軟體工程簡介

吳庭育
tyw@mail.npust.edu.tw
國立屏東科技大學 資管系

軟體本質的問題

書籍： No Silver Bullet (1986)

作者： FredBrooks Jr.

- 複雜性(Complexity)
- 一致性(Conformity)
- 隱藏性(Invisibility)
- 易變性(Changeability)

複雜性(Complexity)

- 軟體系統的複雜程度，往往隨著程式的大小及軟體元件個數以非線性的方式，甚至是等比級數的方式遞增。
  - 系統的狀態、程式碼的大小、系統功能、程式靜態結構、系統動態行為

一致性(Conformity)

- 在大型的協作環境下發展軟體系統，介面跟介面間、模組跟模組間、系統跟系統間的介接，都存有一致性問題需要解決，因此需要透過各種方法來轉換或是介接不一致的地方。

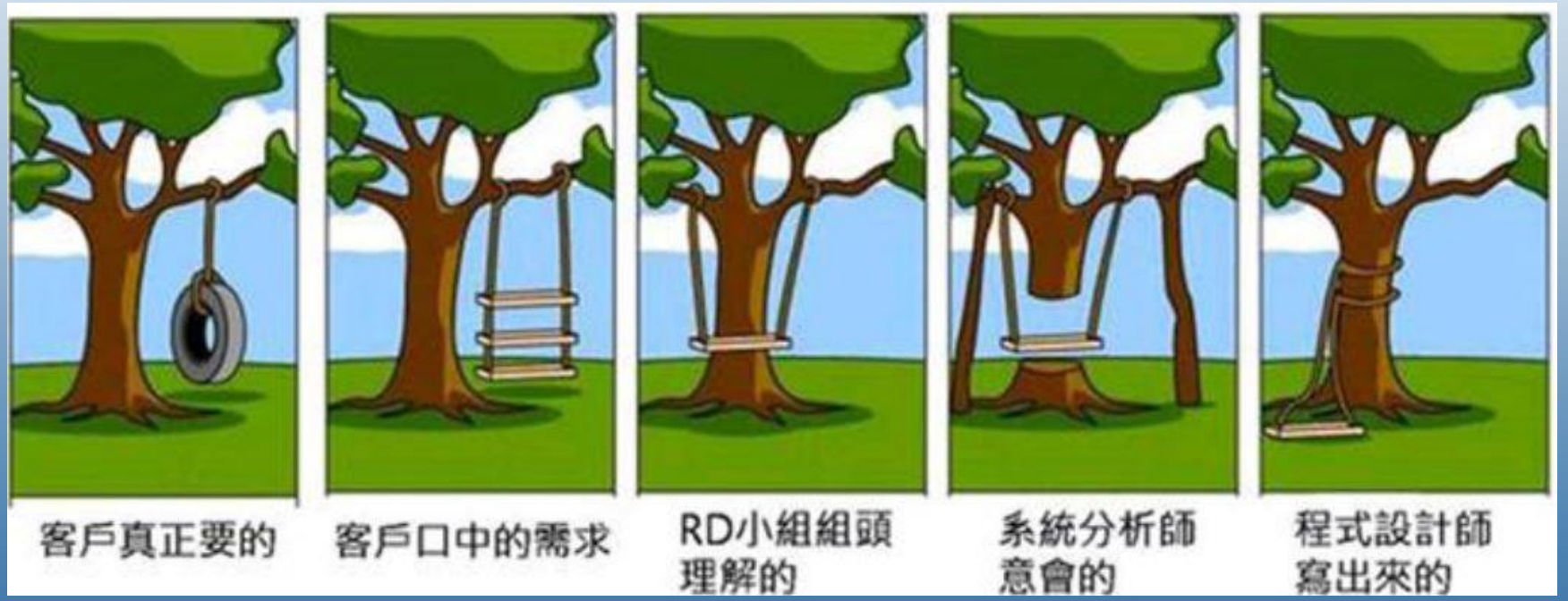
隱藏性(Invisibility)

- 軟體本身是看不到、摸不著的，導致需求容易存有誤解、疏忽的地方不容易被發現，而大大的妨礙彼此溝通的進行。

易變性(Changeability)

- 為了滿足客戶的需求，一套成功的軟體系統，從開發到完成、從產品交付到營運維護，隨時都可能要做變更。

這件事情容易嗎?



