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

黃明祥
屏東科技大學資訊管理系

江秉翰
屏東科技大學資訊管理系

蕭衛鴻
屏東科技大學資訊管理系

摘要

軟體專案度量是軟體專案管理的一項重要研究議題。軟體專案度量不僅是成本與時程估計的重要資訊來源，同時，它亦提供專案管理者有關專案狀態評量與預測之決策資訊。近年來，許多軟體專案已經移轉至網際網路環境下進行專案開發工作，若軟體專案度量活動仍然採用人工作業方式則相當困難且作業品質不佳。有鑑於此，本研究乃探討在網際網路環境下之軟體專案度量議題，其中包括成本、時程、品質與生產力等度量之重要議題，並以COCOMO II 估計模式、ISO/IEC 9126-1品質模式與IEEE Std. 1045生產力度量標準為基礎，結合差異分析法與加權評分法等提出一個軟體專案度量程序模式，據以發展一套Web-based的軟體專案度量資訊系統協助軟體專案人員進行軟體專案度量工作，同時，本系統尚結合智慧型代理人協助進行專案狀態評量、預測以及相關決策方案分析建議，其中包含狀態評量、決策分析與訊息通知等三種代理人程式之功能，可以有效的解決在網際網路環境下進行大型軟體專案度量的重要問題，並且提供一些因應之建議性方案供專案管理者之參考。本研究係以國內一家大型鋼鐵公司之研發部門所進行的大型軟體專案作為實證研究的對象，據以了解軟體專案度量資訊系統在實務應用方面的可行性與實用性。根據本研究實證分析結果，本系統結合代理人程式與知識庫系統之運用，能夠達成專案狀態評量、決策分析與訊息通知等作業自動化之功能，以及減少個人及主觀因素所造成的誤差，提升度量作業的品質與正確性；專案管理者可以在專案的開發過程中透過本系統即時監控專案，確實掌握專案現況，並且根據系統所提供的專案狀態資訊與決策建議，利於專案後續的改善活動；此外，根據已發生的實際資訊預測專案可能之變化，以降低專案開發風險；專案的重要資訊得以保存，可作為日後相關專案開發之重要參考。

關鍵字：軟體專案度量、軟體專案管理、智慧型代理人

Developments and Applications of a Software Project Measurement Process Model: A Case Study of the R&D Department of a Large Scale Steel Company in Taiwan

Ming-Shang Huang
Department of Information Management, National Pingtung University of Science and Technology

Ping-Han Chiang
Department of Information Management, National Pingtung University of Science and Technology

Wei-Hung Hsiao
Department of Information Management, National Pingtung University of Science and Technology

Abstract

Software project measurement is an important research issue in software project management. Metrics of software project are major source for cost and schedule estimations and there also provides decision-making information about status assessment and forecasting for a project manager. Recently, many software projects have been implemented gradually on the web. Therefore, it is difficult to obtain the software project metrics through the labor work. Based on above descriptions, this research investigates key issues of software project metrics on the web including four kinds of metrics for cost, schedule, quality and productivity. In this study, we use COCOMO Ⅱ, ISO/IEC 9126-1 and IEEE Std. 1045 as a foundation combing with the variance analysis and weighted ranking method to construct a software project measurement process model. Moreover, a Web-based software project metrics information system is developed. This system applies status assessment agent, decision making agent and notification agent to assist the tasks such as assessment of project progress, forecasting and decision-making to resolve the issues of large scale software project metrics on the web. To demonstrate the feasibility and applicability of

the developed system, a case study of the R&D department of a large scale steel company in Taiwan is conducted. Research findings show that an intelligent agent-based software project metrics information system not only effectively support the activities of software project measurement in automation and it also provides detailed and accurate decision-making information about the status assessment and forecasting for software project management. Thus, risks of software project development can be reduced dramatically. Furthermore, some important information of software projects is accumulated and the improvements of effectiveness and quality of software project management are expected.

Key words: Software project metrics, software project management, intelligent agent

壹、緒論

近年來，由於軟體專案的開發活動有逐漸朝向大型化與分散化趨勢，許多大型軟體專案是以多個子專案分散在不同地點協同進行專案的開發活動，因此專案在開始執行後，專案管理者不易獲知軟體專案的運作現況，更遑論掌控目前專案的進行，建構與維護分散式的中大型軟體專案變成一項複雜與困難的工作(Paul et al. 1999)。根據研究調查指出，美國每一年投入在軟體專案的金額約有二百七十五億左右，但是僅有百分之二十六的軟體專案是完全成功的，而有百分之七十的軟體開發與資訊科技專案面臨失敗、取消或無法符合高階主管與使用者的預期(Jones 1995; Liu et al. 2003)。其中，專案失敗與取消的主要因素在於專案開發過程中，專案相關人員無法獲知與監控進行中的專案真實情況(Simmons & Wu 2000)，無法及時掌握問題發生的先機，造成專案執行難以管控。為了改善此一情況，透過軟體專案度量對專案做持續的評估與監控，可確實掌握專案進行的實際情況(Paul et al. 1999)，進而對軟體專案的績效產生關鍵性影響。一般而言，為了有效掌控專案開發活動，在軟體專案進行中，進行整體性度量活動可以獲取完整的專案動態資訊，對於軟體開發的工作是具有其必要性。另一方面，專案度量資訊對專案的預算使用、開發進度、生產效率與產品品質對專案具有關鍵性的影響(Chee et al. 1998; 黃世禎 2003)。

在軟體專案開發過程中，進行度量活動之主要目的如下(林信惠等 2002)：(1)度量是具體而清晰的目標，也是一種明確的管理目標，(2)度量是持續改進的依據，將衡量結果即時回饋作為不斷改善之基礎，(3)度量可以作為模式建立及預測的基礎，透過明確且數量化的度量變數可建構複雜的管理模式，(4)度量的精密度及多樣性反應了管理水準的高低，提升度量的即時性與精確性，可以有效反應管理目的，與(5)度量是一種交換或檢驗的標準，透過明確客觀的標準及比較基準，管理者與顧客才有判斷的依據。從以上說明得知，軟體專案量測指標是以專案度量的估計值做為基準，用以控制成本、監控專案時程、評量軟體產品的品質與生產力，並應用在專案開發的流程改善工作，同時，可以累積相關資訊做為未來分析專案績效以找出成功案例做為日後專案的標竿。由此可見，軟體專案度量對於軟體開發者、專案管理者與高階管理者有相當大的用途與重要性(黃世禎 2003)。

若以傳統人工作業進行分散式大型軟體專案之度量活動，可能面臨下列問題：(1)軟體專案度量的資訊蒐集工作十分繁複，且耗費極大的人力、時間和成本，(2)缺少一致性的度量標準與指標，(3)度量資訊的品質和時效性難以確保，與(4)專案人員之間的溝通和協調工作缺乏效率。另一方面，從管理者觀點而言，尚有下列問題仍待解決：(1)如何進行分散式大型軟體專案之度量活動，以獲取完整的專案狀態資訊，與(2)度量資訊的內容多為低階屬性的資訊，如何萃取一些管理者有用的決策資訊，針對軟體專案開發與管理工作等進行改善活動，據以提升專案績效。

目前，許多探討軟體專案度量之相關研究均採用Web-based的作業平台為基礎 (Selby

et al. 1991; Selby 1992; Callahan & Ramakrishnan 1996; Liu & Viswanathan 1999; Chee et al. 1999; Liu et al. 2003; Zhang & Wang 2004) ，其原因在於Web技術具有跨平台、多人同時在不同地點使用等特性。因此，Web-based的作業平台適用在分散化專案開發工作的度量工具之開發，可以有效的解決軟體專案度量的諸多問題，例如專案度量資訊的搜集、分析與應用等問題(Chee et al. 1999)。另一方面，由於智慧型代理人具有自主性(Autonomous)、合作性(Cooperative)與調適性(Adaptability)等特性，適合用於分散式的軟體開發環境，而且能結合知識庫系統(Knowledge-based System; KBS)，賦予智慧型代理人決策的功能，能夠解決許多專案開發所面臨問題，例如專案規劃與進度的追蹤(Wu & Simmons 2000; Simmons 2003)、專案的溝通與協調(Hardeep 1997)、專案知識的維護與儲存(O'Connor & Jenkins 1999)、將專案資訊轉換為有利於專案進行的決策資訊等(黃明祥 & 盧煥文 2006)。因此，本研究擬結合智慧型代理人程式發展一套軟體專案度量資訊系統，期能解決在網際網路作業環境下進行軟體專案度量的重要問題。

本研究除了建立一個軟體專案度量程序模式外，並以此模式為基礎發展出一套軟體專案度量資訊系統，提供專案決策資訊據以有效的協助專案管理者進行大型軟體專案度量之決策分析工作，期能達成下列目的：(1)蒐集軟體專案開發過程中的重要度量資訊，(2)依據度量資訊對專案開發過程與整體現況進行評量與預測，(3)因應決策方案的評估與選擇，與(4)專案現況與因應方案的訊息主動通知等。因此，透過軟體專案度量程序模式的指引與以智慧型代理人為基礎的軟體專案度量資訊系統可以協助專案管理者解決許多複雜的溝通、協調問題與例行性的工作，使得專案管理者能夠快速獲知專案的實際狀態，有效的掌控軟體專案開發活動。

在過去的研究中，對於軟體度量議題例如成本、時程、生產力與品質等有深入探討，但是透過整合性觀點探討軟體專案度量的研究則並不多見，或是僅提出一些概念性架構，對於軟體專度量程序模式、執行步驟及後續的決策應用(Paul et al. 1999; Gopal et al. 2002; Zhang & Wang 2004) 亦缺少詳細而深入的討論與分析。有鑑於此，茲將本研究目的敘述如下：(1)以應用觀點探討重要的軟體專案度量議題，據以提供學術研究主題之參考用途，(2)以軟體專案度量理論為基礎，提出一個軟體專案度量程序模式，作為發展軟體專案度量資訊系統之參考架構，與(3)將本研究所發展的以智慧型代理人為基礎的軟體專案度量資訊系統應用於大型軟體專案度量活動，作為專案管理者與開發人員一項輔助性的決策工具，據以提升軟體專案開發與管理績效。

貳、文獻探討

一、軟體專案度量理論

度量(Metrics)是一種數量化或符號的指標，用以表示真實世界中某一個實體或抽象概念的特徵(林信惠等 2002)。根據過去度量理論的相關研究(林信惠等 2002; Fenton 1994; Kitchenham et al. 1995; Riguzzi 1996; Pfleeger et al. 1997)發現，度量理論內涵主要包括：(1)度量是一種實證關聯系統(Empirical Relation System)—度量是從真實世界的實體與屬性

對映到數值化世界的數字符號之系統化理論，透過對實體、屬性以及實體間的關係明確的定義後，轉換為數值化的程式，與(2)度量具有不同尺度類型與特質—度量包含幾種基本的尺度類型，用以擷取度量之實體屬性中更多的資訊，其中尺度的類型分別為：(a)名義尺度(Nominal Scale)，(b)順序尺度(Ordinal Scale)，(c)區間尺度(Interval Scale)，(d)比例尺度(Ratio Scale)，與(e)絕對尺度(Absolute Scale)。

以軟體度量之實體屬性而言，又可分為內部屬性與外部屬性。內部屬性可直接對實體進行度量取得度量值，例如物件耦合度，巢狀迴圈的深度等等，但多為技術層次資料；外部屬性為軟體產品品質與專案生產力績效等，雖無法直接對實體進行度量，需要透過間接度量反應相關數值事實，卻是管理者所重視的高階資訊。因此，軟體度量的重點是藉由量測實體的內部屬性，再透過模式的建構反應出軟體開發工作的外部屬性，並建立外部屬性的適當度量指標，進而評估或預估的事實，例如藉由度量軟體開發所耗費的容錯性與回復性，反應出目前軟體的可靠性(Fenton 1994; Riguzzi 1996)。

有關軟體專案度量的種類相當多，其中以軟體專案規模度量、成本、時程、生產力與品質等度量議題為軟體專案相關人員所重視(林信惠等 2002)。以下擬針對各種類之度量說明如下：

