以網路分析法為基礎之敏捷功能點計算模式

韓文銘*
德明財經科技大學資訊管理系

張均漢
德明財經科技大學資訊科技與管理研究所

摘要

功能點分析是目前最廣泛被使用的軟體功能性規模度量方法論，儘管其定義的 43 條自然語言法則可以提供量化的規模度量值來支援成本與時程預估的活動，然而自然語言法則的模糊性與功能複雜度權重判斷的困難性亦導致實務使用上的挑戰。本研究整合 ANP 與 FPA 以發展敏捷功能點計算模式，透過 5 個主要度量步驟：建立群組層級架構、定義功能相依性、評估功能相對複雜性、計算功能相對複雜性權重以及計算敏捷功能點數，不僅可以考量功能相依性以完整反映系統功能全貌，也可以減少 FPA 自然語言判斷規則的使用數量以加快度量時間與克服複雜度權重值久未更新的問題，此外，本研究亦透過發票處理系統的實務個案度量來驗證本研究成果的應用可行性。

關鍵詞：軟體專案度量、功能點分析、軟體規模、網路分析法、軟體專案管理

* 本文通訊作者。電子郵件信箱：wmhan@takming.edu.tw
2011/02/16 投稿；2011/07/05 修訂；2011/10/05 接受

An Agile Function Point Counting Approach Based on ANP

Wen-Ming Han*
Department of Management Information Systems, Takming University of Science and Technology

Chun-Han Chang
Graduate Institute of Information Technology and Management, Takming University of Science and Technology

Abstract

Functional Point Analysis (FPA) is a widely adopted approach that measures the functional size of software systems. This approach uses 43 rules for counting function points provide a quantifiable size measure to support the cost and schedule evaluations. These evaluations are complex and stretch their use of new practical questions. This study integrated Analytic Network Process and FPA to develop agile function point counting approach. Through 5 major steps: the establishment of hierarchical structure, the definition of functional dependencies, the assessment of functional relative complexity, the calculation of relative complexity weights and the calculation of agile function point, which can consider the functional dependencies so as to reflect a complete profile of system function and reduce used natural language rules of FPA to improve the efficiency of measurement and overcome the un-updated problem for weighted values of functional complexity. An invoicing system is presented to illustrate the use of this approach to confirm the feasibility of this approach.

Keywords: Software project measurement, Functional point analysis, Software size, Analytic network process, Software project management

* Corresponding author. Email: wmhan@takming.edu.tw
2011/02/16 received; 2011/07/05 revised; 2011/10/05 accepted

壹、研究背景與動機

軟體規模（Software Size）是一個持續受到國內外學者重視且廣泛討論的研究議題，因為其不僅是工作量、成本和時程預估的重要基礎，亦是基準比較分析的必要度量單位，在眾多估算軟體規模的方法中，由 IBM 工程師 Allan Albrecht（1979）於 1979 年所提出之的功能點分析（Functional Point Analysis; FPA）是最受到學術界與實務界廣泛討論與建議使用來度量軟體規模的方法論（Albrecht 1979; Albrecht & Gaffney 1983），原因是其計算架構定義嚴謹以及具備實證研究支持。

FPA 的計算架構可以概分為三大步驟：(1)識別五個基本功能性元件（內部邏輯檔案、外部介面檔案、外部輸入、外部輸出與外部查詢）的個數；(2)依據 DET、RET 與 FTR 的個數多寡來判斷基本功能性元件（Basic functional component; BFC）的複雜度等級；(3)使用功能點實務手冊（Counting Practices Manual; CPM）所定義的計算公式求得未調整功能點數（IFPUG 2010）。依據 CPM 4.3.1 的規範定義，步驟一共有 24 條判斷規則，步驟二共有 19 條判斷規則，隨著軟體功能性需求日趨複雜與具有高變動性的趨勢，專案管理者的度量工作負擔將日漸加重與困難。

研究結果顯示使用 FPA 所產生的度量結果誤差值可達到 30%（Low & Jeffery 1990; Christopher 1999; Wei et al. 2008），其主要原因可以歸類於以下二點：(1)使用者缺乏經驗或未受過完整的教育訓練以致於錯誤使用自然語言型態的判斷規則；(2)功能複雜度的等級判斷難以在軟體開發早期就可以完整確定以致於錯估軟體規模。此外，就實務觀點而言，軟體是由一連串相互依賴的功能流程所組成以完成使用者的期望需求，因此近代研究亦已證實 BFC 間具有相依性（Kitchenham et al. 1993; Jeffery et al. 1996; Turetken et al. 2008），持續忽略 BFC 存在相依性不但可能導致無法反映所欲度量軟體功能性規模的全貌，也可能使得高估或低估軟體規模的機率上昇，並進而導致成本與時程預估失真。

基於真實世界的問題複雜性是非相互獨立與不易被層級分解的事實，Saaty（1996）提出考量相依（Interdependent）與回饋（Feedback）機制的網路分析法（Analytic Network Process; ANP），此法係透過其所定義之超級矩陣計算以實現量化表達元素間相依性的觀點，目前 ANP 已經被廣泛應用於許多研究領域問題的探討，包含 R&D 專案評選（Meade & Presley 2002）、軟體架構評估方法設計（Jihyun et al. 2009）與財務危機預測（Niemira & Saaty 2004）等，而目前已被研究學者廣泛應用的層級分析法可被視為 ANP 的特例。

由於 ANP 能夠表達與量化元素間相互依賴及互相影響的觀點，此項特點正

好可以用來改善 FPA 的不足之處，即假設功能獨立以及複雜性判斷規則繁瑣與權重久未更新，因此本研究整合 ANP 與 FPA 以發展敏捷功能點計算模式。本篇研究的架構是由五個章節所組成，第一章是描述與強調研究背景與動機；第二章是以文獻回顧的方式來探討功能點分析與網路分析法的特性與計算步驟；第三章是具體描述實行本計算模式的五個步驟；第四章是以個案探討的方式來協助讀者瞭解本計算模式的應用步驟與驗證應用可行性；結論與未來研究方向則是在第五章整理說明。

貳、文獻回顧

一、功能點分析

軟體規模（Software size）是資源分配、成本預估與時程安排的重要決策基礎，此外，軟體規模也可以擔任基礎度量值（Base measures）的角色，以產生具備基準比較意義之衍生度量值（Derived measures），例如單位成本與平均缺失數等。在 1980 年代以前，我們均是透過程式碼行數（Line of Code; LOC）來量化欲開發之軟體產品的規模，然而受限於程式碼行數具有先天性的缺點（例如與程式語言種類相關以及無法在專案早期得知等），利用程式碼行數做為資料分析的基礎以支援軟體專案管控的代表性與價值並不高（Halstead 1977; Fenton & Pfleeger 1996）。

功能點分析是 Allan Albrecht（1979）所發展的軟體規模度量方法論，量化軟體產品規模的單位是以功能點（Function Point; FP）來表示，具有與技術平台無關及可以跨專案/組織基準比較的優點等，因此受到學術界與實務界廣泛地討論、重視與使用。圖 1 顯示以 Albrecht 所提出之功能點分析為基礎構想所延伸發展的 5 個已成為國際標準的功能性規模度量方法，分別是 ISO 19761（2003）、ISO 20926（2009）、ISO 20968（2002）、ISO 24570（2005）與 ISO 29881（2010），表 1 顯示這五種功能性規模度量國際標準的分析比較，而目前功能點分析是由國際功能點使用者協會（International Function Point Users Group; IFPUG）持續維護以使其度量規則可以符合現今資訊科技環境。

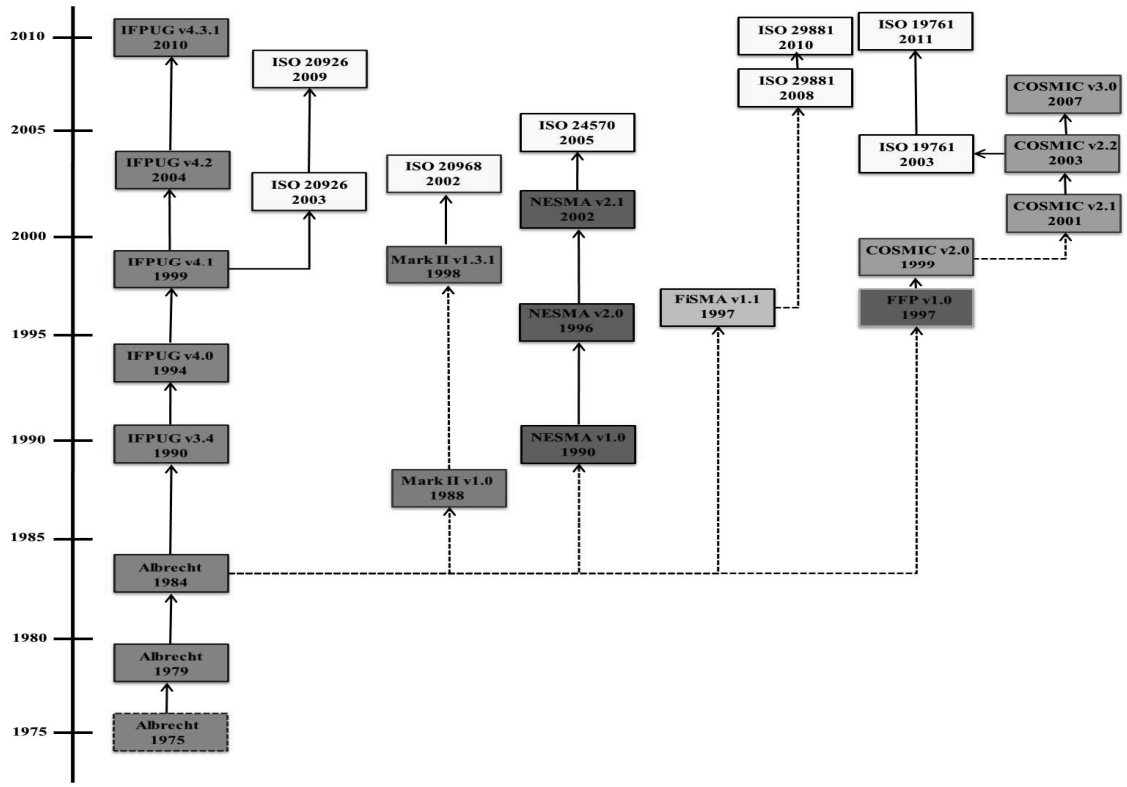

圖 1：功能性規模度量方法論的歷史進展趨勢

表 1：五種國際功能性規模度量標準的分析比較

	ISO 19761	ISO 20926	ISO 20968	ISO 24570	ISO 29881
標準前身	COSMIC	FPA	Mark II	NESMA	FiSMA
最新版本	2011	2010	2002	2005	2010
功能性領域	即時系統	MIS	所有系統	所有系統	所有系統
應用頻率	高	高	低	低	低