本章內容

- 軟體工程對政府、社會、整個國家，甚至跨國企業與機構的運作都是絕對必要的。現代世界沒有了軟體就無法運作。
- 我們生活週遭的娛樂業，包括音樂、電腦遊戲、電影和電視業也都大量使用軟體。

1. **系統複雜度越來越高**：由於新的軟體工程技術能幫助我們建構出更大、更複雜的系統，因此需求也改變了。現在系統的開發和交付速度必須更快，同時系統本身也更大、更複雜，而且還必須具備某些從前被認為不可能做到的新功能。因此需要開發新軟體工程技術，來應付這些新的挑戰。
2. **未能成功使用軟體工程方法**：不使用軟體工程方法和技術來撰寫電腦程式，其實是比較容易的。很多公司是隨著他們的產品和服務的發展而投入軟體開發，他們並沒有習慣在每天的工作中使用軟體工程方法，結果造成生產出來的軟體經常是成本更高，同時又比較不可靠。針對這個問題，我們需要更好的軟體工程教育和訓練。

1.1 專業軟體的開發

問題	解答
何謂軟體？	電腦程式和相關的說明文件。可以是為某特定客戶或一般大眾市場所開發的軟體產品
好的軟體有哪些特性？	軟體必須具備使用者要求的功能和效能，而且應該具有容易維護、可信任以及容易使用的特性
何謂軟體工程？	軟體工程是一門著重在軟體生產的各方面，從初步想法到運作和維護整個過程知識的工程學科
軟體工程的基本活動有哪些？	軟體規格制訂、軟體開發、軟體確認和軟體演進
軟體工程與電腦科學有何不同？	電腦科學是著重在電腦理論與基本觀念的學科；軟體工程則是著重在開發和發行有用軟體相關的實用知識
軟體工程與系統工程有何不同？	系統工程是涵蓋電腦系統開發過程的各方面，包括硬體、軟體和程序等。軟體工程是這個過程的一部分
軟體工程面臨的主要挑戰為何？	要應付日益增加的多樣性、縮短開發時間的要求，以及開發出的軟體必須值得信任的要求
軟體工程的成本是什麼？	開發成本大約佔 60%，測試成本約佔 40%。對於客製化的軟體而言，其軟體演進成本經常會超過開發成本
什麼是最好的軟體工程技術和方法？	雖然所有的軟體專案都必須以專業方式來管理和開發，但是不同種類的系統，適合使用的技術也不同。例如遊戲軟體的開發應該是製作一連串的原型產品，而安全關鍵的控制系統則是需要開發完整且可供分析的規格。因此沒有哪種方法或技術能適用在所有情況
網際網路對軟體工程造成什麼不同影響？	網際網路不只促使業界開發出龐大、高度分散、以服務為主的系統，同時也支持了行動裝置使用的 App 軟體產業的蓬勃發展，後者已經改變軟體業界的經濟版圖

圖 1.1　軟體工程相關常見問題集

1.1 專業軟體的開發

1. **通用產品 (generic product)**：它是由軟體開發公司所生產的獨立系統，公開銷售給市場上的任何顧客。例如行動裝置上的App，還有PC上的軟體。特定用途所設計的，牙醫病歷系統等。
2. **客製化產品 (customized (bespoke) product)**：這類系統是專門為某個客戶特別訂做的系統。例如電子設備的控制系統、航管控制系統。

1.1 專業軟體的開發

- 這兩種不同類型的軟體最主要的差異在於，開發通用軟體產品的公司可掌控軟體的規格，因此假如在開發時遇上問題，公司可以重新思考改變開發方向。
- 客製化產品的規格則通常是由訂購此軟體的客戶來制訂與掌控，軟體開發人員必須遵循客戶所制訂的規格來開發。

1.1 專業軟體的開發

產品特性	說明
可接受度 (Acceptability)	軟體必須能夠讓它的設計目標使用者可以接受。這表示它必須是可理解和可用的，而且必須與使用者所使用的其他系統相容
可信賴度 (Dependability) 和保全性 (Security)	軟體的可信賴度包括可靠性 (reliability)、保全性及安全性 (safety)。在系統發生故障時，可信賴的軟體不應該造成實體或經濟上的損失。軟體應該進行保全措施，不讓惡意使用者存取或破壞系統
效率 (Efficiency)	軟體不應該浪費系統資源，例如記憶體和處理器時間。因此，效率包含了回應速度、處理時間、記憶體利用率等因素
可維護性 (Maintainability)	軟體在撰寫時必須注意，它將來要能夠隨著客戶的需求改變而演進。這是一項非常重要的特性，因為在變動的商業環境中，軟體的改變是無法避免的需求