(一)軟體專案規模之度量

在進行軟體專案度量前，通常需要先計算出軟體規模以作為度量的基準(黃世禎 2004)。而目前用以計算軟體規模最常用及公認的方法為功能點分析法(Function Point Analysis; FPA)。在1983年時，Albrecht與Gaffnet提出功能點分析法，其主要概念是在系統初期之需求訪談階段中，從終端使用者的觀點進行軟體功能性規模大小之預估。由於功能點分析法的特點是能夠在系統開發的早期，預估軟體功能性的大小，因此適合應用於軟體專案規模之度量。

(二)軟體專案成本之度量

軟體成本度量主要在於蒐集軟體開發過程中完成一個工作項目或一個階段所耗費的實際工作量，並且能夠真實反應實際的成本度量值；但除了度量值的蒐集外，尚須有預先估計之成本估計值做為績效衡量指標的基準。因此，軟體成本的估計是軟體成本度量工作中首要進行的步驟，而成本預估工作主要取決在人力(Manpower)、工作量(Effort)和時程(Schedule)等三個項目(Vigder & Kark 1994)。由於軟體發展方法與工具以及電腦輔助軟體工程工具的不斷開發與進步，使得軟體成本估計工作愈形複雜。有鑑於此，Boehm等人於1995年針對原有的COCOMO成本估計模型再度進行修正，產出COCOMO II的新版本(Boehm et al. 1995)。COCOMO II改善了原本COCOMO只適用於瀑布式開發流程的缺點，而針對現代軟體開發流程，提出反覆式和漸進式估計的概念，使估計的結果能隨著專案的進行而提高正確性。同時，在實務應用方面，由於COCOMO II具有適用性廣、容易使用和過程透明等優點，因此，COCOMO II適合應用於中大型軟體專案之成本與時程估計工作(Boehm et al. 1995)。

在軟體專案進行過程中，除了須持續蒐集實際成本度量值外，為能夠對專案成本作

進一步的績效控管，差異分析法(Variance Analysis)則是歷史最悠久也是應用最普遍的一種衡量專案績效的方法(Saad 2003)。差異分析法具有簡單又容易使用的特性，透過專案預估值與實際值之差異比較，即可獲得專案成本與時程之績效(Saad 2003)。再者，根據黃明祥與盧煥文(2006)之研究發現，差異分析法應用於實務之專案控管效果良好。由此可知，COCOMO II估計模式與差異分析法經常被應用在軟體專案成本與時程之估計及控管工作。

(三)軟體專案時程之度量

軟體專案時程度量主要是量測軟體生命週期中各階段與開發活動所花費的時間，傳統在進行時程評估工作有三方面的問題：(1)人員方面：透過人員衡量時程的方式，通常取決於人員的直覺判斷。由於人員背景相異與主觀看法的不同，對於時程評估有明顯的差異，當面對時程嚴重落後時，人員也可能呈報不實資訊，使問題無法在短期間浮現及解決，(2)工具方面：度量工具選擇不當或缺乏相關工具，導致缺少或無法有效反映專案實際進度，因而無法即時發現問題，與(3)計算方式與制度方面：時間度量缺少精確的計算方式與制度，無法有效估算專案時程，造成專案初期常因無法準確預估時程，導致專案完工期限往往無法符合約定交期。基於上述問題，學者Simmons(1992)透過度量的方法來改善傳統時程評估的問題，利用客觀的量化數據來解決的人員主觀判斷或缺乏相關計算方式與制度等問題。因此，透過度量方式包含下列優點：(1)有效協助管理者進行專案時程規劃、時程管理和人力的派用，(2)降低專案失敗風險，與(3)大幅減少軟體開發和專案時程監控里程碑的次數與時間(Simmons 1992)。

進行專案時程度量與成本度量的共同作業在於須先對時程做預先估計以利績效的衡量。常用於一般專案的時程估計與控管方法，例如甘特圖(Gantt-Chart)、計劃評核術(PERT)與要徑法(Critical Path Method; CPM)等較不適用於中大型規模且分散式的軟體專案時程評估與控管(林信惠等 2002)。因此，本研究採用COCOMO II模式的時程度量公式作為時程預估的計算方法(Boehm et al. 1995; Boehm et al. 1997)。另一方面，有關專案時程績效控管，本研究同樣係採用差異分析法作為專案時程控管之方法。

(四)軟體生產力之度量

生產力(Productivity)是衡量軟體專案績效的一種重要指標，主要是衡量組織資源投入(Input)與產出(Output)之間相對效益，其定義為：「生產力=產出÷投入」(IEEE Std. 1045 1992；林信惠 1992；林信惠等 2002)。IEEE於1992年制訂有關軟體度量指標的相關標準為IEEE Std. 1045，其中，針對軟體生產力度量之定義為：「軟體生產力度量此專門術語係指確保度量資料的瞭解，包括：原始碼(Source Code)與文件產物(Document Production)。」由於軟體生產力度量標準中IEEE Std. 1045對於軟體生產力的產出及投入之計算方式均有詳細規範，因此，本研究係參考該標準作為軟體生產力度量之依據。

(五)軟體品質之度量

一般而言，軟體品質是指軟體滿足使用者需求能力與特性的程度(Pressman 2001)。有關軟體品質度量，學者林信惠等(2002)指出利用品質模式評估軟體品質是一種最常使

用的方法。在1991年，ISO/IEC 所制定的9126-1品質模式是目前普遍被國際公認為軟體品質模式中最詳細的一種標準。ISO/IEC(1991)將軟體品質模式定義為：「一組品質特徵與特性間之關聯，用以提供明確說明一個產品的品質需求與品質之評估。」因此，ISO/IEC 9126-1品質模式主要可分為六個品質主特性。其中，六個主要品質特性又各擁有一組次特性，共可展開為二十七個品質次特性(如圖1所示)。由於ISO/IEC 9126-1品質模式為每個子特性提供了明確的度量指標且符合國際認證標準，因此，該模式適用於軟體品質度量之用途。

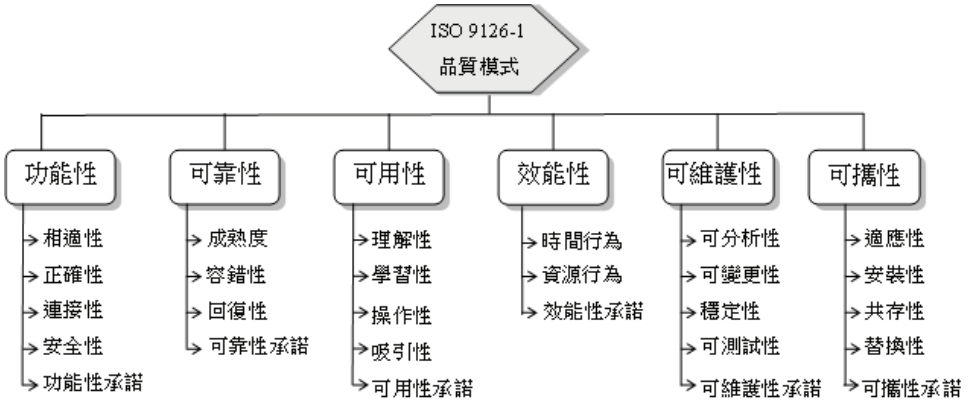

圖1：ISO/IEC 9126-1軟體品質模式
(資料來源：ISO/IEC 1991)

綜合以上所述，軟體專案度量的主要工作內容如下：「軟體專案度量為針對軟體專案規模、軟體專案成本、軟體專案時程、軟體生產力與軟體品質等五項度量，依據其特殊的內部屬性藉由度量模式進行量化數值蒐集以反應出外部屬性的事實，並發展一些合適的量測指標，用以評量與預測軟體專案開發的狀態與程度，據以提供專案管理者明確的決策資訊。」

二、智慧型代理人在軟體專案之應用

智慧型代理人(Intelligent Agent)一般被稱為軟體代理人(Software Agent)，由於智慧型代理人本身的多樣性，至今尚未有統一的定義，但可將其定義歸納為以下幾點：(1)它是一種電腦軟體程式或模組；(2)具有判斷能力並可執行代理者所賦予的工作任務；(3)有自我學習、調適的特性；(4)可協助使用者處理複雜且例行性工作減輕資訊處理的負擔。換言之，智慧型代理人是一種電腦程式，可協助使用者執行各種費時耗力的資訊蒐集工作，包括在網路中各種不同資訊來源之資訊搜尋及擷取、資訊不一致的重新解讀、不相關和多餘資訊的過濾、龐大與複雜資訊來源的整合以及隨著時間來調適使用者的需求等(Sycara et al. 1996)。基本上，智慧型代理人主要的三項特性如下(Chen & Su 2003)：(1)自治性(Autonomy)：代理人程式能自我行動，可控制自我行為與內部狀態，(2)社會

性(Social Ability)：代理人可與其他代理人透過資訊的傳遞與交換溝通、協調達到資訊交流，其溝通方式必須採用相同的資料傳輸格式和溝通協定才能順利進行，與(3)自主性(Reactivity)：代理人能依外在環境與各個代理人的狀態做適當的反應回饋。

一般而言，為使智慧型代理人具備推理機制，可在不同的狀態下做出選擇與決策的能力，主要透過下列三種方法所以賦予代理人推理機制(Kalakota & Whinston 1996)：(1)以規則為基礎的方法(Rule-based Approach)：事先寫好決定代理人必須採取行動的命令或規則，代理人依據規則行動，(2)以知識庫為基礎的方法(Knowledge-based Approach)：利用領域知識專家編寫大量資訊，再將資訊提供給代理人演繹出適當的行為，與(3)學習式方法(Learning Approach)：讓代理人一邊工作一邊學習，透過收集過去的統計資料，並吸收新的知識，使代理人學習未來的行動。

Yan等人(2000)曾提出使用多重代理人系統(Multi-agent System)方法協助分散式環境的專案管理，其使用的代理人分別為行動代理人(Activity Agent)、資源代理人(Resource Agent)與服務代理人(Service Agent)等三大類，其目的是提供分散式的軟體決策方案(Distributed Software Scheme)給專案人員，以提高各專案的決策能力並解決資訊蒐集因人為因素而造成資料失誤的問題。Sharma與Capretz(2001)曾將智慧型代理人技術應用於軟體應用程式之維護工作，其所發展之多重智慧型代理人系統包含下列數種特定任務之代理人：(1)監控代理人(Monitor Agent)：用來監控軟體原始碼，(2)衝擊代理人(Impact Agent)：用來判斷任何軟體原始碼的改變所可能造成的影響，(3)搜尋代理人(Search Agent)：用來搜尋相關的檔案，如軟體規格書和操作手冊等，與(4)電子郵件代理人(E-mail Agent)：用來自動傳遞電子郵件訊息給相關的軟體維護人員等。該系統藉由多重代理人之間的協調與合作，可用於解決網際網路環境中，開放原始碼之軟體程式的修改與維護的問題。

參、軟體專案度量程序模式之建立

本研究主要是探討在網際網路環境下大型軟體專案之度量工作，其中包括成本、時程、品質與生產力等軟體專案度量之重要議題，並針對度量所蒐集到的資訊進行狀態評量、決策與預測。因此，本研究根據前述軟體專案度量理論進行研究架構之設計(圖2)，進而推演出軟體專案度量程序模式(圖3)。研究架構設計主要分為三個階段，各階段採用輸入、處理與輸出(Input-Process-Output)之設計方式，待前一階段工作完成後再進入下一個工作階段。第一階段主要是針對各種度量議題，參考相關度量之模式與標準，並依照其度量模式與標準制定出相關的度量指標規格。接著，透過各種量化計算方法之採用，經由度量指標量化計算方法制定後，產出度量計量方法。其次，在軟體專案執行時，將相關資訊分別輸入至軟體專案度量程序模式，即能產出相關的專案度量資訊。