功能點分析的計算架構是由 5 個功能性元件所組成，分別是內部邏輯檔案、外部介面檔案、外部輸入、外部輸出與外部查詢，並以檢視系統開發相關文件為基礎來識別此 5 種功能性元件的數量與複雜度，彙總此 5 種功能性元件所產生之功能點數即為此系統的總功能點數，圖 2 顯示此 6 個度量步驟以及相關所需要使用到的 43 條自然語言判斷規則數量，圖中的數字代表自然語言識別規則的使用數量，例如內部邏輯檔案與外部介面檔案分享相同的 6 條資料功能性識別規則，而各個步驟所要進行的工作項目內容茲簡述如下：

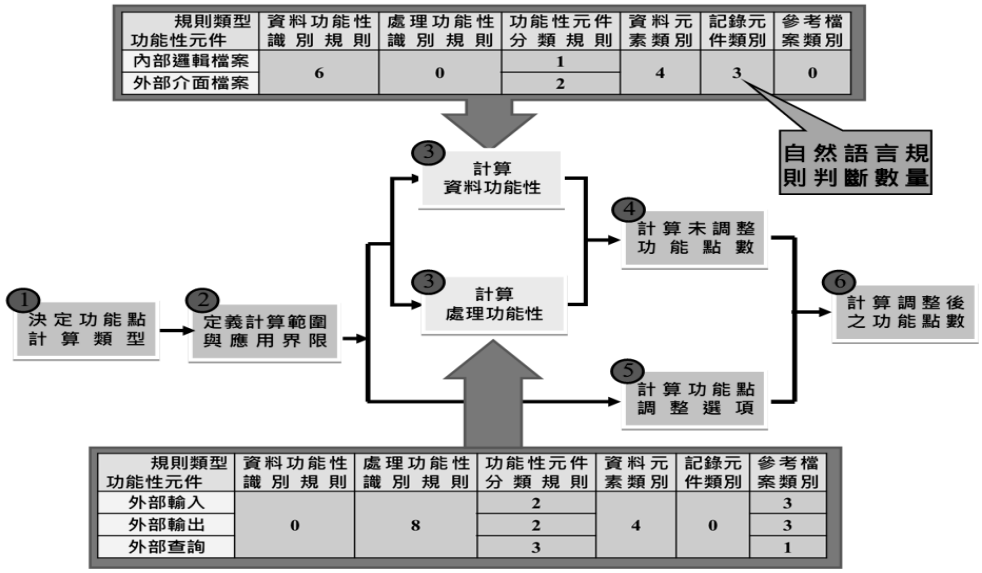

圖 2：功能點分析的度量步驟與相關法則數量

步驟一：決定功能點計算類型

使用 FPA 的第一步驟是定義計算類型，因為計算公式會依據所決定之計算類型而有所差異。在功能點實務手冊將計算類型區分為三種：「開發專案功能點計算」是指當專案完成且欲交付給使用者進行首次安裝時的度量，例如欲瞭解剛開發完成之 Web 系統的軟體規模；「增強專案功能點計算」是指欲新增、修改或刪除已交付系統功能時的度量，例如針對購物功能新增與註冊功能修改所衍生的軟體規模；「應用功能點計算」是指度量使用者某系統的現存功能，例如度量 ERP 系統的軟體規模。

步驟二：定義計算範圍與應用界限

當計算類型決定後就必須要定義計算範圍與界限，亦即界定有哪些功能須納入或排除到功能點分析計算中。例如所要面臨到的思考課題為在量化財務系統的軟體功能性規模時，是否要將匯率轉換系統的功能納入計算，然而此答案要視其管理需求而定即並沒有絕對的要或不要納入。

步驟三：「計算資料功能性」與「計算處理功能性」

此階段為計算欲度量之軟體系統的資料功能性（ILF 與 EIF）與處理功能性（EI、EO、EQ）的功能性規模大小。此步驟會將系統所有功能需求逐一分類為上述的五種功能性元件，並針對每一個功能性元件判斷其所屬的複雜度等級，例如參照圖 2 可以發現識別一個外部邏輯檔案需要 8 條自然語言判斷規則，而將此識別出的內部邏輯檔案判斷其所屬複雜度等級則需要 7 條自然語言判斷規則。

圖 3 為此 5 種功能性元件所使用的複雜度等級判斷矩陣，資料功能性的複雜度矩陣是由資料元素類別（Data Element Type; DET）與記錄元素類別（Record Element Type; RET）所組成，而處理功能性的複雜度矩陣是由資料元素類別與參考檔案類別（File Type Referenced; FTR）所組成，此外內部邏輯檔案與外部邏輯檔案的複雜度等級判斷矩陣是一樣的，而外部輸出與外部查詢的複雜度等級判斷矩陣也是一樣的。例如若是有一個內部邏輯檔案其資料元素類別與記錄元素類別的個數各為 25 個與 4 個，則此內部邏輯檔案的複雜度等級為中等複雜。

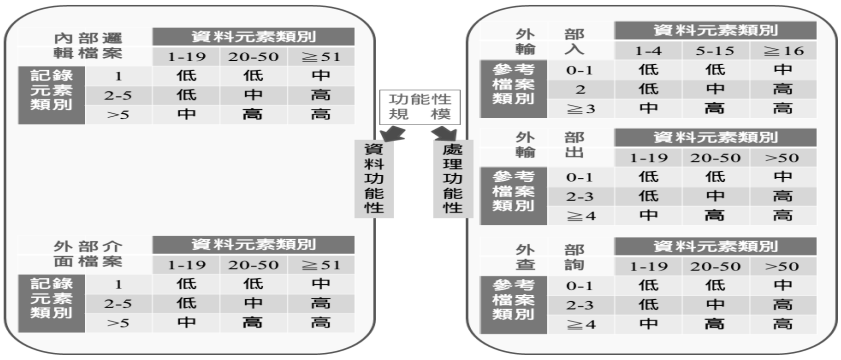

圖 3：功能性元件的複雜度等級矩陣

步驟四：計算未調整功能點數

實行完步驟三後，我們就可得知所預度量之軟體系統的功能性元件數量及其複雜度等級，將此資訊與 FPA 所制定的功能性元件權重分配表進行相乘加總即可計算出未調整功能點數（Unadjusted Function Points; UFP），圖 4 顯示 FPA 所制定的功能性元件權重分配表及其計算範例。

圖 4：功能性元件權重分配表及其計算範例

步驟五：計算功能點調整選項

此階段是依據 14 個一般系統特徵值的總影響程度來計算調整因子數值（Value Adjustment Factor; VAF）。每一個特徵值的影響程度之值介於 0 至 5 的尺度範圍，即從無影響至強烈影響等程度變化。由於「調整因子數值」對於軟體功能性規模的解釋能力一直是個爭論的課題，所以在 ISO/IEC 20926（2009）中並沒有包含此步驟的內容，而 CPM 4.3 則是將此步驟定義為「選項」，亦即可由專案管理者的需要來彈性選擇是否應該要應用。

步驟六：計算調整後之功能點數

此階段為功能點計算的最後一個步驟，主要目的為依據步驟一所決定的計算類型來計算調整後的功能點數（Adjusted Function Point; AFP），例如應用功能點數的計算公式如下，若是不把步驟 6 的 VAF 納入調整則 AFP 就會等於 UFP，因此近代功能點分析文獻均是以 UFP 來當做軟體功能性規模。

$$AFP = UFP * VAF \quad (1)$$

由上述的介紹可以發現隨著軟體功能性需求的複雜化與高變動性，度量人員的工作負擔將持續加重，實證研究亦指出不熟悉自然語言判斷規則所導致的度量結果誤差可達到 30%（Low & Jeffery 1990; Christopher 1999; 胡衣臨 & 黃世禎 2004; Wei et al. 2008），此外，專案早期難以精確計算 DET、RET 與 FTR 的個數亦會增加計算功能點的困難性（劉青峻 & 劉文卿 2003）。而目前探討功能點分析的國內外研究大都將焦點集中於度量應用、工具自動化與精確度改善的研究議題上，例如：以功能點分析來量化度量各種軟體類型之規模與支援委外成本預估（黃明祥等 2008; Koh et al. 2008），自動化轉換計算實體關係圖（Entity Relationship diagram）與統一塑模語言（Unified Modeling Language; UML）的功能點數（Lamma et al. 2004; 彭智賢 & 黃怡誠 2005），案例探討有哪些潛在因素會影響功能點數計算（Symons 1988; 劉青峻 & 劉文卿 2003; 祝淑蘭等 2006）。

就實務觀點而言，軟體是由一連串相互依賴的功能流程所組成以完成使用者所期望的需求，例如購物車功能牽涉到訂購流程、客戶登入/註冊流程與付款流程等，上述這些功能流程若沒有相互溝通協同運作是不可能實現購物車功能的需求，所以近代學者已經發現 BFC 間存在相依關係（Kitchenham & Känsälä 1993; Jeffery et al. 1996; Turetken et al. 2008），例如，Kitchenham 等（1993）利用 Kendall 等級相關係數（Kendall's coefficient of rank correlation）指出 4 種功能性元件間的相關性，分別是外部輸出與外部輸入、外部輸出與外部查詢、外部輸入與外物介面檔案、外部輸出與內部邏輯檔案；不同於 Kitchenham 等（1993）的研究結果，Jeffery 等（1996）搜集 14 筆專案資料並利用皮爾森相關係數