圖 1.2　良好軟體的必要特性

1.1.1 軟體工程

- 軟體工程 (software engineering) 是一門著重在生產軟體各方面知識的工程學科，範圍從最開始的系統規格制訂，到系統上線後的維護階段都包括在內。

1. **工程學科**：他們會應用適當的理論、方法和工具，提出問題的解決方案；也體認到他們的工作必須受組織與財務的限制，尋找解決方案。
2. **生產軟體的各方面**：軟體工程不只包含軟體開發時的技術性過程，還有一些相關的活動，例如軟體的專案管理，以及支援軟體開發的開發工具、方法和理論。

1.1.1 軟體工程

- 軟體工程為何重要有兩個原因：
  1. 個人和社會越來越依賴先進的軟體系統。我們需要能夠經濟而快速的生產出可靠且值得信任的系統。
  2. 長期來看，使用軟體工程方法和技術來開發軟體系統，通常會比把它當作個人的程式設計專案直接寫程式來得便宜。如果未能成功使用軟體工程方法，將導致測試、品質管理，以及長期維護的成本都將提高。

1.1.1 軟體工程

- 下列4個基本的程序活動是所有的軟體程序都有的：
  1. **軟體規格制訂 (software specification)**：由客戶和工程師共同定義軟體的功能以及運作的限制。
  2. **軟體開發 (software development)**：設計與撰寫軟體。
  3. **軟體確認 (software validation)**：軟體必須經過確認是否符合客戶的需求。
  4. **軟體演進 (software evolution)**：軟體必須持續修訂，以符合客戶和市場的需求變化。

1.1.1 軟體工程

- 以下4個一般性的議題對於許多類型的軟體都有影響：
  1. **異質性**：近年來，軟體系統的運作逐漸被要求必須像跨網路的分散式系統，也就是它會包含各種不同類型的電腦及行動裝置。
  2. **企業與社會的變化**：由於新興經濟體的快速開發與新技術的發明，企業與社會正在以令人驚訝的速度改變中，它們必須能夠快速修改現有軟體和開發新軟體。
  3. **保全性與信任度**：隨著軟體深入我們生活的各個層面，軟體也必須能讓我們更信任才行。
  4. **規模**：目前已開發問世的軟體其規模變化很大，小到從穿戴式裝置內建的嵌入式系統，一直到能服務全世界的網際網路規模的雲端系統都有。

1.1.2 軟體工程多樣性

- 而實際上採用的具體方法、工具和技術，則依據開發軟體的機構、軟體類型，以及開發程序中牽涉到的人員而定。
- 世上並沒有一種萬用的軟體工程方法，可以適合所有的系統和所有的公司，反而是在過去的50年裡已經發展出形形色色的軟體工程方法和工具。

1.1.2 軟體工程多樣性

1. **單機應用程式**：這類應用程式是在個人電腦上執行的應用軟體，以及在行動裝置上執行的App。而且不需要連上網路就能操作。例子包括PC上的辦公室應用程式、旅遊App。
2. **交談式異動應用程式**：這類應用程式是在遠端電腦上執行，使用者是從他們的電腦、手機或平板來存取。例如電子商務應用程式。
3. **嵌入式控制系統**：這是負責控制和管理硬體裝置的軟體控制系統。以數量來看，嵌入式系統可能比其他任何一種系統的個數都更多。
4. **批次處理系統**：這類商業系統是設計成分批次 (batch) 來處理大量資料。

1.1.2 軟體工程多樣性

5. **娛樂系統**：這類系統主要是個人使用，目的是娛樂使用者。
6. **塑模和模擬用的系統**：這類系統是由科學家和工程師所開發，目的是自然程序或情況描述成為模型，內含許多分開而彼此互動的物件。
7. **資料收集和分析的系統**：資料收集系統是一種從環境中收集資料，並將資料傳送給其他系統處理的系統。
8. **由系統組成的系統**：這類系統是由許多其他的軟體系統組合而成。

1.1.2 軟體工程多樣性

- 有些軟體工程基本概念可適用在所有類型的軟體系統上：
  1. 它們應該使用某種能被掌控且理解的開發程序來開發。當然，不同類型的軟體會使用不同的程序。
  2. 無論是哪種系統，可信賴度和效能都是很重要的。
  3. 瞭解和管理軟體規格和需求 (軟體應該做的事) 是很重要的。
  4. 最有效率的利用。只要適合就應該再利用已經開發好的軟體，而不是撰寫新軟體。