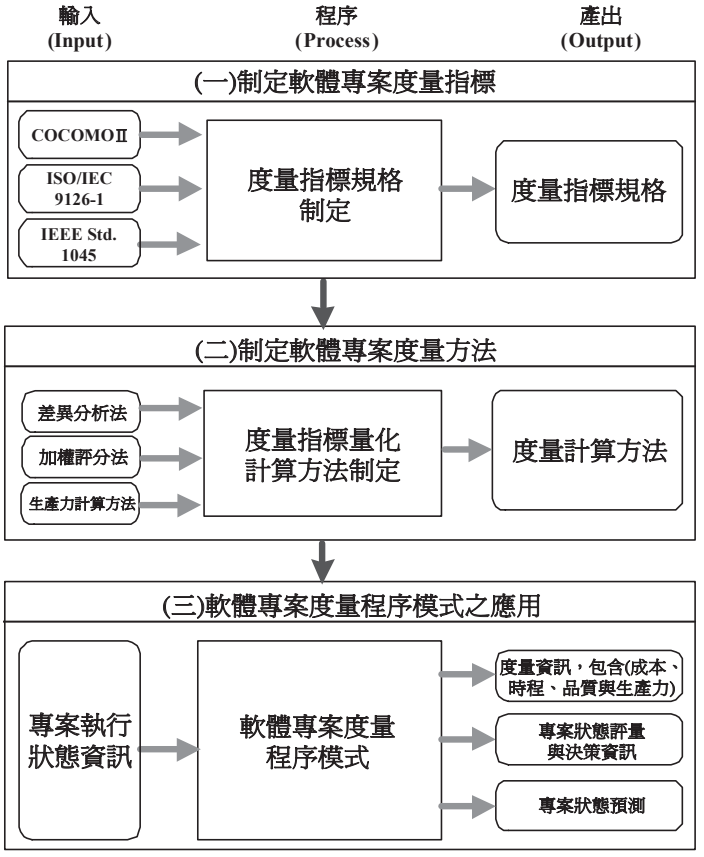

圖2：研究架構

基本上，軟體專案度量程序模式中，在成本度量方面，本研究採用的估計方法為COCOMO II估計模式，由於COCOMO II估計模式具有適用性廣、容易使用及應用過程透明等優點，因此適合應用於軟體專案之成本估計工作。在時程度量方面，COCOMO II估計模式亦提供時程預估之公式，因此，本研究採用COCOMO II作為時程估計的方法。其次，在取得軟體專案之實際成本與時程度量值後，即可運用差異分析法分別推算出專案的成本績效與時程績效。至於軟體品質度量部分，本研究採用ISO/IEC 9126-1軟體品質模式，由於ISO/IEC 9126-1為目前國際公認之品質模式，具有相當的代表性，適合作為軟體品質的度量指標。基本上，軟體品質模式中的主特性與次特性皆屬於高階屬性之意涵，非低階量化之數據。因此，難以將軟體品質度量指標予以彙總，進而得到總體軟體品質。有鑑於此，本研究首先利用加權評分法獲得專案人員與使用單位人員對於軟體品質度量指標之主特性與次特性的權重看法，接著依據各項度量指標特性之權重值與專案群組的評分值分別計算出軟體品質特性之量化數值，依次操作直到獲得所有軟體品質主特性之量化數值，最後經由彙總即得到該軟體產品品質的量化數值。最後，軟體生產力度量方面，由於IEEE Std. 1045對於軟體生產力度量均有明確的規範，本研究乃根據IEEE Std. 1045軟體生產力度量標準定義軟體生產力之度量。

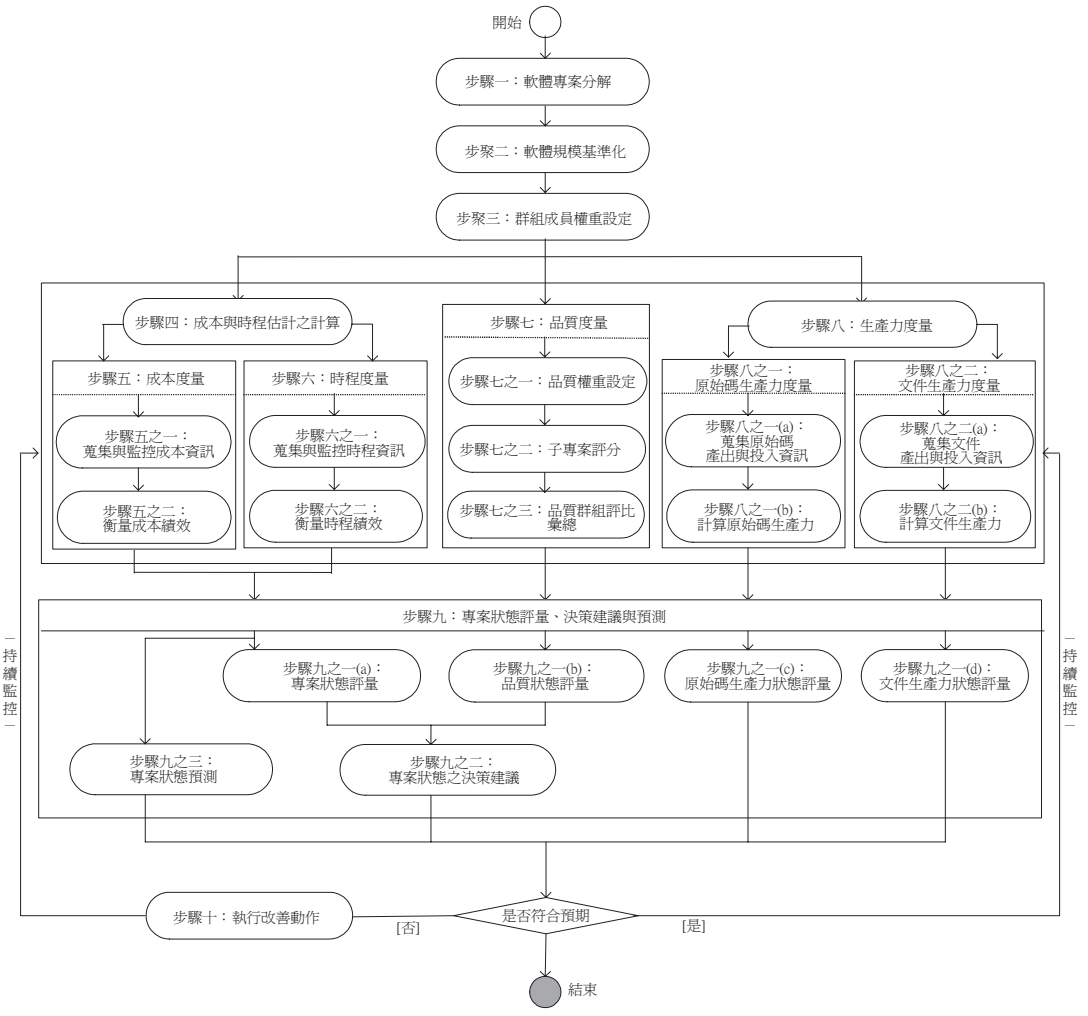

圖3：軟體專案度量程序模式

有關軟體專案度量程序之實施步驟，本研究進一步說明如下：

步驟一：軟體專案分解

本研究所進行的軟體專案度量是以大型軟體專案為度量對象，由於現今的大型軟體專案通常是由數個子專案協同開發所組成。因此，首先須將軟體專案分解為多個子專案，以利後續度量作業的進行。子專案開發則由不同開發團隊，分散在不同位置負責開發與建置工作。

步驟二：軟體規模基準化

將大型軟體專案分解成多個子專案後，為能夠有統一的度量基準，採用功能點分析法來對各個子專案進行軟體規模估計，其輸入資料為各個子專案之功能規格或需求規

格，而輸出資料為功能點數與原始碼行數，其功能點計算公式如下：

$$TUFP = \sum_{i=1}^{5}\sum_{j=1}^{3} W_{ij} \times Z_{ij}$$

其中TUFP為未調整之功能點數，W_{ij}為其對應之權重，而Z_{ij}則為功能元件i在複雜度層次為j的數目總和，其功能元件i主要涵蓋五類：外部輸入(External Input)、外部輸出(External Output)、邏輯內部檔案(Logical Internal File)、外部介面檔案(External Interface File)、外部查詢(External Inquiry)。而每一類的功能元件又分為低(Low/Simple)、中(Average)、高(High/Complex)三種複雜程度 j (Albrecht & Gaffney 1983; Maston et al. 1994)。在得出系統之未調整功能點數之後，可依其所採用的程式語言或開發工具，透過功能點與原始碼行數對照表計算出預估的原始碼行數(Jones 1999)。

步驟三：設定群組成員權重

一個大型軟體專案的度量工作通常是由多人共同完成，由於各個成員在專案中所扮演的角色不同，因此，所評估之度量值也有比重的不同，所以需計算各個度量成員的權重。度量成員權重所採用的指標，根據Buglione 與 Abran(1999)採用學歷權重(Degree Weight; DW)、經驗權重(Experience Weight; EW)與角色權重(Role Weight; RW)等三個指標計算出群組成員權重值(Group Weight; GW)。

有關個別成員之DW_S、EW_S與值RW_S，本研究乃根據黃明祥與盧煥文(2006)先前之研究，在DW_S值部分：博士為1、碩士為0.8、大學為0.6、專科為0.4；在EW_S值部分：年資≧10為0.5、年資>5為0.3、年資>1為0.2、年資≦1為0.1；在RW_S值部分則由專案經理人彈性制定各類角色的權重值。

群組成員權重計算計算過程如下列公式。首先，給定個別成員DW_S、EW_S與RW_S值，公式中s代表群組中第s位成員，將其DW_S、EW_S與RW_S予以合計，除以群組中共P位人員DW_S、EW_S與RW_S值之合計值，即可得到該成員在群組內的權重值GW_S。

$$GW_s = \frac{(DW_s + EW_s + RW_s)}{\sum_{s=1}^{p}(DW_s + EW_s + RW_s)}, s = 1,..., p$$

步驟四：成本與時程之估計

1.軟體專案成本之估計

(1)個別成員估計軟體子專案已調整之開發工作量之基本公式

$$Effort_{ks} = A \times \left(Size_{ks}\right)^{1.01+\sum_{i=1}^{5} SF_i} \times \prod_{j=1}^{17} EM_j, s = 1,..., p, k = 1,...., n$$

其中，$Effort_{ks}$為個別成員估計軟體子專案已調整之開發工作量，A為軟體專案所設定的調準係數，預設值為2.94(Boehm 2000)，Size為預估之原始碼行數，SF_i為五項規模因素(有無相關專案之前例、開發彈性、架構與風險解決、團隊的向心力、程序成熟度)，EM_j為十七項工作量乘數(Boehm 1981; Boehm et al. 1995; Boehm et al. 1997; Boehm 2000)，s為1到P位群組成員，k表示第k個子專案，全部共有n個子專案。

(2)軟體子專案之成本估計群組彙總公式

$$Effort_k = \sum_{s=1}^{p} (Effort_{ks} \times GW_s), s = 1,...., p, k = 1,..., n$$

其中，$Effort_k$ 為軟體子專案的成本估計群組彙總值，GW 代表群組成員之權重值，s 為1到P位群組成員，k 表示第 k 個子專案，全部共有個 n 子專案。

(3)整體軟體專案之成本估計群組彙總公式

$$Total\_Effort = \sum_{k=1}^{n} Effort_k, k = 1,...., n$$

其中，$Total\_Effort$ 為整體軟體專案的成本估計群組彙總值，$Effort_k$ 為軟體子專案的成本群組彙總值，k 表示第 k 個子專案，全部共有 n 個子專案。

2. 軟體專案時程之估計

$$Schedule_k = \left[C \times (Effort_k)^{(0.33+0.2\times\sum_{i=1}^{5} SF_i)} \right] \times \frac{SCEDPercentage}{100}$$

(1)軟體子專案之時程估計公式

其中，$Schedule_k$ 為軟體子專案已調整之時程估計值，C為軟體專案所設定的調準係數，預設值為2.66(Boehm et al. 1997)，$Effort_k$ 是軟體子專案的成本估計群組彙總值，SF_i 為五項規模因素，$SCEDPercentage$ 為工作量乘數中專案時程之限制(Boehm 1981; Boehm et al. 1995; Boehm et al. 1997)。