（Pearson's correlation coefficient）發現 3 種功能性元件間的相關性，分別是內部邏輯檔案與外部輸入、內部邏輯檔案與外部查詢相關、外部輸入及外部查詢相關，儘管上述功能性元件實證研究的結果並不完全一致，但是均指出功能性元件間並非相互獨立，因此我們相信 BFC 間的相依性關係亦有可能會使得功能點分析的計算結果產生失真。

二、網路分析法

當決策問題越複雜時，獨立性的假設與過度簡化的層級架構即無法完整呈現問題原貌，有鑑於此，Saaty（1996）教授於 1975 年提出同時考量相依（Interdependent）與回饋（Feedback）特性的網路分析法（Analytic Network Process; ANP)，此方法最主要的特色（優點）是以超級矩陣（Supermatrix）的方式來呈現與量化元素間具有相依影響關係的課題，即評估群組與群組間的外部相依（Outer dependence）以及評估同一群組中各元素彼此相互影響的內部相依（Inner dependence），其基本結構如圖 5 所示。

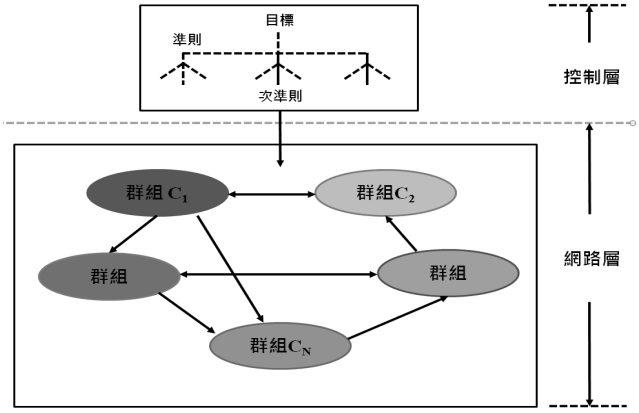

圖 5：ANP 的基本結構

目前已廣泛被應用的層級分析法（Analytical Hierarchy Process; AHP）可被視為 ANP 的延伸特例，因此根據 Saaty 教授的規範定義，應用 ANP 來解決實務問題仍需要有以下 7 點假設前提：

1. 層級內的元素可用上一層級內某些或所有元素作為評判準則以進行評估。
2. 進行比較評估時，可將絕對數值尺度轉換成比例尺度。
3. 成對比較後，可使用正倒值矩陣處理。
4. 偏好關係滿足遞移性。不僅優劣關係滿足遞移性，強度關係也滿足遞移性。

5. 完全遞移性不容易，因此容許不具遞移性的存在，但需測試一致性的程度。
6. 元素的優勢程度是經由加權法則（Weighting Principle）而求得。
7. 結構中的任何元素均被視為與整個評估結構有關，而非檢視結構的獨立性。

以上 7 點假設若是額外加上 2 點假設：「一個系統可被分解成許多種類或成份以形成層級架構」與「層級結構中的每一層級元素均具獨立性」即為 AHP 的 9 點假設，因此這也代表 ANP 與 AHP 的最大差異之處在於是否將問題領域視為獨立或相依。ANP 的實行流程可以分解成以下四個步驟（Saaty 2003; Niemira & Saaty 2004; Jihyun et al. 2009）：

步驟一：問題界定與模式建構

首先需要定義欲解決的問題（目標）並將問題逐步推展以建構成網路型態，例如想要探討速食業市場佔有率的課題，可以由競爭者、廣告、食物品質與其它這四個構面準則來思考，而每一個構面準則又可再細分成子準則，例如食物品質構面準則可再細分為營養、口味與搭配準則等，而其它則可以細分為價格與地點等。在此階段可以採用腦力激盪法、德菲法、群體決策法與決策實驗室分析法等技術來確保模式架構的完整性與正確性。

步驟二：成對比較矩陣與相對權重之計算

當模式建構後即可以開始進行準則間的成對比較，此成對比較的過程可以分為兩個部分，分別是群組間成對比較（例如成對比較競爭者、廣告、食物品質與其它構面準則）與群組內元素成對比較（例如成對比較食物品質構面準則下的營養、口味與搭配準則），此外，群組內元素成對比較又可再分為同一群組內的成對比較（例如成對比較食物品質構面準則下的營養、口味與搭配準則）與不同群組元素的成對比較（例如成對比較食物品質構面準則下的搭配準則與其它準則下的價格準則）。

Saaty（2003）建議以李克特 9 等級量表來進行成對比較的量化評估，所以若是有 n 個評估準則，就需要進行 n(n-1)/2 次的成對比較，量化比較結果可以組成一個成對比較矩陣，此成對比較矩陣的結構可以分為三個部份：上三角、下三角與對角線，上三角部份為量化比較結果；下三角部份為上三角相對位置數值的倒數，例如營養是口味的三倍重要會置於上三角，而反之口味是營養的 1/3 倍重要則是置於下三角；對角線的值均為 1 是代表準則自身的比較結果，圖 6 為成對比較矩陣的範例。最後，由於決策者對於各準則的成對比較判斷有時會產生前後不一致情形，Saaty 提出可以使用一致性指標（Consistency index, CI）及一致性比率（Consistency ratio; CR）來檢驗成對比較結果是否具有前後一致性的參考價值。

	營養	口味	搭配
營養	1	3	2
口味	1/3	1	6
搭配	1/2	1/6	1

圖 6：成對比較矩陣的範例

步驟三：建立超級矩陣

ANP 利用超級矩陣來呈現與量化元素間的相依強弱關係，所以超級矩陣內的數值代表兩個元素間的相依程度，而 0 即則代表元素間不存在相依性關係（彼此獨立），ANP 定義超級矩陣是由多個子矩陣（sub-matrix）所構成，每一個子矩陣內所顯現的相對權重（值）均是來自於矩陣內元素間的成對比較結果（特徵向量）。

超級矩陣的運算過程可再細分為三個矩陣，分別為未加權超級矩陣（Unweighted supermatrix）、已加權超級矩陣（Weighted supermatrix）以及極限化超級矩陣（Limiting supermatrix）。其中，通過一致性檢定的原始成對比較矩陣即為未加權超級矩陣，而已正規化行向量的未加權超級矩陣則為已加權超級矩陣，此已加權超級矩陣的建立是為了符合隨機原則（Column-Stochastic），即行值總和等於 1，最後，已加權矩陣自乘多次方會逐步收斂且呈現每一列欄位數字相等，此具有固定權重值的矩陣即為極限化超級矩陣或可稱之不可分解矩陣（Irreducible matrix），而矩陣內的權重值即為考量相依性的相對權重值，圖 7 為超級矩陣的示意圖。

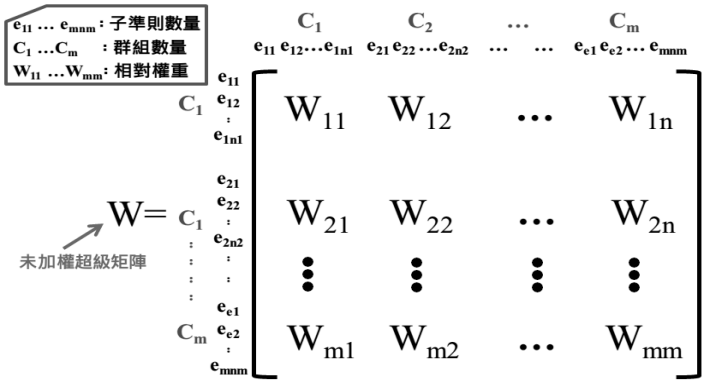

圖 7：超級矩陣的範例

由於超級矩陣計算過程的繁瑣可能會導致 ANP 不易被推廣及應用，Saaty 教

授為此發展 Super Decision 軟體（http://www.superdecisions.com/），此軟體可以「視覺化」群組架構、量化評估元素間的成對比較與檢定評估結果的一致性，改善傳統使用 EXCEL 來計算 ANP 的不便。

步驟四：決定與排序重要性

根據上述的步驟，即可以求得已考量相依性之各個準則的相對權重，權重愈大者代表愈重要，例如食物品質構面準則下的營養、口味與搭配準則之相對權重分別為 0.3、0.4 與 0.1，則我們就可以得知口味是最重要且該關注的準則，因此就會將口味設定為優先改善的首要目標。

ANP 的最大特性（優點）為可量化表達相依性觀點，如圖 8 所示，因此相較於假設功能彼此獨立的觀點，ANP 具有能更真實反映呈現系統功能間相互依賴的協同合作處理精神。而上述 ANP 的優點正好可以用來改善 FPA 的不足之處，即假設功能獨立以及複雜性判斷規則繁瑣與權重久未更新，因此本研究將 ANP 與 FPA 進行整合以建構敏捷功能點計算模式。

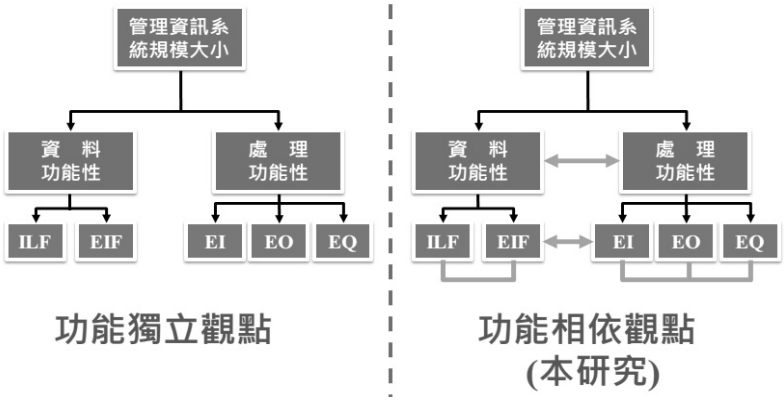

圖 8：管理資訊系統的功能相依觀點

參、敏捷功能點計算模式

敏捷功能點計算模式共包含 5 個度量流程，如圖 9 所示，每個度量流程下均有一組相關的工作活動以循序漸進地實現敏捷功能點數的計算，以下各小節為各個度量流程的概要說明，本研究亦透過第四章的個案來引導讀者更深入地瞭解本研究成果的應用方式。