1.1.3 網際網路軟體工程

- 一開始web主要是個全世界都可以存取的資訊庫，當時對軟體系統幾乎沒有影響。這些軟體系統在本機電腦上執行，而且只能從公司內存取。
- 大約到2000年時，web開始演進，越來越多的功能被增加到瀏覽器中。這意味著我們可以開發只需透過web瀏覽器而非專用的使用者介面即可存取的web系統。
- 所撰寫的軟體不必部署在使用者的PC上，而是部署在web伺服器上。

1.1.3 網際網路軟體工程

- 「軟體即服務」(software as a service，參見第17章) 這個觀念是在21世紀初提出的。
- 運算雲 (computing cloud) 是一群數量龐大而且相互連結的電腦系統，由眾多使用者共用。使用者不必購買軟體，但是根據軟體使用量的多少而付費。

1.1.3 網際網路軟體工程

- 軟體架構的這個重大改變，已經對web系統的軟體工程產生重大影響。比如說：
  1. 軟體再利用 (reuse) 已變成建構web系統的主流方式。當建立這些系統時，你要思考如何從現有的軟體元件和系統組合它們。
  2. 現在大家一般都已體認到，對這類系統要求事先定義好所有的需求是不切實際的。以遞增方式逐步開發和交付來進行。
  3. 軟體也可能以服務導向軟體工程方法來實作，該方法使用的軟體元件是獨立的web服務。
  4. 目前像AJAX (Holdener 2008) 和HTML5 (Freeman 2011) 這類的介面開發技術已經問世。

1.1.3 網際網路軟體工程

- 軟體工程基本觀念，同樣可以應用在web軟體上。

1.2 軟體工程的道德議題

- 身為一位軟體工程師，所牽涉的不只是如何應用個人的技能，還包含了更廣大的責任。軟體工程師若要讓別人以專業人士來看待，則行為上必須謹守倫理與道德的規範。
- 在可接受的行為標準裡，有一些模糊地帶並沒有受到法律的規範，而純粹是職業上的道德。例如：

1. **保密 (confidentiality)**：不管有沒有簽署正式的保密合約，工程師都應該遵守雇主或客戶的保密要求。
2. **稱職 (competence)**：工程師必須根據自己的能力做稱職的表現，不應該故意接受無法勝任的工作。

1.2 軟體工程的道德議題

3. **智慧財產權 (intellectual property right)**：工程師應該注意有關智慧財產權的法律相關規定，例如專利或版權。他們應該確保雇主和客戶的智慧財產有受到保護。
4. **濫用電腦 (computer misuse)**：軟體工程師不應該利用本身的技術與能力濫用其他人的電腦設備。濫用電腦的情況從單純的利用雇主機器玩電腦遊戲，到非常嚴重的散播電腦病毒等都算。

- 例如ACM、IEEE (Institute of Electrical and Electronic Engineers) 和英國電腦協會 (British Computer Society) 等組織，都有發行專業人員應該遵守的行為準則或道德規範，加入這些組織的會員必須接受並遵守這些規範。

1.2 軟體工程的道德議題

軟體工程的道德規範與專業守則

ACM/IEEE-CS 聯合工作小組制訂的軟體工程道德與專業守則

前言

這個簡短版本的規範是以較高階抽象的方式，摘要說明軟體工程專業人士應有的抱負。完整版本中的條文有加入範例和詳細說明，闡述這些抱負如何影響軟體工程專業人士的行為。如果沒有這些抱負，詳述部分就變成只是冗長的法律條文；但是如果沒有詳述的部分，這些抱負又會變成空談。因此，這些抱負和詳述合在一起形成了一套緊密的規範。

軟體工程師應該承諾讓軟體的分析、規格制訂、設計、開發、測試及維護等成為有益大眾且受尊重的專業。依照軟體工程師對大眾的健康、安全和福利的承諾，他們應該遵守下列 8 個準則：

1. PUBLIC（公眾）：軟體工程師應該維護大眾的利益。
2. CLIENT AND EMPLOYER（客戶和雇主）：軟體工程師應該以讓他的客戶和雇主得到最佳利益為職責，同時維護大眾利益為職責。
3. PRODUCT（產品）：軟體工程師應該確保他的產品和相關的修改儘可能符合最高的專業標準。
4. JUDGMENT（判斷）：軟體工程師在專業判斷上應該維持正直與中立。
5. MANAGEMENT（管理）：軟體工程的經理人和主管應該在軟體開發與維護上支持與提倡合乎道德的管理方法。
6. PROFESSION（專業）：軟體工程師應該提升專業的誠實與名譽，並符合大眾利益。
7. COLLEAGUES（同僚）：軟體工程師應該公平對待並支援同事。
8. SELF（自身）：軟體工程師應該在他的專業領域上終身學習，並且應該在專業領域上提倡合乎道德的方法。

圖 1.3 ACM/IEEE 道德規範 (ACM/IEEE-CS Joint Task Force on Software Engineering Ethics and Professional Practices, short version. http://www.acm.org/about/se-code) (© 1999 by the ACM, Inc. and the IEEE, Inc.)