(2)整體軟體專案之時程估計公式

$$Total\_Schedule = \left[C \times (Total\_Effort)^{(0.33+0.2\times\sum_{i=1}^{5} SF_i)} \right] \times \frac{SCEDPercentage}{100}$$

其中，$Total\_Schedule$ 為整體軟體專案開發時程估計值，C為軟體專案所設定的調準係數，$Total\_Effort$ 是整體軟體專案的成本估計群組彙總值，SF_i 為五項規模因素，$SCEDPercentage$ 為工作量乘數中專案時程之限制(Boehm 1981; Boehm et al. 1995; Boehm et al. 1997)。

步驟五：成本之度量

1.步驟五之一：蒐集與監控成本資訊

軟體專案成本估計值計算以後，接下來便能以此成本估計資訊做為專案成本控管的基準。因此，當軟體專案活動開始進行時，專案管理者必須能在專案執行過程中進行各個子專案的成本資訊蒐集與監控，此時，專案管理者將能針對整體專案和子專案進行成本績效控管，以確實掌握軟體專案的成本狀態。

2.步驟五之二：衡量成本績效

差異分析法(Variance Analysis)可作為衡量專案成本與時程績效之作法(Saad 2003)。運用差異分析法推算出「成本差異(Cost Variance)」與「時程差異(Schedule Variance)」以達成專案成本與時程之績效控管。因此，當蒐集到整體專案和子專案執行時實際發生之成本值，接著便可將其與原先預估之成本值進行比較，據以獲得整體專案及子專案之成

本績效，其基本公式如下：

$$\text{成本差異（\%）} = \frac{\text{（實際發生之成本-原先預估之成本）}}{\text{原先預估之成本}} \times 100\%$$

步驟六：時程之度量

1.步驟六之一：蒐集與監控時程資訊

經由先前的軟體專案的時程預估結果之後，將時程估計資訊做為專案控制的基準。當軟體專案活動開始進行之後，由於各個子專案皆有其獨立的開發生命週期，時程進展情況也不相同。因此，專案管理者必須蒐集與監控各個子專案時程資訊，以瞭解目前各個子專案工作團隊的實際進度，方能掌握專案整體的時程狀態。

2.步驟六之二：衡量時程績效

當蒐集到各個子專案工作團隊實際的進度資訊時，接著便可將其與原先預估的時程進行比較，以衡量整體專案及子專案的時程差異的績效度量值，其基本公式如下：

$$\text{時程差異（\%）} = \frac{\text{（實際運作之時程-原先預估之時程）}}{\text{原先預估之成本}} \times 100\%$$

步驟七：品質度量

有關本研究之品質度量考慮多個評估者和多個子系統的主要理由如下：(1)基於許多大型軟體專案通常分割成多個子專案的形式，而每個子專案則由分散在各地的不同的專案團隊人員所負責進行開發。再者，軟體專案是由多位具有相當專業能力的專案人員所共同完成的智慧資產。因此，本研究在進行軟體品質度量時，會考量各個子專案成員(評估者)的權重值以進行群組估計，並進一步計算出子專案或整體專案的品質績效，與(2)基本上，此種作法的優點主要如下：(a)獲得各個子專案或整個大型專案的績效資訊更完整，與(b)多人同時進行估計是一種可達成共識的作法，同時希望去除極值並減少評估者受權威人士的影響，以求公正與客觀化(林信惠等 2002)。

1.步驟七之一：品質權重設定

ISO/IEC 9126-1品質模式是目前國際公認最為詳細的軟體品質度量標準。因此，本研究採用此標準作為軟體品質度量指標之依據。其中，品質主特性是依據ISO/IEC 9126-1(1991)所制定的品質模式，其主要包括六個品質主特性：功能性(Functionality)、可靠性(Reliability)、可用性(Usability)、效能性(Efficiency)、可維護性(Maintainability)、可攜性(Portability)，而個別的主特性之下又包含數個次特性。因此，品質度量是以實際所進行的軟體專案為評估對象，透過問卷調查專案人員與客戶對於該軟體品質的看法，本研究是採用李克特五等分量尺(極重要、重要、普通重要、不重要、極不重要)，待問卷回收後進行專案品質權重值計算。計算後的品質主特性權重值再由專案管理者進行專案品質權重值設定，作為專案品質主特性權重依據。首先，品質主特性的原始權重值為RDW_i，經由公式計算後的權重值為W_i，而權重值加總後其值為1。專案主特性品質權重計算過程如下列公式。其中，RDW_i為問卷所搜集之品質特性原始權重看法，i代表任一品質特性(例如功能性)，除以所搜集的六個品質主特性原始權重值加總，即可得到第i個

品質主特性之權重W_i值。

$$(1)\ W_i = \frac{RDW_i}{\sum_{1}^{6} RDW_i}, i = 1,...,6$$

品質主特性下之各個次特性權重值W_{ij}之計算過程如下列公式，其中，W_{ij}代表第i個品質特性下第j個次特性之權重值，每個品質主特性下則有1到m個不等的次特性。

$$(2) W_{ij} = \frac{RDW_{ij}}{\sum_{j=1}^{m} RDW_{ij}}, i = 1,...,6, j = 1,...,m$$

2.步驟七之二：子專案品質評比

在專案軟體產品產出後，由專案群組人員進行各個子專案品質評比，可得到評比之品質分數。首先，針對專案中子專案中各個品質主特性下的次特性進行評比，並將其為彙總為子專案品質主特性品質分數。其中，S_{ksi}代表第s位評審人員對第k個子專案第i個品質主特性之評分(例如，i=1代表功能性品質主特性，i=2代表可靠性品質主特性，餘此類推)。S_{ksij}代表第s位評審人員對第k個子專案第i個品質主特性之下之第j個次特性評分。當評比結束後，將評比值S_{ksij}分別乘上各自的權重值w_{ij}可得到次特性加權分數，將其合計後可得到主特性品質加權分數，其計算過程如下列公式：

$$(1) S_{ksi} = \sum_{j=1}^{m} (S_{ksij} \times W_{ij}), i = 1,..,6, j = 1,....,m, k = 1,...,n, s = 1,...,p$$

第二部分則將子專案的品質進行加權計分，其中，S_{ksi}代表第s位評審人員對第k個子專案第i個品質主特性之評分。將評比值分別乘上個別品質特性之權重值，便可得到軟體子專案品質主特性加權分數的度量值(Software Sub-Project Quality Measurement; SSPQM)。其中，W_i代表第i個品質主特性之權重。第s位評審人員對第k個子專案各個品質主特性評比計算過程如下列公式：

$$(2)\ SSPQM_{ks} = \sum_{i=1}^{6} (S_{ksi} \times W_i), i = 1,....,6, k = 1,...,n, s = 1,...,p$$

3.步驟七之三、品質群組之彙總

一個大型軟體專案的評比通常是由多人共同完成，當專案群組成員將各自的評比工作完成後，即可將各個成員做進行的評比值予以彙總。並利用先前設定的群組人員權重，計算出最終的加權評比結果。本研究建立之專案品質度量計算公式，透過先前設定的群組成員權重與原始軟體子專案品質度量值，將其彙總為群組軟體子專案品質度量值(Group Software Sub-Project Quality Measurement; GSSPQM)，彙總計算過程如下列公式。其中，GW代表群組成員之權重值，s則為1到P位群組成員。

$$(1) GSSPQM_k = \sum_{s=1}^{p} (SSPQM_{ks} \times GW_s), s = 1,....., p$$

將群組軟體子專案品質度量值彙總後,可得到整體專案品質度量值(Software Project Quality Measurement; SPQM)，其計算過程如下列公式：

$$(2)\ SPQM = \sum_{k=1}^{n} (GSSPQM_k \times WT_k), k = 1, ..., n$$

其中，WT_k為子專案k的權重，表示子專案k的成本估計值佔整個專案總成本估計值的比率，因此WT_k的計算公式如下：

$$WT_k = \frac{Effort_k}{\sum_{k=1}^{n} Effort_k}, k = 1, ..., n$$

其中，$Effort_k$為軟體子專案的成本估計群組彙總公式，表示第k個子專案，全部共有n個子專案。

步驟八：生產力之度量

有關軟體生產力之度量，本研究係根據IEEE Std. 1045(1992)針對原始碼(Source Code)與文件產物(Document Production)等二項軟體生產力度量指標，分別計算出原始碼生產力與文件生產力。再者，軟體生產力的計算，其概念為生產力=產出÷投入。因此，本研究乃根據此二個度量指標進行軟體生產力度量。

步驟八之一：原始碼生產力

1.步驟八之一(a)：蒐集原始碼之產出與投入資訊

當軟體專案活動開始進行時，專案管理者必須要能在專案執行過程進行軟體生產力資訊的監控，此時，專案管理者應能蒐集整體專案和子專案實際之原始碼生產力產出與投入資訊，進一步推算出原始碼之生產力，以確實掌握軟體開發情況。

2.步驟八之一(b)：計算原始碼之生產力

在計算原始碼產出部分，本研究根據IEEE Std. 1045(1992)之Physical Source Statement, PSS計算方式，其Source Statement即代表原始碼行數(Lines of Code; LOC)。Source Statement計算包括Original Software Module及New Software Module二個模組。因此，計算項目涵蓋Removed SS、Modified SS、Reused SS與Added SS；在計算原始碼投入部分，本研究採用直接交付人員(Direct Delivered)，即從事系統分析、設計、編碼、測試等直接參與系統開發的人員(林信惠等 2002)進行原始碼產出工作所投入的人時(Staff-Hour)。因此，原始碼生產力(Source Code Productivity; SCP)計算公式如下：

(1) 子專案原始碼之生產力計算

$$SCP_k = \frac{\text{Re}moved\_SS_k + Modified\_SS_k + \text{Re}used\_SS_k + Added\_SS_k}{LOC\_Direct\_Delivered\_Staff\_Hour_k} \times 100\%$$

其中，SCP_k代表子專案原始碼生產力，$\text{Re}moved_SS_k$為子專案SS的移除數量，$Modified_SS_k$為子專案SS的修改數量，$\text{Re}used_SS_k$為子專案SS的再用數量，

$Added_SS_k$為子專案SS的新增數量，$LOC_Direct_Delivered_Staff-Hour_k$為子專案之直接交付人員進行原始碼產出工作所投入的人-時。

(2) 整體專案原始碼之生產力計算

$$Total\_SCP=\frac{\sum_{k=1}^{n}\text{Re}moved\_SS_k+\sum_{k=1}^{n}Modified\_SS_k+\sum_{k=1}^{n}\text{Re}sused\_SS_k+\sum_{k=1}^{n}Added\_SS_k}{\sum_{k=1}^{n}LOC\_Direct\_Delivered\_Staff\_Hour_k}\times 100\%, k=1,...,n$$

其中，$Total_SCP$代表整體專案的原始碼生產力，k表示第k個子專案，全部共有n個子專案。

步驟八之二：文件生產力

1.步驟八之二(a)：蒐集文件產出與投入資訊

在軟體專案活動開始進行時，專案管理者必須要能在專案執行過程進行軟體生產力資訊的監控，此時，專案管理者應能蒐集整體專案和子專案實際之文件生產力產出與投入資訊，進而推算出文件生產力，以確實掌握軟體開發情況。

2.步驟八之二(b)：計算文件之生產力

在計算文件產出部分，本研究根據IEEE Std. 1045(1992)之Document Page Count文件計算方式，即計算不包含空白頁(Blank Pages)的文件總頁數，有關常見的文件種類主要有：專案計畫書(Project Plan; PP)、系統規格書(System Specification; SS)、需求分析書(Requirement Analysis; RA)、設計規格書(Design Specification; DS)、程式規格書(Program Specification; PS)、測試規格書(Test Specification; TS)、系統操件手冊(System Operation; SO)、系統技術手冊(System Technique; ST)、系統安裝手冊(System Installation; SI)、系統維護手冊(System Maintenance; SM)等十種(黃明祥 & 盧煥文 2006)；在計算文件投入部分，本研究同樣採用直接交付人員進行文件製作所投入的人時。因此，文件生產力(Document Productivity; DP)計算公式如下：