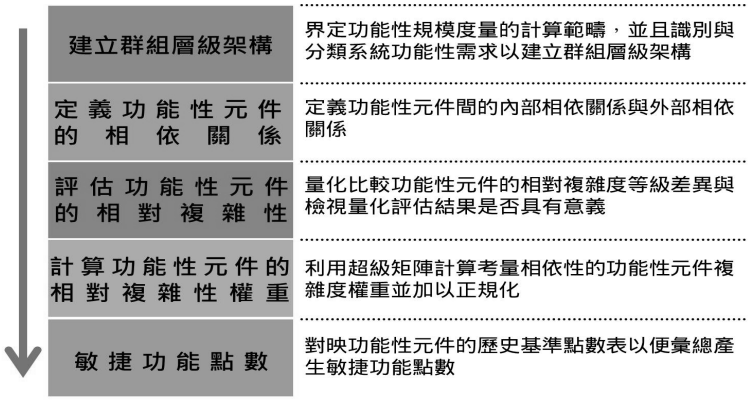

圖 9：敏捷功能點計算模式的度量流程

一、建立群組層級架構

此步驟包含 2 個工作活動以建立群組層級架構，分別是界定計算應用範疇與識別功能性元件。根據圖 5 的 ANP 基本結構，我們將本計算模式的群組架構分為 3 層，第一層是「目標」：定義我們欲度量資訊系統的功能性規模大小；第二層是「準則」：定義資訊系統的功能性規模大小可以分為 5 個功能性元件構面（內部邏輯檔案、外部介面檔案、外部查詢、外部輸入與外部輸出）；第三層是「次準則」：描述每一個功能性元件構面下所包含的功能性需求數量，如圖 10 所示。由於群組架構的建立與準則設定均是採用 FPA 的架構定義，因此並不會產生群組架構不完整與缺乏代表性的問題，而 BFC 的識別則是使用 FPA CPM 4.3.1 既有的 23 條規則以將功能性需求識別與分類為 ILF、EIF、EI、EO 與 EQ。

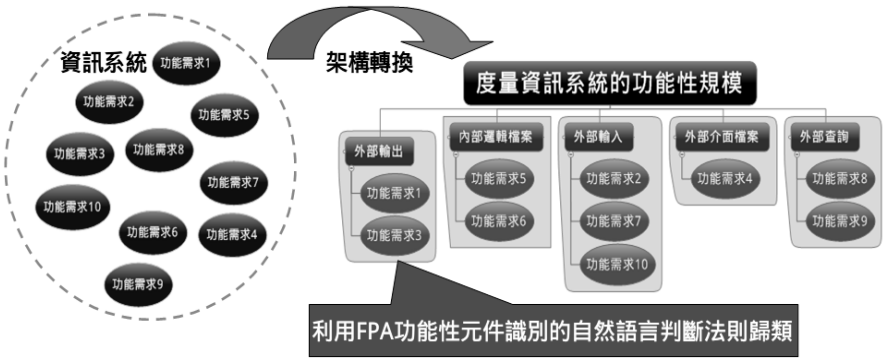

圖 10：敏捷功能點計算模式的度量流程

二、定義功能性元件的相依關係

此步驟包含 2 個工作活動以反映系統功能真實全貌，分別是定義外部相依性與內部相依性。由於功能相依性關係會隨時系統類型不同而有所差異，因此組織必須要儘可能地針對目前所要開發的資訊系統來搜集有利於系統相依性判斷的資料，容易在開發階段早期取得且有價值的判斷資料包含有業務計劃書（Business case）、建議書徵求文件（Request For Proposal; RFP)、資訊徵求說明文件（Request For Information RFI）、專案工作說明書（Statement of Work）、合約（Contact）與組織歷史專案資產（Organization historical project asset）等，而可以支援整合功能相依性關係判斷的技術包含有專家訪談、焦點團體、群體決策技術與座談會（開發者、使用者與顧問）等。

三、評估功能性元件的相對複雜性

此步驟包含 2 個工作活動以量化比較功能性元件的相對複雜度等級差異以及檢視相對複雜度等級差異評估結果是否具有意義，分別是成對比較與檢定判斷一致性。根據 Saaty 教授的建議（Satty 2003），本計算模式是使用李克特 9 等量表來進行功能性元件的兩兩比較量化評估（由非常複雜到非常不複雜），如表 2 所示，優點是愈多的量表尺度等級也代表愈能夠呈現評估者所想要表達的細微複雜差異。圖 11 顯示外部輸入之功能需求間的成對比較方式。

表 2：功能複雜度比較的 9 等距量表

評估尺度	尺度定義
1	兩個功能需求「同等複雜」
3	前項功能需求較後者功能需求「稍微複雜」
5	前項功能需求較後者功能需求「複雜」
7	前項功能需求較後者功能需求「非常複雜」
9	前項功能需求較後者功能需求「超級複雜」
2,4,6,8	中間補值，折衷值介於之前評估尺度間

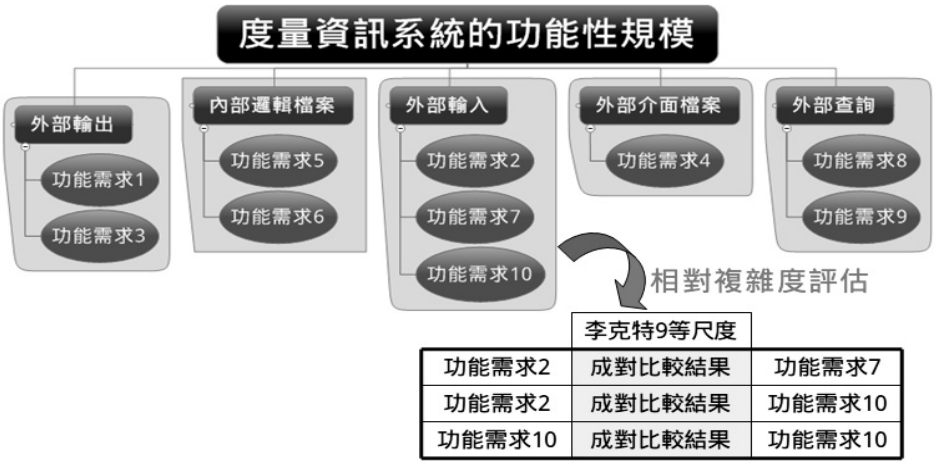

圖 11：外部輸入之功能需求間的成對比較方式

一旦功能性元件相對複雜度等級判斷結束後，即需要進行一致性檢定以確認該評估判斷結果是否具有前後一致的參考價值，例如有 3 個功能性元件，分別是功能性元件 1、功能性元件 2 與功能性元件 3，功能性元件 1 是功能性元件 2 的 2 倍複雜且功能性元件 1 是功能性元件 3 的 3 倍複雜，若另一個成對比較結果為功能性元件 2 比功能性元件 3 的複雜則代表具備遞移性（符合 ANP 的第 3 與第 4 點假設），反之則代表成對比較結果（功能性元件 3 比功能性元件 2 的複雜）可能有誤。上述遞移性關係的確認需使用 Saaty 所提出之一致性指標（Consistency index; CI）及一致性比率（Consistency ratio; CR）來做為判斷依據，一致性指標與一致性比率的計算公式分別如下(2)與(3)所示：

$$CI = \frac{(\lambda_{mzx} - n)}{(n-1)} \tag{2}$$

$$CR = CI/RI \tag{3}$$

當一致性指標為 0 時，代表前後判斷完全一致；一致性指標大於 0 時，則表示前後判斷不一致；一致性指標小於 0.1 時為可容許的偏誤，只有在一致性比率小於等於 0.1 時，代表成對比較結果具有參考性，此時才適合進行超級矩陣計算，反之則代表決策者必須重新思考功能性元件間的成對比較結果。而 CR 公式中的隨機指標（Random index; RI）則是需要經由查表獲得，如表 3 所示。例如有 3 個元素要進行成對比較，此時會建立 3 乘 3 的成對比較矩陣，所以階數(n)即為 3 且對應到的 RI 值為 0.58。

表 3：R.I.隨機指標表

n	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
RI	0.00	0.00	0.58	0.90	1.12	1.24	1.32	1.41	1.45	1.49	1.51	1.48	1.56	1.57	1.59

四、計算功能性元件的相對複雜性權值

此步驟包含 2 個工作活動以便量化產生考量相依性的功能性元件複雜度並加以正規化，分別是計算超級矩陣與正規化複雜度相對權重。圖 12 為超級矩陣計算過程的示意圖，首先是將步驟三的成對比較矩陣集合排列組成未加權矩陣，而正規化未加權矩陣的行向量即會產生已加權矩陣，最後將已加權矩陣自乘多次方即會收斂產生一個固定權重值。