1.2 軟體工程的道德議題

- 這些準則背後的緣由闡述在完整版中最前面兩段：
  - 電腦對於商業、工業、政府、醫療、教育、娛樂以及整體社會而言，扮演了非常重要的角色，而且其重要性持續的增加。軟體工程師可以透過直接參與或講授教學的方式，對軟體系統的分析、規格制訂、設計、開發、驗證、維護與測試等做出貢獻。
  - 由於軟體工程師參與了軟體系統的開發工作，所以他們有很好的機會對軟體系統做出好的貢獻或是造成傷害，也可以讓別人或影響別人做出好的貢獻或造成傷害。為了儘可能確保他們的努力是用在正途，軟體工程師必須承諾讓軟體工程成為有益的且受尊重的專業。

1.2 軟體工程的道德議題

- 「規範」(code) 部分包含與專業軟體工程師的行為和決策有關的8項「準則」(principle)，包括對自行開業者、教育人員、經理人、主管、政策制訂者以及受訓學員或在校學生等的規範。

1.2 軟體工程的道德議題

- 道德上的兩難。當你不同意公司管理高層的政策時，你要如何因應？
- 專業工程師遇到最麻煩的情況是當他的雇主不遵守道德時。假設某家公司負責開發一個非常重視安全性的系統，但是由於時間的壓力，他們偽造安全驗證紀錄。此時工程師的責任是為公司保密，還是要趕緊警告客戶這個系統可能不安全？
- 其他的道德問題例如參與軍事和核子相關的系統開發。

1.3 本書案例簡介

- 本書使用的案例系統類型為以下4種：
  1. **嵌入式系統**：針對糖尿病患設計的控制胰島素幫浦的軟體系統。
  2. **資訊系統**：病歷系統。
  3. **以感應器為主的資料收集系統**：荒野測候站。
  4. **學習支援環境系統**：來協助學生在校學習的數位學習環境為例。

1.3.1 胰島素幫浦控制系統

- 胰島素幫浦 (insulin pump) 是一種醫療系統，它模擬人體內胰腺的運作。這個系統是個由軟體控制的嵌入式系統，它從感應器收集資料，並控制幫浦輸送適當劑量的胰島素給病患。
- 這種系統是給糖尿病患使用的。
- 血糖若太低，短時間內將會造成暫時性的腦功能障礙，更嚴重則會造成失去意識，甚至導致死亡。血糖若偏高，在長時間下將會導致眼睛失明、腎功能損害以及心臟疾病等問題。