(1) 子專案文件之生產力計算

$$DP_k=\frac{PPC_k+SSC_k+RAC_k+DSC_k+PSC_k+TSC_k+SOC_k+STC_k+SIC_k+SMC_k}{Doc\_Direct\_Delivered\_Staff\_Hour_k}\times 100\%$$

其中，DP_K代表子專案文件生產力計算，PPC_K為子專案之專案計畫書總頁數，SSC_K為子專案之系統規格書總頁數，RAC_K為子專案之需求分析書總頁數，DSC_K為子專案之設計規格書總頁數，PSC_K為子專案之程式規格書總頁數，TSC_K為子專案之測試規格書總頁數，SOC_K為子專案之系統操作手冊總頁數，STC_K為子專案之系統技術手冊總頁數，SIC_K為子專案之系統安裝手冊總頁數，SMC_K為子專案之系統維護手冊總頁數，$Doc_Direct_Delivered_Staff_Hour_k$為子專案之直接交付人員進行文件製作所投入的人-時。

(2) 整體專案文件之生產力計算

$$Total\_DP=\frac{\sum_{k=1}^{n}PPC_k+\sum_{k=1}^{n}SSC_k+\sum_{k=1}^{n}RAC_k+\sum_{k=1}^{n}DSC_k+\sum_{k=1}^{n}PSC_k+\sum_{k=1}^{n}TSC_k+\sum_{k=1}^{n}SOC_k+\sum_{k=1}^{n}STC_k+\sum_{k=1}^{n}SIC_k+\sum_{k=1}^{n}SMC_k}{\sum_{k=1}^{n}Doc\_Direct\_Delivered\_Staff\_Hour_k}\times 100\%, k=1,...,n$$

其中，*Total_DP*代表整體專案的文件生產力，*k*表示第*k*個子專案，全部共有*n*個子專案。

步驟九：專案狀態評量、決策建議與預測

1.步驟九之一：專案狀態評量

在完成上述各項度量後，如何將軟體專案的度量結果進行狀態評量、決策與預測乃是相當重要的步驟。因此，在獲得相關的各種專案度量資訊之後，接著是進行狀態評量的動作，以瞭解目前專案與各個子專案的執行狀態，如成本超支或節餘、時程進度超前或落後、品質是否符合需求、生產力是良好或低落等。在評量出目前專案與各個子專案的現狀後，即可作為後續專案決策建議之分析依據。

(a)專案狀態之評量

基於預算與時程是專案控管的主要依據(林信惠等 2002)。有鑑於此，本研究係根據林信惠等(2002)對於探討預算、時程績效與專案管理的意義，研究指出以專案成本與時程狀態數值10%作為評量專案狀態之基準(Baseline)是相當合適且易於專案管理者進行專案控管。因此，以成本與時程績效初步評量專案狀態結果如表1所示。

表1：專案成本與時程初步狀態評量結果

成本績效狀態 / 時程績效狀態	預算節餘 (大於-10%)	預算符合預期 (±10%)	預算透支 (大於+10%)
時程超前 (大於-10%)	《狀態1》 預算節餘 且時程超前	《狀態2》 預算符合預期 且時程超前	《狀態3》 預算透支 但時程超前
時程符合預期 (±10%)	《狀態4》 預算節餘 且時程符合預期	《狀態5》 預算及時程 均符合預期	《狀態6》 預算透支 但時程符合預期
時程落後 (大於+10%)	《狀態7》 預算節餘 但時程落後	《狀態8》 預算符合預期 但時程落後	《狀態9》 預算透支 且時程落後

(資料來源：本研究整理)

根據林信惠等(2002)指出，採用綜合性的評估指標是一種化繁為簡的方式，同時可依不同的目的選擇報告的內容，綜合性、摘要性的資料較符合管理階層的需求。因此，本研究擬將初步專案狀態評量結果，依專案問題之嚴重性進一步分類如表2所示，分類A代表預算與時程均在預期範圍內或符合預期，分類B為預算透支但時程在預期範圍內或符合預期，分類C表示時程落後但預算在預期範圍內或符合預期，分類D則是預算透支且時程落後。

表2：專案狀態之分類

專案狀態分類	分類條件	綜合專案狀態
分類A	代表預算與時程均在預期範圍內或符合預期	狀態1、2、4、5
分類B	代表預算透支但時程在預期範圍內或符合預期	狀態3、6
分類C	代表時程落後但預算在預期範圍內或符合預期	狀態7、8
分類D	代表預算透支且時程落後	狀態9

(資料來源：本研究整理)

(b)品質狀態之評量

將品質群組彙總後的評比結果進行品質狀態之評量，瞭解目前品質度量值所代表的意涵。因此，本研究參考(ISO/IEC 1991; Fusani 1995; Buglione & Abran 1999)先前之研究，將品質評比等級(Rating Levels)分成以下五類：極好的、良好的、尚可的、粗劣的與品質不存在的。同時，這五類品質評比等級又依全面性等級(Global Rating)劃分成二類之品質狀態：(1)滿意(涵蓋極好的、良好的、尚可的)，與(2)不滿意(包括粗劣的與品質不存在的)，如表3所示。

表3：品質狀態之評量

品質評比等級(Rating Levels)	評比區間	品質狀態(Global Rating)
極好的	0.9～1.00	滿意
良好的	0.7～0.89	
尚可的	0.5～0.69	
粗劣的	0.3～0.49	不滿意
品質不存在的	0.0～0.29	

(資料來源：本研究整理)

(c)原始碼與文件生產力狀態之評量

在計算出原始碼與文件生產力度量值後，本研究係參考林信惠(1992)與林信惠等(2002)先前之研究，將生產力狀態評量分為三個等級，如表4所示。若生產力高於90%以上代表生產力良好，70%至90%代表生產力普通，如果生產力低於70%以下則代表生產力低落。在獲得相關的生產力狀態資訊之後，專案管理者即可得知整體專案和子專案目前的原始碼與文件生產力現況。有關步驟九之一專案相關狀態之評量標準可依據實務公司對軟體專案度量的實際需求與情況進行調準，以提升度量資訊的準確性。

表4：生產力狀態之評量

生產力 \ 生產力狀態	生產力狀態
90%以上	生產力良好
70%~90%	生產力普通
70%以下	生產力低落

(資料來源：本研究整理)

當專案完成上述狀態評量工作後，便能進一步進行專案決策的分析工作。

2.步驟九之二：專案狀態之建議方案

根據林信惠等(2002)之研究指出，決定專案成敗的三項主要因素為時程、成本與品質。有鑑於此，本研究擬採用專案狀態搭配品質狀態分析出各種專案狀況下可行的因應方案，如表5所示。例如：某一個子專案的狀態評量結果為分類C(代表時程落後，但預算在預期範標內或符合預期)，但是品質狀態是另人不滿意的。因此，專案管理者可以根據表5綜合專案狀態與品質狀態之建議得知，目前此專案應加派人手或儘速趕工，但是必須檢討整體專案狀態，加強品質之控管，以防止專案擴大化(Escalation)。

表5：綜合專案狀態與品質狀態之建議

品質狀態 / 專案狀態	滿意	不滿意
分類A	專案執行最佳狀態，並持續維持品質	專案執行在預算與時程內，但應檢討並加強品質控管
分類B	應增加工作量或縮短專案時程，並持續維持品質	應增加工作量或縮短專案時程，在範圍內加強品質控管
分類C	加派人手或儘速趕工，但需維持品質	加派人手或儘速趕工，但須檢討整體專案狀態，加強品質控管
分類D	應重新檢視整體專案的開發工作，修正人力配置與原先預估時程，但品質維持不變	應重新檢視整體專案的開發工作，修正人力配置與原先預估時程，並檢討整體專案品質

(資料來源：本研究整理)

3.步驟九之三、專案狀態預測

軟體專案度量資訊不僅能進行專案狀態評量與決策建議，並且可針對目前專案進行專案狀態預測，透過蒐集專案已發生之實際成本與時程資訊，配合所調整的未發生預估成本與時程資訊，即可預測專案後續進行的成本與時程狀態，以提升專案控管之效益(林信惠等 2002)，其計算公式如下：

(1)預計完工的總成本=已發生的實際成本+調整的未發生成本

調整的未發生成本=(已發生的實際成本/已發生的預估成本)×未發生的預估成本

(2)預計完工的總時程=已發生的實際時程+調整的未發生時程

調整的未發生時程=(已發生的實際時程/已發生的預估時程)×未發生的預估時程

在得到預計完工後的總成本及時程後，將原先預估之總成本與時程進行比較，衡量目前所預測總成本(或時程)與原先預估總成本(或時程)之落差，據以預測專案後續的可能狀態，其預測公式如下：

$$(1)\text{專案執行預期成本落差}=\frac{(\text{預計完工的總成本-原先預估的總成本})}{\text{原先預估的總成本}}\times 100\%$$

$$(2)\text{專案執行預期時程落差}=\frac{(\text{預計完工的總時程-原先預估的總時程})}{\text{原先預估的總時程}}\times 100\%$$

步驟十：執行改善動作

軟體專案度量循環的最後一個步驟是依據目前專案狀態及因應的決策方案，執行必要的修正動作，例如當專案發生成本超支、時程落後或品質不滿意等狀況，便可針對執行狀況較差的子專案進行調整或重新配置等措施，並將相關的狀態與決策資訊通知各個專案團隊成員。相對地，當專案狀態符合預期時，則持續進行專案的監控，重覆執行模式中的步驟四中到步驟九，直到專案開發完成結束。

整體而言，本研究所提出的軟體專案度量程序模式，其應用範圍涵蓋整個軟體專案開發過程，可作為專案管理者在進行大型軟體專案度量工作時的一套標準化作業程序。有鑑於此，本研究擬透過系統實作以瞭解軟體專案度量程序模式之可行性與實用性，同時，該系統亦可提供實務界在軟體專案度量應用之用途。

肆、系統實作

一、個案背景分析

本論文的實證分析對象為國內一家大型鋼鐵公司（簡稱A公司），主要產品為冷軋鋼品、線條、鋼圈、L型鋼等。由於A公司研發部門成員的素質相當整齊且擁有相當高水準的技術研發能力，因此A公司研發部門經理希望能夠整合與運用先前研發人員累積的經驗與知識，乃著手進行一個智慧型軟體專案知識入口網站開發與建置之大型專案計畫，經由初步的需求分析後，該專案分成三個不同的子專案(整合式軟體專案入口網站子專案、知識本體描述子專案、智慧型代理人子專案)，交由分散在不同開發地點之專案團隊負責開發，最後再進行整體專案的整合工作。

根據A公司之專案管理者指出，先前進行分散式中大型軟體專案開發所遭遇的問題如下：(1)分散式大型軟體專案計畫推動困難，經常面臨專案問題擴大化，(2)專案人員之間的溝通和協調工作缺乏效率，(3)專案管理者難以隨時掌握專案各種現況，(4)軟體專案度量工作不易落實與進行，且耗費極大的人力、時間和成本，(5)度量資訊的品質、完整性與時效性難以確保，與(6)專案管理者往往憑個人經驗主觀判斷專案狀態，容易造成誤差之情況發生。有鑑於此，A公司之專案管理者認為，在專案執行過程中，藉助於一套合適的專案度量輔助工具，能夠有效協助專案管理者進行分散式大型軟體專案的評估與監控，使專案順利開發完成是實務所需要的。因此，經過A公司研發部門內部會議討論後，基於以下考量而不直接採購度量相關套裝軟體的主要原因如下：(1)直接採購套裝軟體的方式雖然快速，但考量後續的軟體維護成本及修改成本，可能會造成A公司更大的

負擔，(2)目前市面上的套裝軟體售價相當高，對A公司而言，反而是一種極大的負擔，與(3)由於目前市面上的套裝軟體主要是依據國外公司情況所進行開發，未必能國內企業之需求。再者，自行發展系統的優點，即能配合A公司之實際情況、需求與歷史資料以進行相關參數調準，進而增加系統的彈性與度量資訊的準確性。所以A公司決定與本研究團隊進行產學合作，並導入本研究所發展的軟體專案度量資訊系統於該專案之軟體專案度量活動。