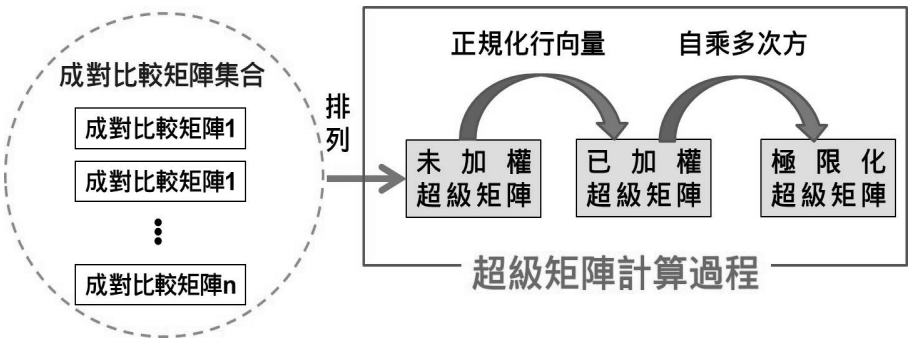

圖 12：超級矩陣的示意圖

由於原始超級矩陣中的值是來自於功能性元件的成對比較結果計算（步驟三），此成對結果（權重值）尚未考慮功能性元件的相依性，而極限化超級矩陣中的收斂值則是代表已考量相依性的相對複雜度權重值，所以可用來代表與解釋已考量相依性之功能性元件間的複雜度關係，例如圖 13 顯示 3 個功能性元件的相對複雜度關係，如果以功能性元件 3 為比較基準進行正規化，我們就可以瞭解功能性元件 1 是功能性 3 的 3.198 倍複雜，而功能性元件 2 是功能性元件 3 的 1.921 倍複雜，因此我們將此收斂值命名為相對複雜性權重值，圖 13 為功能性元件複雜度關係的量化表現方式。

	功能性元件相對複雜度權重值	功能性元件基準化複雜度權重值
功能性元件1	**0.185**	**3.198**
功能性元件2	**0.111**	**1.921**
功能性元件3	**0.058**	**1.000**

圖 13：功能性元件複雜度關係的量化表現方式

五、計算敏捷功能點數

傳統功能點分析是先取得功能性元件的複雜度等級個數，再將其乘以對應之複雜度等級權重以便產生該功能性元件的功能點數，例如系統包含 2 個低複雜度等級的 ILF、1 個高複雜度等級的 ILF 以及 1 個低複雜等級的 EI，參照圖 4 的計算方式，ILF 有 29（2*7+1*15）個功能點，而 EI 有 3 個功能點（1*3）。參考功能點分析的設計邏輯，本研究之相對複雜性權重值代表的是功能性元件間的相對複雜度倍數關係，而歷史基準點數則是代表功能性元件的歷史平均功能點數（即最有可能出現的功能點數規模大小），所以我們定義此兩者的乘積為該功能性元件的敏捷功能點數。

歷史基準點數可以透過具有代表性之軟體專案資料庫（例如 ISBSG）或是組織內部可靠的度量資料來取得。而表 4 為目前本計算模式所內建提供的歷史基準點數，其資料來源為 3 個具有高可靠性的功能點度量個案（Garmus & Herron 2001; IFPUG 2005; Bundschuh & Dekkers 2008），此外，受限於歷史個案文獻搜集彙整的困難性（需考量 FPA 的度量版本、歷史個案度量透明度等因素），因此本研究所能夠提供「可通用所有系統的歷史基準點數表」，而沒有針對不同的軟體系統類型（例如 ERP 與 CRM 等）提供「客製化歷史基準點數表」，因此未來如果能夠針對不同的軟體系統類型提供其專屬的歷史基準點數表，本度量計算模式的精確性將可以有效提昇。最後，我們亦需強調「歷史基準點數表」的可靠性亦可由累積及持續更新組織歷史專案度量資料來加以確保。

表 4：功能性元件的歷史基準點數表

功能性元件	內部邏輯檔案	外部介面檔案	外部輸入	外部輸出	外部查詢
歷史平均功能點數	7	5	4	4	3

肆、發票處理系統的敏捷功能點度量

「Fast Cars」(Parthasarathy 2007)是一間具有 20 年以上的汽車代理銷售經驗公司，該公司除了提供汽車銷售之商業行為外，也提供汽車售後服務、零組件銷售與汽車貸款等服務，為了使繁複的商業處理流程可以更具有效率，該公司擬開發一套自動化發票處理系統以有效支援其營運管理所需，圖 14 顯示發票處理系統在不同工作流程中所扮演的角色，而此系統內之功能模組的互動行為如圖 15 所示。

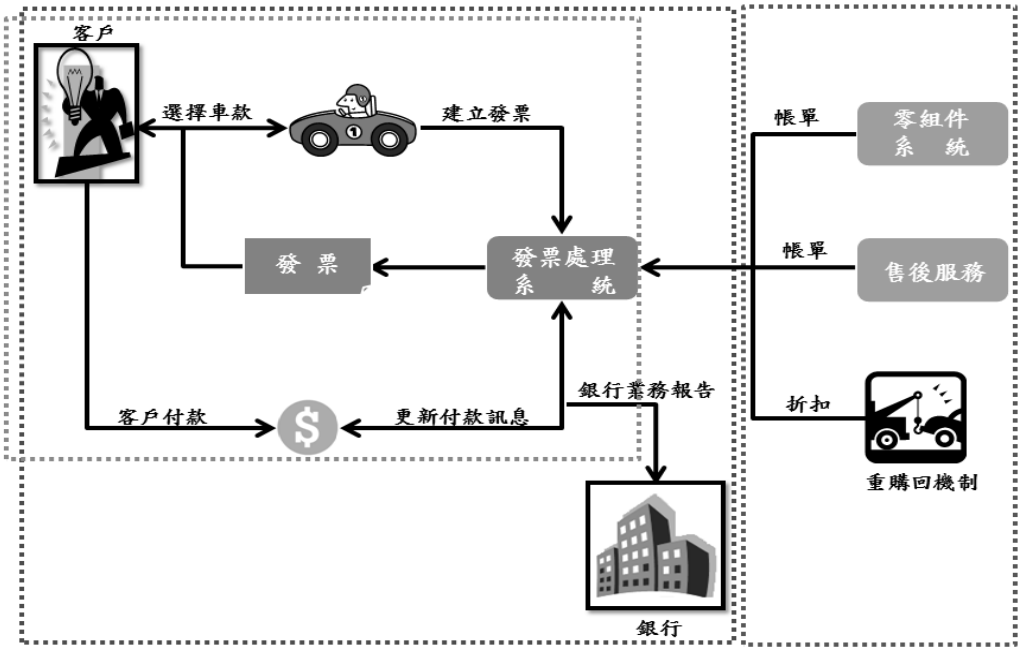

圖 14：發票處理系統相關之工作流程

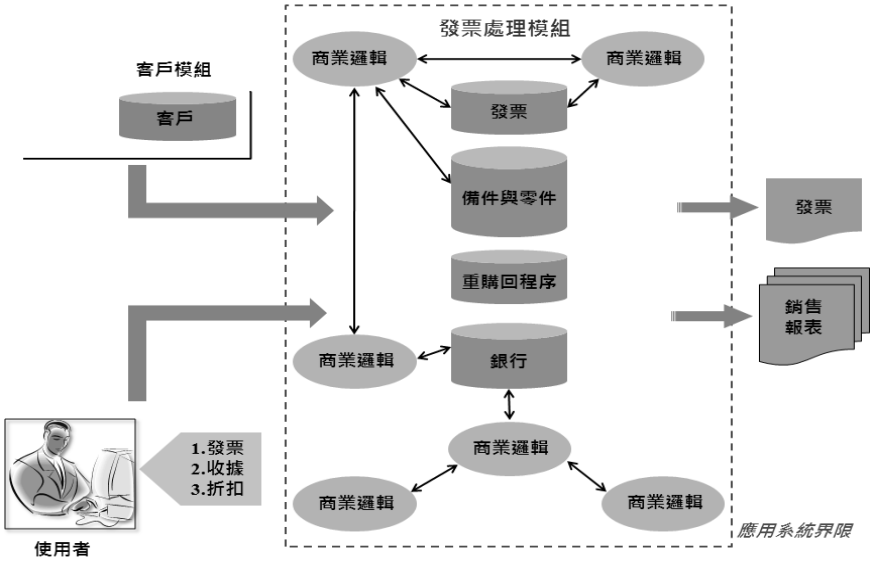

圖 15：發票處理系統內之功能模組互動行為

一、建立群組網路架構

依據敏捷功能點計算模式的第一步驟（建立群組層級架構），如圖 10 所示，我們可以將發票處理系統區分成 3 個群組架構，第一層是「目標」：描述我們欲度量發票處理系統的功能性規模大小；第二層是「準則」：描述此發票處理系統的功能性規模大小共分為 5 個功能性元件（ILF、EIF、EI、EO 與 EQ）；第三層是「次準則」：描述此發票處理系統在每一個功能性元件下的功能性需求有哪些，表 5 舉例說明「客戶資料」功能性需求被視為外部介面檔案的原因是因為其滿足了 4 條外部介面檔案的自然語言判斷規則，以此類推，將發票處理系統的所有功能性需求逐一判斷該被視為哪一種功能性元件。

表 5：「客戶資料」之外部介面檔案的判斷結果

判斷規則	判斷結果與說明
一組邏輯上相關且使用者可識別的資料或控制訊息。	是（使用者可識別的資料）
一組應用系統所參考的外部系統資料。	是（發票系統所參考之外部資料）
並非由所參考的應用系統來維護。	是（並非由發票系統所維護）
由其他應用系統所維護的 ILF。	是（由「客戶應用系統」所維護）

將上述分類結果以軟體 Super Decisions（Satty 2003）加以建構即可產生如圖

16 的群組架構，由圖中可以發現此發票處理系統共有 3 個功能性需求被分類為內部邏輯檔案、1 個功能性需求被分類為外部介面檔案、13 個功能性需求被分類為外部輸入、5 個功能性需求被分類為外部輸出以及 5 個功能性需求被分類為外部查詢。