1.3.1 胰島素幫浦控制系統

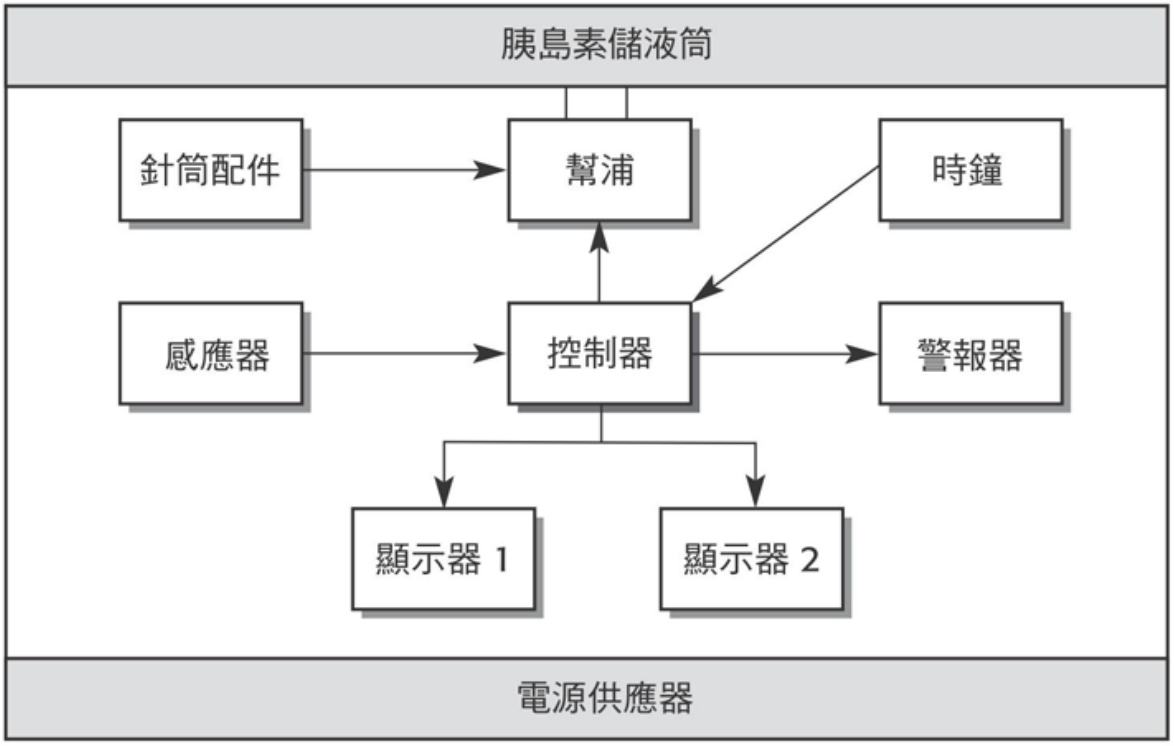


圖 1.4　胰島素幫浦硬體結構

1.3.1 胰島素幫浦控制系統

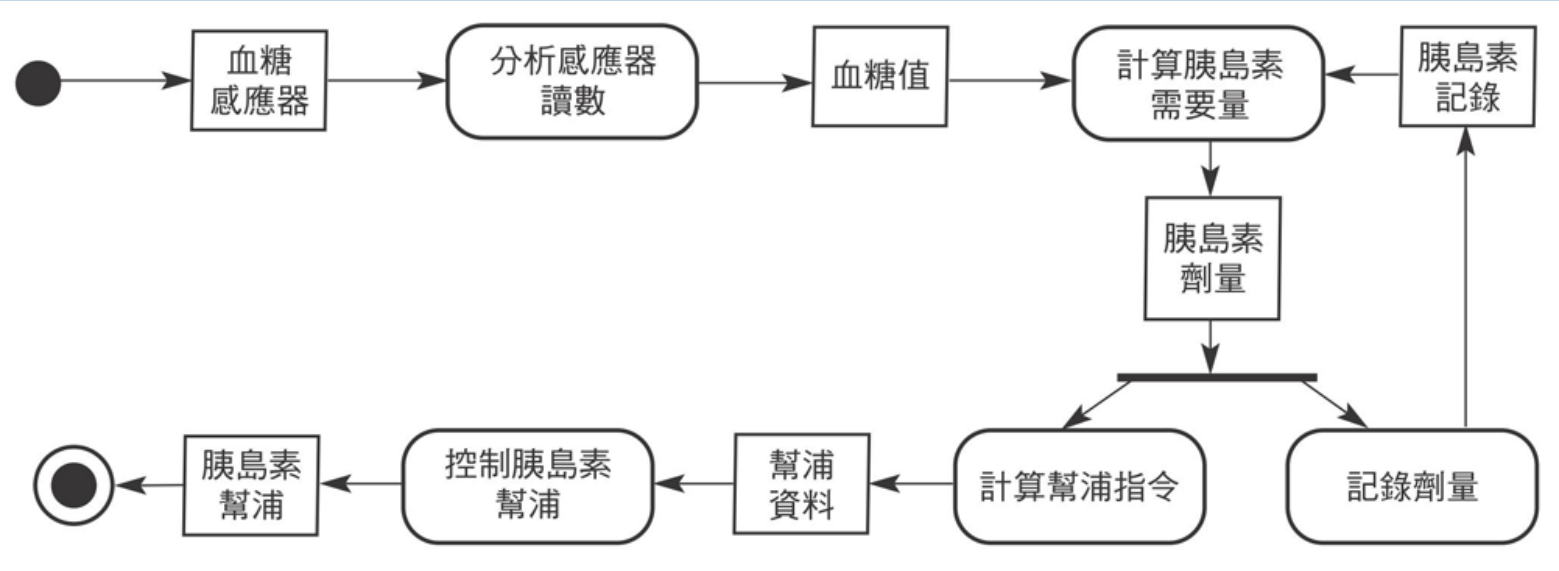


圖 1.5　胰島素幫浦系統的活動模型

1.3.1 胰島素幫浦控制系統

- 顯然這是個安全關鍵 (safety-critical) 系統。
- 因此這個胰島素給藥系統有2個必要的高階需求一定要達成：
  1. 在病患需要時，系統必須能夠傳送胰島素給病患。
  2. 系統必須根據目前的血糖程度，可靠的運作並傳送正確劑量的胰島素給病患。