二、系統架構

基本上，該大型軟體專案是分割成數個可獨立的子專案，並分散在不同地點進行開發。因此，在分散式的開發環境下，每一個子專案都由所負責的開發團隊進行開發工作，也各有其獨立的軟體開發生命週期。由於每一個子開發團隊均專注於軟體專案中所負責的部份，加上各個開發團隊的所負責之子專案功能相異，執行期間狀況也有所不同。此時，專案管理者必須有效地控管各個子專案的開發現況，例如：當某個子專案面臨預算透支、時程落後、生產力低落或品質不滿意時，專案管理者應能及早獲知這些狀態資訊，並掌握專案過程中的異常情況，透過網路發佈命令指派，採取適當的因應措施，以防止專案問題擴大化(Escalation)。為解決上述問題，本研究根據軟體專案度量程序模式發展一套以智慧型代理人為基礎的軟體專案度量資訊系統，協助專案管理者在分散式軟體專案開發環境中進行軟體專案度量工作。如圖4所示，該系統主要涵蓋以下部分：(1)軟體專案度量資訊系統：包括度量資訊系統介面、軟體專案度量程序應用、軟體專案度量資料庫，(2)知識庫系統：包含知識擷取介面、工作記憶區、知識庫、推論機，與(3)智慧型代理人程式：狀態評量代理人、決策分析代理人、訊息通知代理人等。

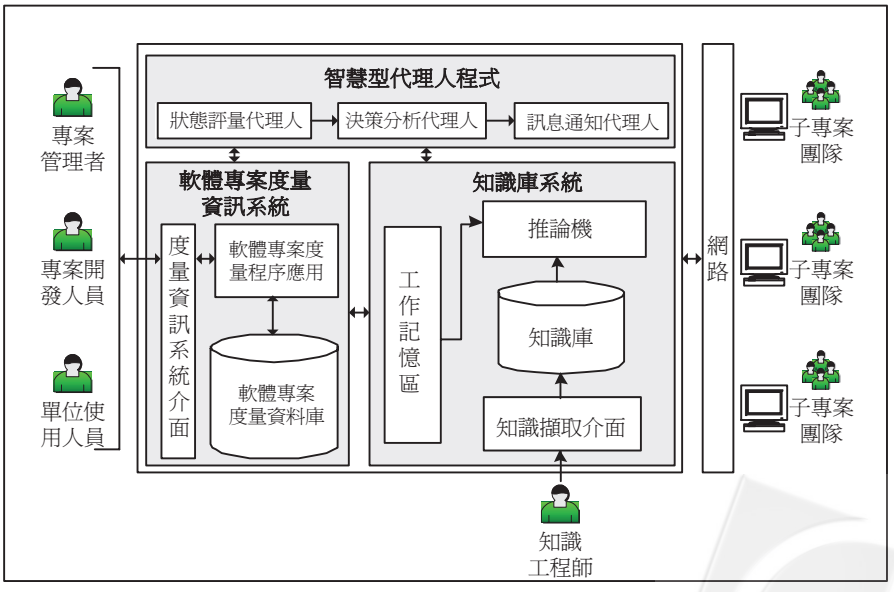

圖4：系統架構圖

以下針對系統架構中各個元件之功能進行說明：

(一)使用者與開發團隊

本系統主要使用者有專案管理者、專案開發人員與單位使用人員等角色。當使用者欲透過本系統進行軟體專案度量時，可透過度量資訊系統介面進行操作。而子專案開發團隊也可以透過本系統回授目前子專案之開發現況。

(二)軟體專案度量資訊系統

1. 度量資訊系統介面：本系統的使用者依據權限而分別擁有不同的介面功能，操作上則以Web介面的瀏覽器(Browser)為主，透過瀏覽器以HTTP的方式遠端登入至系統中進行操作。
2. 軟體專案度量程序應用：進行軟體專案度量活動之相關步驟，包含成本、時程、生產力與品質等度量議題。
3. 軟體專案度量資料庫：資料庫主要用於貯存所有現行與歷史專案資料，專案執行期間與歷史性的度量資料同時也貯存於資料庫中。

(三)知識庫系統

1. 知識擷取介面：知識工程師將軟體專案度量的相關規則(Rule)透過知識擷取介面將規則編著並儲存到知識庫中。
2. 工作記憶區：工作記憶區儲存所有工作階段中的運算資料，其資料為暫時性而非永久性，當系統關機或重新載入時，資料便會遺失。
3. 知識庫：知識庫中儲存所有以規則呈現的知識內容，而規則的形式則是以if-then-else的方式來表達。例如：當專案狀態評量結果為“預算節餘且時程超前”，其規則為if成本績效狀態(大於-10%) and時程績效狀態(大於-10%) then顯示專案預算節餘且時程超前。
4. 推論機：負責處理知識規則的推論運算，可透過向前推論或向後推論的方式，以推導出新的事實(Fact)。例如：欲推論出目前專案的初步狀態，此時，需將工作記憶區運算後所儲存的成本狀態與時程狀態二者資訊，根據知識庫所制定的專案狀態評量規則，透過推論機推論其結果(例如：推論結果為目前專案預算節餘且時程超前)。

(四)智慧型代理人程式

1. 狀態評量代理人：狀態評量代理人根據實際的成本、時程、生產力及品質等度量資訊，以判斷出整體專案和子專案的現況，並將相關專案狀態資訊傳遞給決策分析代理人。
2. 決策分析代理人：決策代理人負責根據所獲得專案狀態資訊來進行決策推理，其決策的過程會將相關資訊交由知識庫系統之推論機來進行推論運算，以判斷專案目前的現況，最後並將相關決策建議資訊傳遞給訊息通知代理人。
3. 訊息通知代理人：訊息通知代理人負責將最新的專案狀態資訊自動傳遞給專案利

害關係人，包括專案管理者及各個子系統工作站之軟體開發人員等。

三、系統開發技術與工具

本研究採用典型的三層式架構(Three-tier)，將系統部署為展示層、邏輯層與資料層。在展示層方面，使用者可以透過瀏覽器介面和網際網路，經由HTTP的方式遠端登入到系統。在邏輯層部分是以Microsoft Windows XP Professional作為伺服器作業平臺，網站應用容器為Apache Tomcat Server，系統開發則使用Java Server Page (JSP)與JavaBean，而知識庫系統則採用JESS(Java Expert System Shell)做為核心，以進行啟發式的推論。JESS是一種利用Java物件導向程式語言實作而成的規則式專家系統外殼(Rule-based Expert System Shell)，可透過JESS所提供的Java類別庫與Java應用程式相關技術整合運作，具備優良的嵌入能力。其中，JESS語法是以CLISP(Common LISP)語法為基礎所發展。本研究之知識庫實作主要將JESS類別方法內嵌在程式中，透過呼叫知識擷取介面貯存於知識庫的規則，將事實與規則傳入知識庫系統的工作記憶體與推論機進行推論，產生相關的結果。

另一方面，在智慧型代理人實作方面所採用的開發套件為JADE(Java Agent Development Framework)做為多重代理人開發平臺。JADE是符合FIPA協定規格的多重代理人系統的架構或平臺，同樣是以Java程式語言所開發而成，具有物件導向程式設計的，可利用JADE所提供的代理人類別庫，與 Java相關技術做良好的結合與應用，可以有效降低實作多重代理人的複雜度。JADE的代理人平臺並提供圖形化介面工具，可在分散式環境下透過遠端方式進行各種代理人組態、除錯與部署功能，是一個相當適合發展跨平臺智慧型代理人的系統平臺。資料層方面，本研究的關聯式資料庫方面為Microsoft SQL Server 2000，透過JDBC(Java Database Connectivity)技術進行溝通與連接。

四、系統實作之範例

當專案管理者在專案規劃初期進行軟體專案度量作業時，則須先新建專案，設定專案相關的基本資料，如專案代號、專案計劃名稱、專案預估開始日期、結束日期與度量人員群組等，然後將專案分解成多個可獨立作業的子系統計畫，並設定各個子系統所預計採用的開發工具或程式語等。接下來再利用功能點分析法根據各個子系統之需求規格或功能規格進行軟體規模估計，以計算出各個子系統與整體專案之預估功能點數和原始碼行數等。如圖5所示，專案管理者可以維護並瀏覽該專案下各子系統中的功能點總數和預估原始碼行數等資訊。接著，即可進行度量群組成員權重之設定工作，如圖6所示。

圖5：軟體規模基礎化之相關資訊

圖6：度量群組成員權重之設定

當獲得軟體專案預估之相關參數或調準值後，配合本系統所提供各項系統功能(成本、時程、品質與生產力度量)進一步獲得專案相關的度量資訊與決策建議。茲將本系統之相關應用說明如下：

(一)成本度量之應用實例

在專案進行過程中，專案管理者可透過專案成本估計值與實際值之差異，即時觀察整體專案及子專案的成本績效。圖7為專案成本評估與監控資訊，包含專案成本估計值、實際值、差異值與差異度等，系統中的狀態評量代理人會根據以上成本資訊自動推估專案成本狀態。由表中得知整體專案預算執行尚符合預期範圍內，並且該專案下的整合式軟體專案入口網站子專案的開發工作的成本績效最佳，而知識本體描述子專案的預算執行在合理的範圍內，但智慧型代理人子專案的成本績效則呈現嚴重透支情形。

圖7：專案成本評估與監控資訊

(二)時程度量之應用實例

時程度量與成本度量的功能介面類似，將估計與實際時程進行時程績效的度量即可掌握專案的時程狀態。圖8專案時程評估與監控資訊，由表中得知整體專案時程績效尚符合預期，並且該專案下的整合式軟體專案入口網站與知識本體描述子專案的時程績效最佳，但智慧型代理人子專案的開發可能缺乏效率，時程落後情形嚴重。

圖8：專案時程評估與監控資訊

(三)品質度量之應用實例

首先，軟體品質因素權重是由問卷搜集專案相關人員對於專案規劃初期時對軟體品質的預期看法，經過程式中的品質權重公式計算出品質主因素與次因素權重，以進行軟體品質因素權重設定，接下來群組成員便可針對子專案進行品質評分工作，之後由系統

自動進行軟體品質群組彙總，如圖9所示。當完成群組彙總工作後，由於品質群組彙總表僅呈現整體專案與子專案的品質評分，並無法提供專案管理者有意義的資訊。因此，透過系統中的狀態評量代理人進行品質狀態評量，以顯示品質狀態評價資訊。如圖10所示，目前該專案下的整合式軟體專案入口網站子專案之軟體品質狀態是良好的且另人滿意。