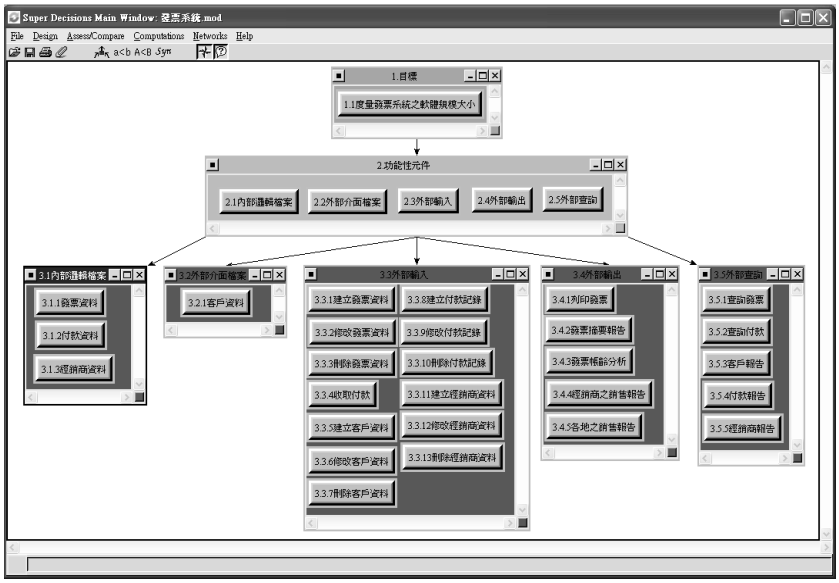

圖 16：發票處理系統之群組架構

二、定義功能性元件相依性

此步驟是根據上一個步驟所建立的發票處理系統群組架構來建立功能性元件間之相依性關係，其中元件相依性之定義可分成同一群組內之功能需求間的相依性以及不同群組間之功能需求間的相依性。系統相依性的判斷可以經由單人（專案經理）或多人（專家訪談、焦點團體、群體決策技術與座談會等）的方式來決定。而本研究的發票處理系統案例的相依性判斷則是以指導教授與研究生反覆討論的方式來產生，這是因為發票處理系統案例是用來說明呈現本計算模式的「應用可行性」而非精確度比較。圖 17 顯示此 27 個功能性元件間的相依性關係，由圖中亦可以清楚發現並不是所有的功能性元件都會相互影響，此外，建立/修改/刪除發票資料與其他功能性元件具有較多的相依性關係，亦可推論此三個功能需求在發票系統中具有較大的影響力，

	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
1.發票資料		√	√	√	√	√	√	√										√	√	√	√	√	√	√			
2.付款資料	√		√	√								√	√	√				√	√	√	√		√		√		
3.經銷商資料	√	√		√	√	√	√	√							√	√	√	√					√	√			√
4.客戶資料	√	√	√		√	√	√	√	√	√	√							√	√	√	√	√	√	√	√		
5.建立發票資料	√		√	√		√	√	√	√	√	√	√	√	√	√	√	√	√	√	√	√	√	√				
6.修改發票資料	√		√	√	√		√	√	√	√	√	√	√	√	√	√	√	√	√	√	√	√	√				
7.刪除發票資料	√		√	√	√	√		√	√	√	√	√	√	√	√	√	√	√	√	√	√	√	√				
8.收取付款	√		√	√	√	√	√		√	√	√	√	√	√	√	√	√										
9.建立客戶資料				√	√	√	√	√		√	√	√	√	√	√	√	√	√	√	√	√	√	√	√	√		
10.修改客戶資料				√	√	√	√	√	√		√	√	√	√	√	√	√	√	√	√	√	√	√	√	√		
11.刪除客戶資料				√	√	√	√	√	√			√	√	√	√	√	√	√	√	√	√	√	√	√	√		
12.建立付款資料		√			√	√	√	√	√	√	√		√	√	√	√	√	√	√	√	√	√		√		√	
13.修改付款資料		√			√	√	√	√	√	√	√	√		√	√	√	√	√	√	√	√	√		√		√	
14.刪除付款資料		√			√	√	√	√	√	√	√	√	√		√	√	√	√	√	√	√	√		√		√	
15.建立經銷商資料			√		√	√	√	√	√	√	√	√	√	√		√	√	√			√		√	√			√
16.修改經銷商資料			√		√	√	√	√	√	√	√	√	√	√	√		√	√			√		√	√			√
17.刪除經銷商資料			√		√	√	√	√	√	√	√	√	√	√	√	√		√			√		√	√			√
18.列印發票	√	√	√	√	√	√	√		√	√	√	√	√	√	√	√	√		√	√	√	√	√				
19.發票摘要報告	√	√		√	√	√	√		√	√	√	√	√	√				√		√	√	√					
20.發票帳齡分析	√	√		√	√	√	√		√	√	√	√	√	√				√	√		√	√					
21.經銷商之銷售報告	√	√		√	√	√	√		√	√	√	√	√	√	√	√	√	√	√	√		√					
22.各地之銷售報告	√	√		√	√	√	√		√	√	√	√	√	√				√	√	√	√						
23.查詢發票	√		√	√	√	√	√		√	√	√				√	√	√	√						√	√	√	√
24.查詢付款	√	√	√	√	√	√	√		√	√	√	√	√	√	√	√	√						√		√	√	√
25.客戶報告				√					√	√	√												√	√		√	√
26.付款報告		√										√	√	√									√	√	√		√
27.經銷商報告			√												√	√	√						√	√	√	√	

√: 代表「建立發票資料」與「查詢付款」存在功能相依性

圖 17：發票處理系統的功能相依性

一旦確定發票處理系統的功能相依性無誤後，我們就可以在 Super Decision 軟體上依據圖 17 的內容來「手動建立」相依性關連，如圖 18 所示，因此圖 18 是顯示已考慮功能相依性的發票處理系統群組架構，而 Super Decision 軟體會根據所建立的功能相依性關係來進行超級矩陣計算以產生考慮功能相依性的量化相對權重，圖 18 箭頭代表功能性元件間之相依關係，若群組間是雙向箭頭，則代表兩功能性元件群組間存在相互關係；若同一個功能性元件群組具有迴圈箭頭，則代表同一功能性元件群組內不同功能性需求間存在相互關係。

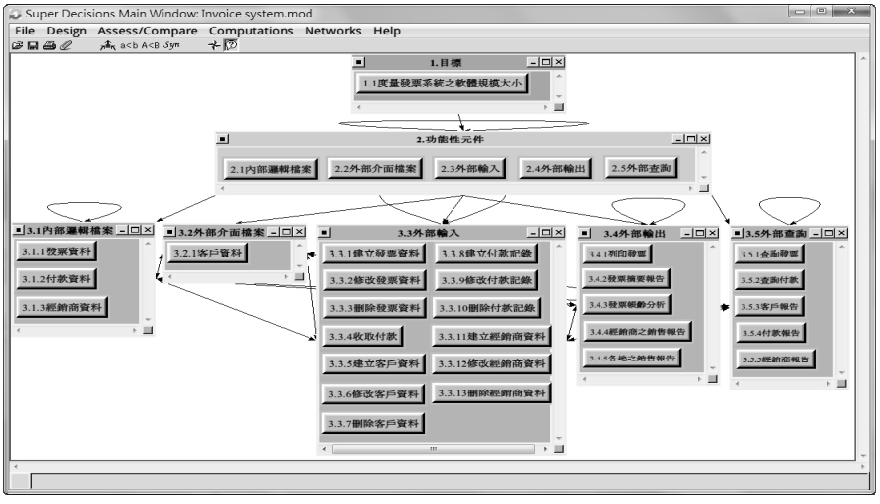

圖 18：發票處理系統之功能相依性關係

三、評估功能性元件相對複雜性

當發票管理系統的網路架構被有效建構後，就必須透過功能性元件間之成對比較計算以進一步取得各功能性元件的相對重要性。同一階層的功能需求會兩兩比較，所以第 2 層如果有 3 個功能需求，比較次數計算公式為 n(n-1)/2，即 3 次。此外，當同一階層的 2 個功能需求有相依關係時，也需要進行成對比較。例如內部邏輯檔案功能性元件有 3 個元素：「發票資料」、「付款資料」與「經銷商資料」，此 3 個元素會兩兩比較，而由於「發票資料」與外部介面檔案功能性元件的「客戶資料」及外部輸入功能性元件的「經銷商資料」、「經銷商資料」與「經銷商資料」有相依關係，所以這 5 個功能需求也會成對比較。最後，不同階層的功能需求並不會進行兩兩比較，例如內部邏輯檔案功能性元件並不會與外部介面檔案功能性元件的「經銷商資料」功能需求進行成對比較。圖 19 顯示目標群組下的功能成對比較評估方式。

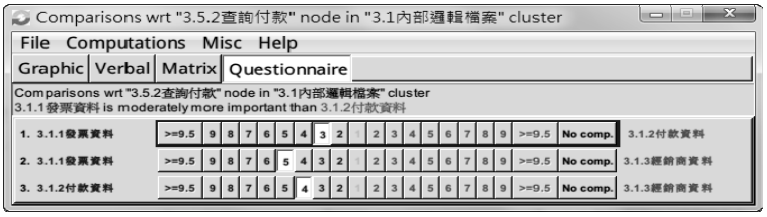

圖 19：發票處理系統的功能成對比較評估

利用一致性檢定確認每一個成對比較的評估結果是否發生前後不一致的狀況，如圖 20 所示，當所有成對比較矩陣均通過一致性檢定則就可將所有成對比較結果整合成一個超級矩陣。

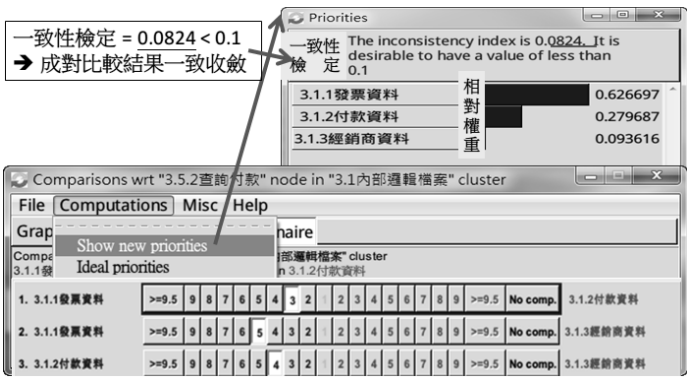

圖 20：發票處理系統的一致性檢定

四、計算功能性元件的相對複雜性權值

當所有成對比較矩陣均通過一致性檢定則就可將所有功能性元件成對比較結果整合排列以形成一個 32×32 大小的超級矩陣，即未加權超級矩陣，透過我們在敏捷功能點計算模式步驟四的示意圖，我們就可以將其轉換為已加權超級矩陣與極限化超級矩陣，而極限化超級矩陣中的收斂值則代表已考量相依性的相對複雜度權重值，所以可用來代表與解釋已考量相依性之功能性元件間的複雜度關係，附錄 A 顯示極限化超級矩陣的結果。