1.3.2 心理保健病患資訊系統

- Mentcare系統(圖1.6)是一種病歷資訊系統，是專為診所使用而設計的。
- 它會使用到儲存病患資訊的集中式資料庫，但是也設計成在筆電上執行，如此一來即使是沒有安全網路連線的地點也可以存取和使用。假如本地系統上有安全的網路存取功能，除了可以使用資料庫裡的病患資訊外，當系統與資料庫中斷連線時，也可以下載病歷副本使用。

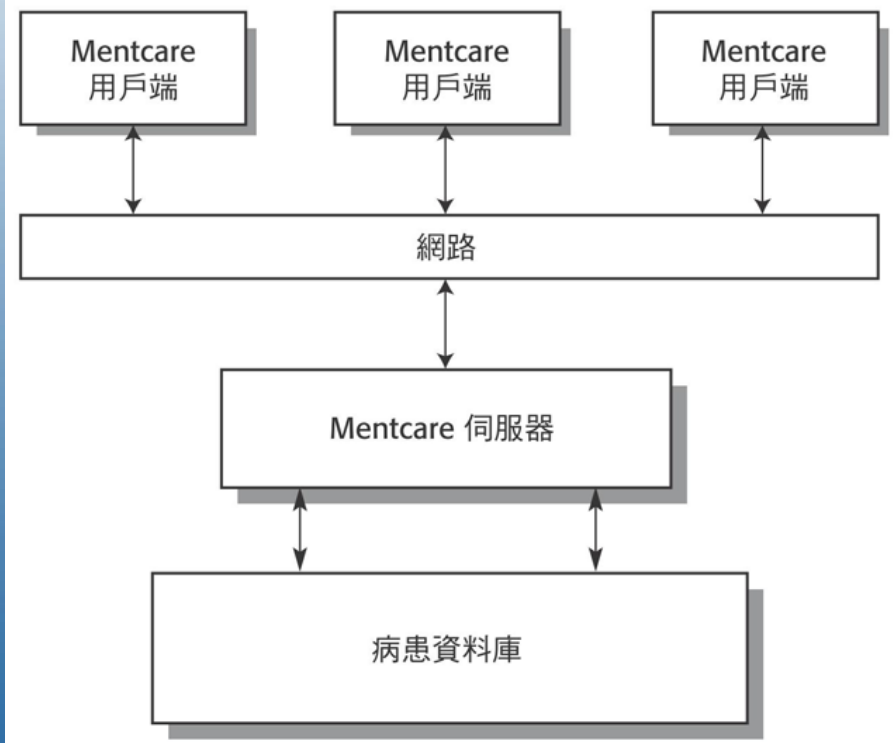


圖 1.6 Mentcare 系統的架構

1.3.2 心理保健病患資訊系統

- 該系統有兩個主要目的：
  1. 產生行政管理方面的資訊，讓衛生服務人員可評估其成效是否達到政府的目標。
  2. 提供醫療人員及時資訊協助治療病患。

1.3.2 心理保健病患資訊系統

- 此系統的重要功能如下：
  1. **個人保健管理**：診所醫生可以為病患建立新紀錄、編輯系統中的資訊、檢視病歷等。系統提供資料摘要功能，因此醫生在面對從未見過的病患時，仍然可以快速知道他的主要問題，以及之前做過什麼治療。
  2. **病患監控**：系統會定期監視病患的治療紀錄，假如偵測到可能的問題就發出警示。因此病患如果逾期一段時間沒有就診，可能就會發出警示訊息。
  3. **行政管理報表**：系統每個月產生管理報表，內容列出每家診所治療的病患人數、進出此照護系統的病患人數、隔離的病患人數、所開的藥物和他們的開銷等。

1.3.2 心理保健病患資訊系統

- 隱私也是此系統的關鍵需求。病患資訊是一定必須保密的，絕對不能洩露給除了經授權的醫護人員和病患自己以外的人知道。
- Mentcare系統也是一種安全關鍵系統，有些心理疾病會讓病患變得有自我毀滅傾向，或是可能危害其他人。每當可能發生這種情況時，系統應該警示醫護人員，病患有自殺或危害他人的可能性。
- 系統在有需要時必須要可以使用，否則就會危及安全性，而且可能無法為病患開立正確的藥物。