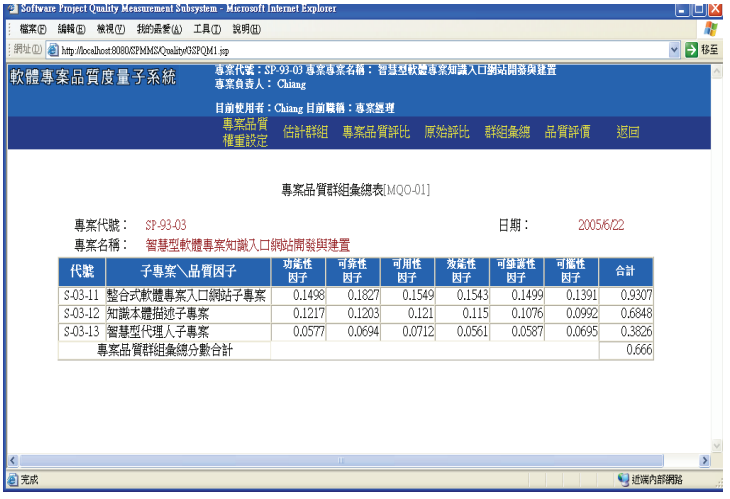

圖9：品質群組彙總

圖10：品質狀態評價資訊

(四)生產力度量之應用實例

在專案開發過程，專案開發人員分別完成原始碼與文件產出工作後，系統便會分別計算整體專案與子專案的軟體生產力。如圖11所示，透過系統中的狀態評量代理人進行

生產力狀態評量，以顯示相關生產力狀態評價資訊。因此，專案管理者可以從中得知目前整體專案與子專案的原始碼與文件生產力情況。

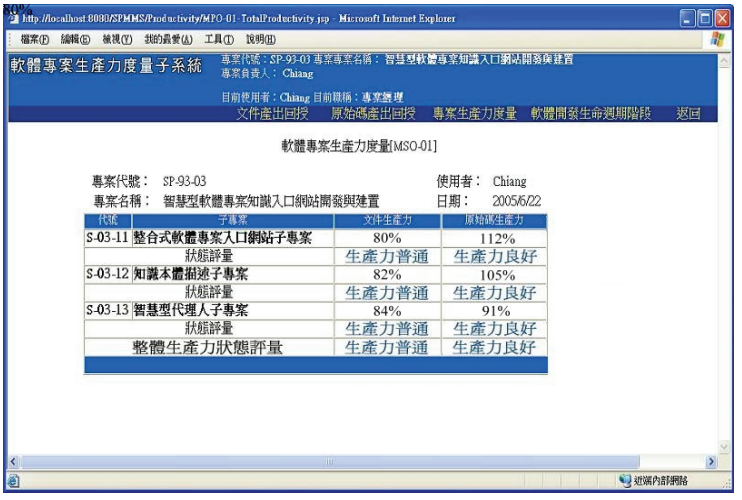

圖11：生產力狀態評量資訊

(五)專案決策建議與預測之應用實例

在完成上述的各項軟體專案度量工作後，即可運用相關的狀態資訊對專案進行決策建議及預測；換言之，當系統中的狀態評量代理人進行相關狀態評量後，即可將相關專案狀態評量資訊，傳遞給決策分析代理人進行專案決策分析與建議，後續再透過訊息代理人將決策建議或預測資訊傳遞給專案管理者，該功能對於專案管理者而言則具有相當程度的助益。

1.專案決策建議

圖12中的軟體專案狀態評量與決策建議對於軟體專案的進行有其重要意涵，其主要以子專案的成本與時程進行綜合評量再搭配品質現況以評估子專案的現況。以畫面中智慧型代理人子專案為例，當專案評量結果為子專案為成本預算超支且時程已落後，以及品質現況為粗劣令人不滿意，其因應對策為應重新檢視整體專案的開發工作或專案管理方法需要改進，並針對品質評價較低之子專案改善。此時，專案管理者可參照此因應方案，進行專案的改善活動。

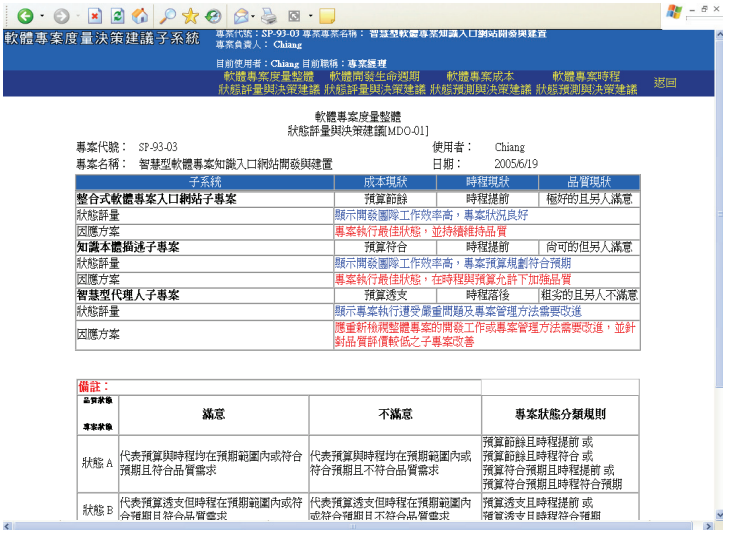

圖12：專案狀態之決策建議

2.專案狀態預測

本系統所蒐集的度量資訊除應用於專案狀態評量與決策建議外，尚可以依據已發生的實際資訊重新預測後續未發生階段與完工後的可能情形。例如：當專案管理者為即時瞭解智慧型代理人子專案的成本後續變化，即可透過專案成本狀態預測功能得知預測資訊，如圖13所示。此外，系統亦提供圖形化的介面呈現出專案預測資訊，以清楚瞭解子專案成本狀態的趨勢，如圖14所示。

圖13：專案成本狀態預測資訊

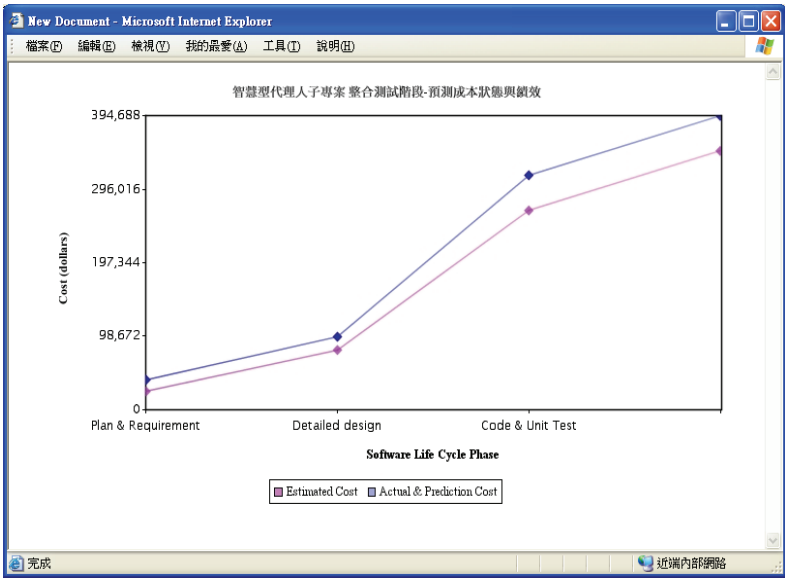

圖14：專案成本狀態預測之累計曲線圖

伍、分析與討論

為分析本研究已開發完成的度量資訊系統之應用成效，本研究係透過訪談方式進行系統效益評估，其訪談對象為個案公司(簡稱A公司)之研發部門人員(一位專案管理者、三位專案開發人員)。茲將效益評估之基本內容彙整如下：(1)A公司已將本系統正式上線使用(從民國94年1月～迄今)；(2)至目前為止，A公司將本系統只應用於「智慧型軟體專案知識入口網站開發與建置」之大型軟體專案計畫；(3)系統使用人員主要為參與該大型軟體專案計畫之專案人員，共計40人，且平均都有8年以上的軟體專案開發經驗；(4)系統執行成本方面，A公司主要計算重點項目包括：委外本研究團隊所開發的度量資訊系統、專案人員之教育訓練課程、硬體設備與其它等成本，約計52萬元。此外，茲將A公司專案人員利用軟體專案度量資訊系統取代傳統人工作業方式進行度量工作以及專案監控活動之導入效益如下：

一、解決分散式開發環境下度量資訊的蒐集與回報之問題

以傳統人工作業進行專案度量資訊蒐集，其過程相當耗時費力，並且缺乏一致性的度量作業流程與標準。因此，往往造成度量資訊本身的時效性、正確性與完整性等難以掌控及確保；而軟體專案度量資訊系統，乃根據本研究度量程序模式所發展的，各項度量議題所進行的步驟及計算方式皆有其明確定義，並可透過網際網路即時蒐集與回報分散式專案開發團隊的重要度量資訊，不但能提升資料蒐集作業的效率與品質，並能節省度量工作時間與人力等成本。

二、專案狀態評量與決策分析

傳統人工作業進行專案狀態評量與決策時，乃根據專案管理者本身的經驗與主觀判斷，因此容易造成誤差之情況發生，同時，當大型專案各種度量議題狀況接續發生時，專案管理者進行專案狀態評量與決策的複雜度提高，使決策時間延遲，導致專案可能中止或擴大化；若使用度量資訊系統進行專案狀態評量，係以量化之數據與相關標準進行相關狀態判斷，以獲得客觀的評量結果，此外，系統便依據專案狀態評量結果進一步分析相關因應方案，作為專案管理者進行決策之參考用途。所有專案狀態評量與決策之工作皆由系統中狀態評量代理人與決策分析代理人程式結合知識庫所完成，能有效解決採用傳統人工進行上述作業之問題。

三、專案狀態之預測

在專案開發過程中欲進行專案狀態預測時，必須先蒐集專案已發生之實際成本與時程資訊，配合原先未發生的預估成本與時程資訊，即可預測專案後續進行的成本與時程狀態。因此，採用傳統人工作業進行大型軟體專案狀態預測，則困難度相當高；若應用度量資訊系統，即可達成上述作業需求，對於專案控管之目的有相當程度的助益。

四、主動提供個人化的專案狀態資訊與決策訊息通知

採用傳統人工作業進行大型專案狀態資訊與決策訊息的溝通與傳遞時，相當耗時費力，並且傳遞的資訊未必符合時效性；若使用度量資訊系統的訊息主動通知機制，透過Web方式或訊息通知代理人程式自動發送電子郵件給個別專案人員，以獲得相關資訊。

根據上述得知，本研究發展的軟體專案度量資訊系統，除了可以完整蒐集專案度量資訊之外，並將度量資訊應用在專案的狀態判斷、決策與預測等工作，對於分散式大型軟體專案開發與管理之績效皆具有極大助益。

有關A公司專案人員應用本系統所遭遇的問題與解決作法之經過情形整理如下：

（一）人員方面

在系統應用初期，由於專案人對於應用本系統進行軟體專案度量之作法感到陌生，以及操作本系統的經驗與熟練度相當欠缺。因此，專案人員不願意改變傳統採用人工之作業方式，並且部份人員有恐懼心理，排斥使用本系統。針對此問題，A公司之專案管理者認為，高階主管應表現強烈企圖心與支持態度，並且不斷的與專案人員進行溝通工作，對於表現良好的專案人員進行公開表揚與獎勵。

（二）作業制度方面

由於先前度量的作業流程普遍是採用個人的經驗法則，因此缺乏標準化制度與流程。針對此問題，A公司之專案管理者與本研究團隊認為，將軟體專案度量程序模式實際應用於專案開發流程，確實能夠將度量工作落實，並且各種軟體專案度量(成本、時程、品質與生產力)之作法，均有標準依據。

（三）財務方面

根據A公司之專案管理者指出，系統應用至一個月後，A公司即發現系統執行成本超出個案公司之預算，以及考量後續系統維護成本需另外支付等問題。因此，A公司之專案管理者認為，財務部門應配合研發部門引進資訊系統的需求，提供一些必要的資金支援。

（四）技術方面

在進行本系統的移轉工作時，本研究團隊發現，透過本系統取代傳統人工作業並不容易。因此，本研究團隊人員建議應加強A公司專案人員的教育訓練、定期評估專案人員的技術能力、應用的初期派遣一些學習力強的人員進行本系統之教育訓練，培養技術移轉工作的種子人員，對未來技術的推廣績效更有助益。

（五）資訊安全方面

A公司對於系統的資訊安全問題相當注著，擔憂本系統遭受駭客入侵及病毒攻擊。因此，採取相關資訊加密機制，以及進行防火牆、防毒軟體安裝與病毒碼更新等工作，以提升資訊安全。

根據本研究之經驗，當企業欲導入一套軟體專案度量資訊系統時，應注意下列幾項實施原則：

1.爭取主管的信任與支持

首先，必須使主管充分瞭解軟體專案度量對於整個專案開發活動及管理工作的重要性與效益，爭取主管的信任與支持是推動系統發展計畫成功與否的主要關鍵所在。

2.鼓勵使用者的參與

如何有效鼓勵使用者的參與，必須透過專案度量制度的運作與激勵措施，從使用者的觀點出發，以提高參與意願。

3.教育訓練工作是導入過程之一項重要活動

使用者對於導入新的軟體工具欲熟練操作，通常需要經歷過一段期間的教育訓練。並且，訓練過程中透過實際問題的操作才能確實瞭解軟體工具的功能。

4.減少使用者的負擔，儘可能使度量作業朝向自動化

在軟體開發過程中，使用者本身已有所需負責之工作任務。因此，為減少使用者的作業負擔與抗拒心態，應使度量作業朝向自動化。例如本系統係結合智慧型代理人程式與知識庫系統，進行相關度量例行性作業(e.g., 專案狀態評量、預測、決策建議與訊息通知等活動)，以降低專案管理者之工作量。