由於極限化超級矩陣收斂值即為已考慮功能相依性的相對複雜性權重，最後我們以各個功能性元件中之最小權重值進行標準化以產生相對倍數關係，此倍數關係即可以做為與功能性元件之歷史基準點數相乘的基礎。例如此發票系統共有 3 個內部邏輯檔案，分別是發票資料（極限化權值為 0.185）、付款資料（極限化權值為 0.111）與經銷商資料（極限化權值為 0.058），由於經銷商資料在此群組中具有最小權重值，所以用其做為求得標準化權值的依據，因此這 3 個內部邏輯檔案的標準化權值分別為 3.198、1.921 與 1.000，這意謂其餘 2 個內部邏輯檔案（發票資料與付款資料）與經銷商資料的倍數（複雜）關係分別為 3.198 與 1.921。

五、敏捷功能點數

表 6 顯示此發票處理系統具有 296.578 個敏捷功能點，從中亦可以觀察「發票處理系統」中的各個功能性元件所佔之比重關係，其複雜性大小由高至低分別為：外部輸入（142.602 點, 48%）、外部查詢（58.366 點, 20%）、內部邏輯檔案（42.832 點, 14%）、外部輸出（47.778 點, 16%）與外部介面檔案（5 點, 2%）。更進一步分析可以發現：在內部邏輯檔案群組中，最複雜的是發票資料，點數貢獻度為 52.3%、在外部輸入群組中，最重要的是收取付款，點數貢獻度為 15.1%、在外部輸出群組中，最重要的是列印發票，點數貢獻度為 47.7%、在外部查詢群組中，最重要的是查詢發票，點數貢獻度為 36.9%、此系統只有一個外部介面檔案（客戶資料），因此其點數貢獻度為 100%，專案管理者可以根據上述之分析結果在開發早期進行資源配置，以提昇「發票處理系統」成功開發之機率。

表 6：發票處理系統的敏捷功能點數

功能性元件	功能需求	功能性元件極限化權值	功能性元件基準化權值	功能性元件歷史基準點	敏捷功能點數
內部邏輯檔案	1.1 發票資料	0.185	3.198	7	22.385
	1.2 付款資料	0.111	1.921		13.447
	1.3 經銷商資料	0.058	1.000		7.000
外部介面檔案	2.1 客戶資料	0.248	1.000	5	5.000
外部輸入	3.1 建立發票資料	0.012	2.579	4	10.314
	3.2 修改發票資料	0.017	3.674		14.698
	3.3 刪除發票資料	0.012	2.605		10.419
	3.4 收取付款	0.025	5.375		21.499
	3.5 建立客戶資料	0.017	3.758		15.031
	3.6 修改客戶資料	0.025	5.355		21.418
	3.7 刪除客戶資料	0.017	3.758		15.031
	3.8 建立付款資料	0.006	1.377		5.506
	3.9 修改付款資料	0.012	2.537		10.146
	3.10 刪除付款資料	0.006	1.380		5.520
	3.11 建立經銷商資料	0.005	1.000		4.000
	3.12 修改經銷商資料	0.006	1.255		5.020
	3.13 刪除經銷商資料	0.005	1.000		4.000
外部輸出	4.1 列印發票	0.068	5.693	4	22.771
	4.2 發票摘要報告	0.026	2.193		8.773
	4.3 發票帳齡分析	0.024	2.012		8.049
	4.4 經銷商之銷售報告	0.012	1.046		4.185
	4.5 各地之銷售報告	0.012	1.000		4.000
外部查詢	5.1 查詢發票	0.033	7.185	3	21.554
	5.2 查詢付款	0.026	5.674		17.023
	5.3 客戶報告	0.021	4.484		13.453
	5.4 付款報告	0.005	1.112		3.336
	5.5 經銷商報告	0.005	1.000		3.000

六、分析與討論

有 2 個顯而易見的特色（優點）可以藉由檢視發票處理系統案例的量化過程來加以確認：(1)系統功能相依性觀點可以透過 ANP 的整合來實現以便完整反映系統功能全貌；(2)排除使用複雜度等級權重的自然規則以便克服權重值久未更新

的問題。此外，隨著專案生命週期逐步擴展與明朗透明，度量者可以將本計算模式的產出資訊（例如已經識別的內部邏輯檔案與外部介面檔案的個數）直接回饋至正式的功能點分析中，因此不但可以縮短計算功能點所需的度量時間，也不致使得度量者的工作量與學習負擔加重。

表 7 彙整比較分析功能點分析與敏捷功能點計算模式（本研究）的特色差異與定位，最明顯的差異在於是否能反映真實世界中功能性元件為相互依賴協同合作的精神，此外，雖然本研究不使用 FPA 的複雜度矩陣來判斷功能性元件的複雜度，即不使用 FPA 自然語言判斷法則來識別 DET、RET 與 FTR，但是由於複雜度權重會因每次的個案評估而有所變動，因此這將導致無法將本計算模式的結果進行跨組織比較。再者，功能點分析已經有許多實證研究在進行多方探討，而本計算模式則有待學者投入心力延伸研究。

表 7：功能點分析與敏捷功能點計算模式的比較分析

度量方式	功能點分析	敏捷功能點計算模式
功能性元件的相關性假設	功能性元件獨立	功能性元件相依
識別 DET、RET 與 FTR	需要充份識別	不用識別
FPA 自然語言規則使用數量	43	24
功能複雜度權重	固定且久未更新	隨個案而定，非固定不變
跨組織比較	可以	不可以
度量人員	最好取得 CFPS 證照	有經驗的專家與管理者
實證研究	很多	有待學者投入

我們無法在本研究中以多案例的客觀精確度比較分析資訊來充份佐證本計算模式的優點，這是因為比較分析不同度量方式(FPA 與本研究產出)的優劣需要透過大量案例累積才可以有效證明。當組織持續使用本計算模式且累積多筆歷史專案資訊，即可以透過量化研究方法（線性迴歸或實驗設計等）來加以分析，例如利用 FPA 與本計算模式來度量每一筆軟體專案且個別搜集其工作量與程式碼資訊，當所搜集到的歷史專案數量大於一個有意義的統計數量（n=30 以上）時，才有可能呈現有意義的精確度比較分析資訊。

最後，本計算模式的發展構想並不是想要「取代」功能點分析的地位，而是要提供度量者另一個量化軟體功能性規模的選擇，基於不同的度量方法有不同的應用時機，所以使用本計算模式的最好時機為「開發早期且 DET、RET 與 FTR 難以有效區別分辨的情況」。換句話說，如果開發早期就可以充份瞭解 DET、RET 與 FTR 資訊，使用有許多實證研究支持的 FPA 來量化軟體功能性規模可能

仍然是現今較佳的選擇，因此我們建議可於開發前期使用本計算模式求得度量粗估值而在開發中期則使用 FPA 求得度量精細值。

伍、結論

本研究整合 ANP 與 FPA 以發展敏捷功能點計算模式，我們必須要強調本研究並不是要取代功能點分析來量化軟體系統功能性規模，而是希望可以提出一個適用於開發早期（尤其是 DET、FTR 與 RET 之數量無法加以有效界定的狀況下）且可以有效快速取得近似於真實功能點數的方法，特別的是，本計算模式額外考量功能性元件（ILF、EIF、EI、EO 與 EQ）的相依性以完整反映系統功能全貌，致使度量結果能夠更真實反映軟體專案之功能性規模。當系統開發資訊伴隨著專案逐步進展而明朗清楚後，使用者亦容易將本研究成果再利用（回饋）於正式的功能點分析計算以做為計算參考依據，這是因為本計算模式並沒有自行提出新的基本功能性元件，這個優點相對於其它技術〔例如 E&Q（Santillo et al. 2005）與 KISS（Pekka 2006）〕將可以更有效提昇度量人員的使用意願與降低學習負擔。

基於本研究的結果，有 3 個未來研究方向可以進行延伸探討：(1)本研究是透過『發票處理系統』的功能性規模度量來說明本計算模式的應用方式，未來，研究者可以引導一個實證研究並大量搜集專案相關資訊以便量化分析比較 FPA 與本計算模式的效果差異（例如精確度與計算時間等）；(2)針對功能需求間的成對比較量化與系統相依性關係塑模發展實務指引，以便降低使用本計算模式的學習門檻與確保一致性；(3)針對不同資訊系統建置各自獨立的歷史基準點數表，以便使本計算模式的度量結果可以更加精確。

誌謝

首先感謝兩位匿名論文審查委員惠賜的寶貴意見，使得本研究的內容能夠更加充實並更臻完備。此外，作者也要感謝國科會的研究經費補助（NSC 100-3114-C-147-001-ES）。

參考文獻

胡衣臨、黃世禎（民 93），『軟體功能點數計算精確度的影響因素與提昇之研究』，未出版碩士論文，國立臺灣科技大學資訊管理研究所，台北市。

祝淑蘭、鄭炳強、葉道明（2006），『迴歸分析法和功能點分析於軟體規模評估上的比較』，*第二屆資訊管理學術暨專案管理實務研討會論文集*，開南管理學院主辦。

彭智賢、黃怡誠（民 94），『基於 XML 之 ER-DFD 自動化功能點分析系統之設計與實作』，未出版碩士論文，國立臺灣師範大學資訊教育學系，台北市。

黃明祥、江秉翰、蕭衛鴻（2008），『軟體專案度量程序模式之建立與應用－以國內某大型鋼鐵公司之研發部門為實證對象』，*中華民國資訊管理學報*，第十五卷，第一期，頁 203-238。

劉青峻、劉文卿（民 92），『功能點分析應用於以物件導向語言開發之資訊系統的實例研究』，*網際空間：科技、犯罪與法律社會學術研究暨實務研討會論文集*，國立台灣師範大學主辦。

Albrecht, A.J. (1979), 'Measuring application development productivity', *Proceedings of IBM Application Development Symposium* (October 1979), pp. 83-92

Albrecht, A.J. and Gaffney, J.E. (1983), 'Software function, source lines of code, and development effort prediction: a software science validation', *IEEE Transactions on Software Engineering*, Vol. 9, No. 6, pp. 639-648.