1.3.3 荒野測候站

- 政府決定在偏遠地區佈署數百個氣象站。這些氣象站收集來自一組儀器的資料，包括溫度、氣壓、日照、雨量、風速和風向等。

1.3.3 荒野測候站

- 圖1.7包括以下這些系統：
  1. **氣象站系統**：此系統負責收集天氣資料，進行某些初步的資料處理，並將結果傳輸到資料管理系統中。
  2. **資料管理與封存系統**：此系統從所有的荒野測候站收集資料，進行資料處理和分析，並將資料以某種格式封存。
  3. **氣象站維護系統**：此系統可透過衛星與所有的荒野測候站通訊，監控這些系統的運作情況，並提供問題報告。

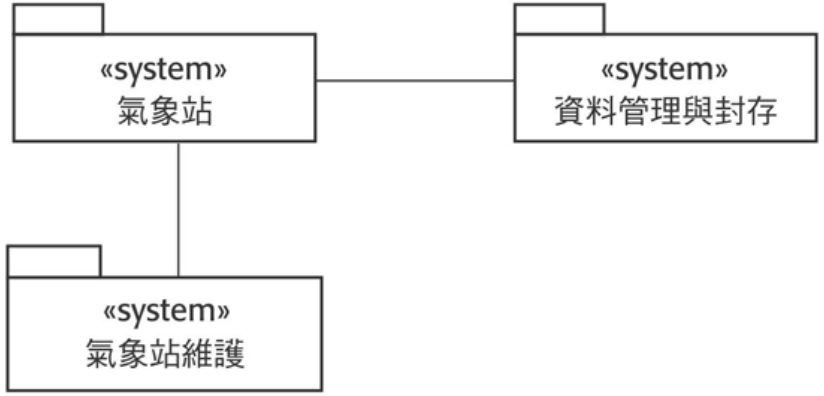


圖 1.7　氣象站的環境

1.3.3 荒野測候站

- 每個氣象站包含一些用來測量天氣參數的儀器，例如風速和風向、地面和空中溫度、大氣壓力，以及24小時內的降雨。這些儀器每個都是由一個軟體系統來控制，它會定期接收讀數，並管理收集自儀器的資料。

1.3.3 荒野測候站

- 因此氣象站軟體不只是要處理資料收集，它還必須：
  1. 監視儀器、電力和通訊硬體，發生故障要回報給管理系統。
  2. 管理系統電力，確保每當環境條件許可的情況下，電池就會充電；此外在可能有危險的天氣條件下 (例如強風) 也要關閉發電機。
  3. 當軟體部分升級時可動態調整組態，而且發生系統故障事件時，可將備用儀器切換到系統中使用。

1.3.4 在校使用的數位學習環境

- 數位學習環境是一個框架 (framework)，內含一組學習工具，其中有些工具是一般用途，有些是專為學習所設計；另外還加上專為針對使用此系統的學習者他們的需求所量身打造的一組應用程式。
- 該環境可打造出多種版本，由教師和學生根據各自需求選擇適合的幾種工具所組成。它們可能是一般的應用程式，如試算表、遊戲、模擬軟體，以及負責管理交作業和評分的學習管理應用程式，例如Virtual Learning Environment (VLE)。

1.3.4 在校使用的數位學習環境

圖 1.8 數位學習環境 (iLearn) 的架構

1.3.4 在校使用的數位學習環境

- 這套系統是個服務導向 (service-oriented) 系統，裡面所有的系統元件都被視為可替換的服務。
  1. **公用 (utility) 服務**：這類服務提供一些與應用程式不相依的基本功能，而且系統中的其他服務也可以使用。
  2. **應用程式 (application) 服務**：這類服務提供特定的應用程式，如電子郵件、會議、照片分享等；而且可存取特定的教育用資源，如科學影片或歷史資料。
  3. **組態設定 (configuration) 服務**：這類服務是用來設定環境讓它適應某一組特定的應用程式服務，以及定義各個服務如何讓學生、教師和家長分享使用。

1.3.4 在校使用的數位學習環境

- 系統的服務整合方式有兩種層級：
  1. **整合式 (integrated) 服務**：這類服務的整合方式是提供某個API (application programming interface) 介面，其他服務經由此API可存取該服務。所以服務對服務的直接通訊是有可能的。
  2. **獨立式 (independent) 服務**：這類服務是只經由瀏覽器介面存取的服務，而且其運作與其他服務獨立不相關。如果想要與其他服務共用資訊，就只能藉由使用者直接操作，例如複製後貼上。此外，每個獨立式服務可能需要各自進行驗證使用者的手續。當獨立式服務逐漸被廣泛使用時，開發團隊就可以考慮將該服務整合進系統，將它轉變成整合式服務。