - 針對每一種應用程式平台，將平台獨立模型轉換成不同的平台特定模型

MDA 轉換動作


計算
獨立模型
平台
獨立模型
平台
特定模型
可執行
程式碼
轉譯程式
轉譯程式
轉譯程式
領域
相關原則
平台
相關模式
和規則
語言
相關模式

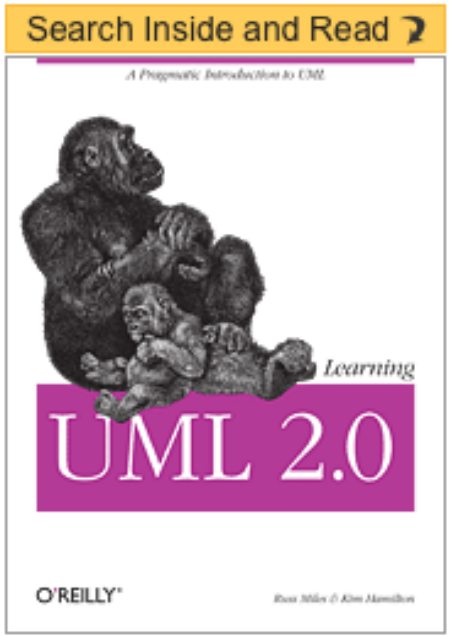
多重平台的模型


J2EE轉譯程式
J2EE模型
Java程式碼
產生器
Java程式
平台
獨立模型
.Net轉譯程式
.Net模型
C# 程式碼
產生器
C# 程式

Executable UML

- 為了建立一個可執行的UML 子集，模型種類的個數必須減少
- 關鍵的模型
  - 領域模型：描述系統最重要關心的東西
  - 類別模型：在此定義類別，還有它們的屬性和運算動作
  - 狀態模型：每個類別相關聯的狀態圖，用來描述類別的生命週期

參考教材



Larger Cover

Learning UML 2.0

A Pragmatic Introduction to UML

By Russ Miles, Kim Hamilton

Publisher: O'Reilly Media
Final Release Date: April 2006
Pages: 290

3.3

Read 3 Reviews | Write a Review

"Since its original introduction in 1997, the Unified Modeling Language has revolutionized software development. Every integrated software development environment in the world--open-source, standards-based, and proprietary--now supports UML and, more importantly, the model-driven approach to software development. This makes...

Full description