陸、結論與建議

總而言之，本研究的重點主要是建立一個軟體專案度量程序模式，模式的建立係以COCOMO II 估計模式、ISO/IEC 9126-1軟體品質模式及IEEE Std. 1045軟體生產力度量標準為基礎，並以差異分析法與加權評分法進行軟體專案度量，並發展一套Web-based軟體專案度量資訊系統，協助進行大型軟體專案開發過程之度量作業，並結合智慧型代理人程式與知識庫系統將度量資訊進行深度化的分析與判斷，產生一些實用的專案狀態判斷資訊與決策方案，同時，將專案狀態資訊與決策以訊息方式主動通知專案相關人員，提供專案管理者進行專案度量活動時的重要輔助性決策工具。

基本上，本研究所發展的軟體專案度量資訊系統，具有下列特色與優點：(1)本系統係根據度量模式所發展，具有一致性與標準化的專案度量作業程序，可作為軟體專案度量之輔助工具，(2)以Web為基礎的系統操作介面，有效解決分散式大型軟體專案度量活動的困難點與問題，(3)系統結合智慧型代理人程式與知識庫系統，達成專案狀態評量、決策分析與訊息通知等作業自動化之功能，(4)減少人為及主觀因素所造成的誤差，提升度量作業的品質與正確性，(5)減少耗費在度量活動的時間與工作量，(6)專案管理者可以在專案的開發過程中，即時監控專案狀態資訊，以確實掌握專案的現況，(7)專案管理者依據系統所判斷的專案狀態資訊與決策建議，利於專案後續改善活動，(8)可根據已發生的實際資訊預測專案可能之變化，以降低專案開發風險，與(9)專案的重要資料得以保存，以作為日後相關專案開發之重要參考。

茲將本研究之成果彙整如下：(1)本研究係根據軟體專案度量理論，建構出一個軟體專案度量程序模式，其涵蓋成本、時程、品質、與生產力等重要的軟體專案度量議題，(2)度量模式中各項度量步驟均有標準依據與計算方法，可作為實務界建立一套度量作業制度之參考架構，(3)建立一個Web的度量作業環境，專案人員可以透過軟體專案度量資訊系統即時取得專案資訊，並進一步針對專案進行狀態評量、預測與決策建議，作為專案控管之平台，與(4)針對個案公司應用軟體專案度量資訊系統進行分析與探討，將研究發現及成果提供相關研究人員之參考。在研究限制方面：(1)本研究模式主要係以瀑布式開發模式為主，而暫不包含其它軟體開發模式，(2)軟體專案度量議題以為專案管理者關切的核心議題為主，其它如人員專業能力與流程成熟度等不在本研究範圍，(3)有本研究之軟體生產力度量是以生產力=產出÷投入之概念，針對編碼與文件製作活動進行原始碼與文件生產力度量，至於其它專案開發活動之生產力，仍有待後續研究予以完整探討。根據本研究的經驗，茲建議未來研究可朝下列方向進行：(1)透過專案度量歷史資料庫的累積，調準軟體專案度量因子的權重值與相關參數，(2)提升軟體專案度量程序模式的適用性與應用範圍，與(3)探討如何應用軟體專案度量資訊提升專案判斷、預測與決策之能力。

致謝

作者感謝評審委員對於本篇論文提供諸多之寶貴建議，使本文之內容更加嚴謹。本研究承蒙行政院國科會(計劃編號：NSC95-2416-H-020-005)經費補助，謹表達感謝之意。

參考文獻

1. 林信惠，1992，國內軟體生產力影響因素及生產力衡量模式研究，資訊工業策進會研究計劃期末報告。
2. 林信惠、黃明祥、王文良，2002，軟體專案管理，臺北：智勝文化出版社。
3. 黃世禎，2003，國防資訊系統軟體品質度量研究—導入軟體度量於電腦輔助軟體工程工具，國防科技學術合作協調小組合作研究計畫成果報告。
4. 黃世禎，2004，以量化軟體度量指標支援CMMI評鑑之研究，行政院國家科學委員會專題研究計畫成果報告。
5. 黃明祥、盧煥文，2006，『以智慧型代理人為基礎的知識庫系統在軟體專案成本估計與控制之應用』，資訊管理學報，第十三卷，第一期：137~167頁。
6. Albrecht, A. J. and Gaffnet, J. E. “Software Function. Source Lines of Code, and Development Effort Prediction: A Software Science Validation,” *IEEE Transactions on Software Engineering* (9:6) 1983, pp: 639-648.
7. Boehm, B., *Software Engineering Economics*, Prentice Hall, Inc., Englewood Cliffs, N.J., 1981.
8. Boehm, B., Clar, B., Horowitz, E., Westland, C., Madachy, R. and Selby, R. “Cost Models for Future Software Life Cycle Processes: COCOMO 2.0,” *Ann. Software Eng*. (1:1) 1995, pp: 1-30.
9. Boehm, B., Chulani, S. and Clark, B. “Calibrating the COCOMO II Post Architecture Model,” http://sunset.usc.edu/publications/TECHRPTS/1998/usccse98-502/CalPostArch.pdf, 1997.
10. Boehm, B., *Software Cost Estimation with COCOMO II*, Prentice Hall PTR, 2000.
11. Buglione, L. and Abran A. “A Quality Factor for Software,” Third international Multidisciplinary Congress in Quality and Reliability, ENSAM-RUFEREQ, Paris, France, 25-26 Mar., 1999.
12. Callahan, J. and Ramakrishnan, S. “Software Project Management and Measurement on the World-Wide-Web (WWW),” Proceedings of the 5th Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises, 1996, pp: 156-161.
13. Chee, C. L., Jarzabek, S. and Paul, R. “F-Metric: A WWW-Based Framework for Intelligent Formulation and Analysis of Metric Queries,” *Journal of Systems and Software* (43:2) 1999, pp: 119-132.

14. Chen, J.Y. and Su, S.W. "Agent Gateway: A Communication Tool for Multi-Agent Systems," *Information Sciences* (150:3-4) 2003, pp: 153-164.
15. Fenton, N. "Software Measurement: A Necessary Scientific Basis," *IEEE Transactions on Software Engineering* (20:3) 1994, pp: 199-206.
16. Fusani, M. "Quality Models for Software Evolution Instruments," International Seminar on Software Measuring & Testing, IEI-CNR/Qualital/SSSUP, Pisa, Italia, 1995.
17. Gopal, A., Krishnan, M. S., Mukhopadhyay, T. and Goldenson, D. R. "Measurement Programs in Software Development: Determinants of Success," *IEEE Transactions on Software Engineering* (28:9) 2002, pp: 863-875.
18. Hardeep, V. J. "SoftCord: An Intelligent Agent for Coordination in Software Development Projects," *Decision Support Systems* (20:1) 1997, pp: 65-81.
19. IEEE Std. 1045-1992, *IEEE Standard for Software Productivity Metrics*, 1992.
20. ISO/IEC, International Standard 9126: Information Technology – Software product evaluation – Quality Characteristics and Guidelines for Their Use, 1991.
21. Johnson, J. R. "A Working Measure of Productivity," *DATAMATION* (23:2) 1977, pp: 106-110.
22. Jones, C. "Patterns of Large Software Systems Failure and Success," *IEEE Computer* (28:3) 1995, pp: 86-87.
23. Jones, C. "Software Sizing," *IEE Review (45:4)* 1999, pp: 165-167.
24. Kalakota, R. and Whinston, A. B. *Frontiers of Electronic Commerce*, Addison-Wesley Publishing Company, Inc., 1996.
25. Kitchenham, B., Pfleeger, S. L. and Fenton, N., "Towards a Framework for Software Measurement Validation," *IEEE Transactions on Software Engineering* (21:12) 1995, pp: 929-944.
26. Liu, X. F. and Viswanathan, R. "A WWW Based Software Metrics Environment for Software Process Management and Software Product Quality Improvement," The Twenty-Third Annual International Computer Software and Applications Conference, 1999, pp: 301-304.
27. Liu, X., Kane, G. and Bambroo, M. "An Intelligent Early Warning System for Software Quality Improvement and Project Management," Proceedings of the 15th IEEE International Conference on Tools with Artificial Intelligence, 2003, pp: 32-38.
28. Maston, E. j., Barrett, E. B. and Mellichamp, M. J. "Software Development Cost Estimation Using Function Points," *IEEE Transactions on Software Engineering* (20:4) 1994, pp: 275-287.
29. O'Connor, R. and Jenkins, J. "Using Agents for Distributed Software Project Management," IEEE 8th International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, 1999, pp: 54-60.
30. Paul, R. A., Kunii, T. L., Shinagawa, Y. and Khan, M. F. "Software Metrics Knowledge and Databases for Project Management," *IEEE Transactions on Knowledge and Data*

Engineering (11:1) 1999, pp: 255-264.

31. Pfleeger, S. L., Jeffery, R., Curtis, B. and Kitchednham, B. "Status Report on Software Measurement," *IEEE Software* (14:2) 1997, pp: 33-43.
32. Pressman, R. S. *Software Engineering: A Practitioner's Approach*, McGraw-Hill, New York, 2001.
33. Riguzzi, F. "A Survey of Software Metric," *DEIS Technical Report*, No. DEIE-LIA-96-010, 1996.
34. Saad, H. A., "Monitoring Systems and Their Effectiveness for Project Cost Control in Construction," *International Journal of Project Management*, 2003, pp: 145-154.
35. Selby, R. W., Porter, A. A., Schmidt, D. C. and Berney, J. "Metric-Driven Analysis and Feedback Systems for Enabling Empirically Guided Software Development," Proceedings of 13th International Conference on Software Engineering, 1991, pp: 288-298.
36. Selby, R. W. "Amadeus Measurement-Driven Analysis and Feedback System," Proceedings of the DARPA Software Technology Conference, USA, 1992.
37. Sharma, A. and Capretz, M. A. M. "Application Maintenance Using Software Agents," Proceedings of the First IEEE International Workshop on Source Code Analysis and Manipulation, 2001, pp: 55-64.
38. Simmons, B. D. "A Win-Win Metric Based Software Management Approach," *IEEE Transactions on Engineering Management* (39:1) 1992, pp: 32-41.
39. Simmons, B. D. and Wu, C. S. "Software Project Knowledge Base," Proceedings of the Twenty-Fourth annual International Computer Software and Applications Conference, Taipei, Taiwan, 2000, pp: 70-77.
40. Simmons, D. B. "Measuring and Tracking Distributed Software Development Projects," Proceedings of the Ninth IEEE Workshop on Future Trends of Distributed Computing Systems, 2003, pp: 63-69.
41. Sycara, K., Pannu, A., Willamson, M., Zeng, D., and Decker, K. "Distributed Intelligent Agents," *IEEE Expert* (11:6) 1996, pp: 36-46.
42. Vigder, M. R. and Kark, A. W. Software Cost Estimation and Control, *Software Engineering Institute for Information Technology*, 1994.
43. Wu, C. S. and Simmons, B. D. "Software Project Planning Associate (SPPA): A Knowledge-Based Approach for Dynamic Software Project Planning and Tracking," The 24th Annual International Computer Software and Applications Conference, 2000, pp: 305-310.
44. Yan, Y., Kuphal, T. and Bode, J. "Application of Multi-Agent Systems in Project Management," *International Journal of Production Economics*, 2000, pp: 185-197.
45. Zhang, S. and Wang, Y. "An Extension of SEMEST: the Online Software Engineering Measurement Tool," Canadian Conference on Electrical and Computer Engineering 2004 (3) 2004, pp: 1519-1522.