Bundschuh, M. and Dekkers, C. (2008), *The IT Measurement Compendium: Estimating and Benchmarking Success with Functional Size Measurement*, Springer Press.

Christopher, J.L. (1999), 'An empirical study of the correlation between function point elements', *Proceedings of the Sixth International Software Metrics Symposium*.

Fenton, N.E. and Pfleeger, S.L. (1996), *Software Metrics: A Rigorous and Practical Approach*, International. Thomson Computer Press.

Garmus, D. and Herron, D. (2001), *Function Point Analysis — Measurement Practices for Successful Software Project*, Addison-Wesley Professional Press.

Halstead, M. (1977), *Elements of Software Science*, New York Elsevier, North-Holland.

IFPUG (2005), 'Function point counting practices: case studies release 2.0'.

IFPUG (2010), 'Function point counting practices manual release 4.3.1'.

ISO (2002), 'ISO/IEC 20968, software engineering — Mk II function point analysis – counting practices manual'.

ISO (2003), 'ISO/IEC 19761, software engineering — COSMIC-FFP — a functional

ISO (2003), 'ISO/IEC 19761, software engineering — COSMIC-FFP — a functional size measurement method'.

ISO (2005), 'ISO/IEC 24570, software engineering — NESMA functional size measurement method version 2.1 — definitions and counting guidelines for the application of function point analysis'.

ISO (2009), 'ISO/IEC 20926, Software engineering — Mk II function point analysis – counting practices manual'.

ISO (2010), 'ISO/IEC 29881, information technology — software and systems engineering — FiSMA 1.1 functional size measurement method'.

Jeffery, R. and Stathis, J. (1996), 'Function point sizing: structure, validity and applicability', *Empirical Software Engineering*, Vol. 1, No. 1, pp. 11-30.

Kitchenham, B. and Känsälä, K. (1993), *Inter-item Correlation among Function Points*, IEEE Computer Society Press, pp. 477-480.

Koh, T.W., Selamat, M.H. and Ghani, A.A.A. (2008), 'Exponential effort estimation model using unadjusted function points', *Information Technology Journal*, Vol. 7, No. 6, pp. 830-839.

Lamma, E., Mello, P. and Riguzzi, F. (2004), 'A system for measuring function points from an ER-DFD specification', *Computer Journal*, Vol. 47, No. 3, pp. 358-372.

Lee, J., Kang, S. and Kim, C.K. (2009), 'Software architecture evaluation methods based on cost benefit analysis and quantitative decision making', *Empirical Software Engineering*, Vol. 14, No. 2, pp. 453-475.

Low, C.G. and Jeffery, R.D. (1990), 'Function points in the estimation and evaluation of the software process', *IEEE Transactions on Software Engineering*, Vol. 16, No. 1, pp. 64-71.

Meade, L.M. and Presley, A. (2002), 'R&D project selection using the analytic network process', *IEEE Transactions on Engineering Management*, Vol. 49, No. 1, pp. 59-66.

Niemira, M.P. and Saaty, T.L. (2004), 'An analytic network process model for financial-crisis forecasting', *International journal of Forecasting*, Vol. 20, No. 3, pp. 573-587.

Parthasarathy, M.A. (2007), *Practical Software Estimation: Function Point Methods for Insourced and Outsourced Projects*, Addison-Wesley Professional Press.

Pekka, F. (2006), 'Faster and more accurate functional size measurement by KISS – keeping it simple', IFPUG Field Support Services.

Saaty, T.L. (1996), *Decision Marking With Dependence and Feedback — The Analytic Network Process*, RWS Publication.

Saaty, T.L. (2003), 'Decision Marking In Complex Environments', Super Decisions.

Santillo, L., Conte, M. and Meli, R.E. (2005), 'Quick function point: sizing more with less', 11th IEEE International Software Metrics Symposium.

Symons, C.R. (1988), 'Function point analysis: difficulties and improvements', *IEEE Transactions on Software Engineering*, Vol. 14, No. 1, pp. 2-11.

Turetken, O., Ozcan, T.O., Ozkan, B. and Demirors, O. (2008), 'The impact of individual assumptions on functional size measurement', Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics).

Wei, X., Luiz, F.C., Danny, H. and Faheem, A. (2008), 'A new calibration for Function Point complexity weights', *Information and Software Technology*, Vol. 50, No. 7-8, pp. 670-683.

附錄 A：發票系統之極限化超級矩陣

		1.1	2.1	2.2	2.3	2.4	2.5	3.1.1	3.1.2	3.1.3	3.2.1	3.3.1	3.3.2	3.3.3	3.3.4	3.3.5	3.3.6	3.3.7	3.3.8	3.3.9	3.3.10	3.3.11	3.3.12	3.3.13	3.4.1	3.4.2	3.4.3	3.4.4	3.4.5	3.5.1	3.5.2	3.5.3	3.5.4	3.5.5
Goal	1.1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
BFC	2.1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	2.2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	2.3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	2.4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	2.5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
ILF	3.1.1	0.185	0.185	0.185	0.185	0.185	0.185	0.185	0.185	0.185	0.185	0.185	0.185	0.185	0.185	0.185	0.185	0.185	0.185	0.185	0.185	0.185	0.185	0.185	0.185	0.185	0.185	0.185	0.185	0.185	0.185	0.185	0.185
	3.1.2	0.111	0.111	0.111	0.111	0.111	0.111	0.111	0.111	0.111	0.111	0.111	0.111	0.111	0.111	0.111	0.111	0.111	0.111	0.111	0.111	0.111	0.111	0.111	0.111	0.111	0.111	0.111	0.111	0.111	0.111	0.111	0.111
	3.1.3	0.058	0.058	0.058	0.058	0.058	0.058	0.058	0.058	0.058	0.058	0.058	0.058	0.058	0.058	0.058	0.058	0.058	0.058	0.058	0.058	0.058	0.058	0.058	0.058	0.058	0.058	0.058	0.058	0.058	0.058	0.058	0.058
EIF	3.2.1	0.248	0.248	0.248	0.248	0.248	0.248	0.248	0.248	0.248	0.248	0.248	0.248	0.248	0.248	0.248	0.248	0.248	0.248	0.248	0.248	0.248	0.248	0.248	0.248	0.248	0.248	0.248	0.248	0.248	0.248	0.248	0.248
EI	3.3.1	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012
	3.3.2	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017
	3.3.3	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012
	3.3.4	0.025	0.025	0.025	0.025	0.025	0.025	0.025	0.025	0.025	0.025	0.025	0.025	0.025	0.025	0.025	0.025	0.025	0.025	0.025	0.025	0.025	0.025	0.025	0.025	0.025	0.025	0.025	0.025	0.025	0.025	0.025	0.025
	3.3.5	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017
	3.3.6	0.025	0.025	0.025	0.025	0.025	0.025	0.025	0.025	0.025	0.025	0.025	0.025	0.025	0.025	0.025	0.025	0.025	0.025	0.025	0.025	0.025	0.025	0.025	0.025	0.025	0.025	0.025	0.025	0.025	0.025	0.025	0.025
	3.3.7	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017
	3.3.8	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006
	3.3.9	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012
	3.3.10	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006
	3.3.11	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005
	3.3.12	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006
	3.3.13	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005
EO	3.4.1	0.068	0.068	0.068	0.068	0.068	0.068	0.068	0.068	0.068	0.068	0.068	0.068	0.068	0.068	0.068	0.068	0.068	0.068	0.068	0.068	0.068	0.068	0.068	0.068	0.068	0.068	0.068	0.068	0.068	0.068	0.068	0.068
	3.4.2	0.026	0.026	0.026	0.026	0.026	0.026	0.026	0.026	0.026	0.026	0.026	0.026	0.026	0.026	0.026	0.026	0.026	0.026	0.026	0.026	0.026	0.026	0.026	0.026	0.026	0.026	0.026	0.026	0.026	0.026	0.026	0.026
	3.4.3	0.024	0.024	0.024	0.024	0.024	0.024	0.024	0.024	0.024	0.024	0.024	0.024	0.024	0.024	0.024	0.024	0.024	0.024	0.024	0.024	0.024	0.024	0.024	0.024	0.024	0.024	0.024	0.024	0.024	0.024	0.024	0.024
	3.4.4	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012
	3.4.5	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012	0.012
EQ	3.5.1	0.033	0.033	0.033	0.033	0.033	0.033	0.033	0.033	0.033	0.033	0.033	0.033	0.033	0.033	0.033	0.033	0.033	0.033	0.033	0.033	0.033	0.033	0.033	0.033	0.033	0.033	0.033	0.033	0.033	0.033	0.033	0.033
	3.5.2	0.026	0.026	0.026	0.026	0.026	0.026	0.026	0.026	0.026	0.026	0.026	0.026	0.026	0.026	0.026	0.026	0.026	0.026	0.026	0.026	0.026	0.026	0.026	0.026	0.026	0.026	0.026	0.026	0.026	0.026	0.026	0.026
	3.5.3	0.021	0.021	0.021	0.021	0.021	0.021	0.021	0.021	0.021	0.021	0.021	0.021	0.021	0.021	0.021	0.021	0.021	0.021	0.021	0.021	0.021	0.021	0.021	0.021	0.021	0.021	0.021	0.021	0.021	0.021	0.021	0.021
	3.5.4	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005
	3.5.5	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005