以階層式物件導向結構與狀態推導強化軟體發展專案管理

Enhancing software project management by an Object-oriented hierarchica structure with status propagation

林至中/葉春秀

朝陽科技大學資訊管理系

台中縣霧峰鄉吉峰東路168號

jjlin/s8714602@mail.cyut.edu.tw

摘　要

此篇論文主要是提出一個協助專案管理與控制的物件導向專案管理(OOPM)模型。它是利用物件導向、Petri net，以及由上而下的分解技術，以提供描述軟體發展程序之階層架構的模型。在這個模型中，對於專案活動及其的元件皆有詳細的描述與圖形表示，並且利用物件導向中的階層架構，經由由上而下的分解技術以提供不同階層的專案管理人員專案進度之資訊。這個模型提供了專案活動在執行時的及時資訊，以便讓專案管理人員能夠對活動之排程做適當的調整。在實際的運作上，這個模型的建構能夠充份地提供專案管理人員更容易瞭解及監控整個專案發展的過程。

關鍵字：資訊系統發展，專案管理，物件導向，Petri-net，由上而下的分解技術。

1.相關研究的探討

在過去對於專案管理與控制的模型著重在專案活動時間排程的控制上有三種:Gantt,CPM,以及 PERT[1-3]。但這三種模型擁有以下的缺點：(1)他們沒有提供專案理者可瞭解的資訊及訂立專案活動的程序；(2)他們沒有呈現活動適當的階層架構以及他們所包含的子活動；(3)他們沒有指出一個活動的再執行而對於活動重新排程的情況；以及(4)他們沒有提供一個活動開始執行所需要被觸發的條件。

對於以上傳統模型所提出之缺點，使 Petri net[4-6]以及 UCLA 圖型模型[7]來提供專案活動的瞭解度以及訂立專案的活動進度。一般來說，採用由上而下的分解技術[8,9]可以適當的表現出活動與子活動之間的階層架構。然而，這些模型仍然有幾項不足的地方，像是(1)他們沒有提供出一個活動開始的條件及活動執行的狀況；以及(2)他們沒辦法區別及表示在專案活動中，不同程序的種類其元件存取如何（如資源，生產，以及狀態報告和活動之間可能的階層架構之關聯。所以，DesignNet 以及 PM-Net 模型[10,11]被發展出來彌補以上的缺點，他們依循 Petri Net、UCLA 以及 AND-OR[12]的分析模型來表現出活動與子活動之間的階層架構以及活動之間的相依性和他們存取的程序元件。尤其是，PM-net 利用 DFD 由上而下技術[13,14]的特性，將活動由上而下的分解出相關的子活動直到最底層。最底層的執行狀況由下而上的將活動存取的元件及活動進行的資訊往上傳到上層。故，不同階層的專案經理可以針對想要瞭解不同階層的相關資訊去進行詢問及監督專案進度的相關細節。一般而言，這兩個模型所代表的是有效率的進行專案管理與控制。然而，在實際的應用上，這兩個模型仍有不足之處：(1) DesignNet 只提供一個種類的專案進度資料，不同於 PM-net 實際是利用 DFD 由上而下的技術去提供專案管理人員在不同的階層有關專案進度的相關資訊；相對的，PM-Net 僅呈現出活動/元件間的一般的 aggregation 的架構，不同於 DesignNet 實際是利用 AND-OR 邏輯觀念去代表這些進行的活動/元件的 classification/aggregation 的架構[15]；(2)他們都沒有指出活動執行及時的資訊以及從底層到最高層的累積資訊，以致專案經理很難去訂立活動的排程，就如同傳統模型所強調的；以及(3)他們沒有提供對於活動存取不同種類程序元件一個較正式的定義；對於不同存取的種類使用相同的表示法（如：對於一個 place 的輸出所造成的 token 的更新或是產生一個 token 進入 place）；這使得專案經理很難對於發展的程序去做充份瞭解及監

督。

我們所提出 OOPM 的目標就是要補足以上的缺點以及利用物件導向的特性去定義 DesignNet 和 PM-net 模型。在模型中，物件型態可以很明確的呈現專案活動以及他們存取程序的元件。由上而下的分解技術是利用物件型態的階層架構，為了提供專案經理在不同階層所呈現的專案執行的資訊，如同PM-net。物件繼承與包含的特性是使用物件型態的架構來表現出不同階層之間 classification 及 aggregation 的關係，如同 DesignNet。因此這個模型提供對於每個階層的活動/元件的抽象表示；專案管理人員可以詢問及監督他們想要知道的階層執行的狀況及進度的相關細節。另外，這個模型可以及時的提供活動執行的相關資訊以及將最底層的資訊往上反映給上層的活動；這可以讓專案管理人員去訂定每個階層活動的時間排程 最後，這個模型利用及延伸 Petri net 的符號去提供活動/元件一個表示的定義以及他們之間的關聯。這可以讓專案經理很容易的去瞭解及監督發展的程序。

2. 介 紹

為了能夠成功地完成專案活動的發展，適當的對資訊系統監督與控制是主要的關鍵。為了達到上述目標，適當的模型對於描述專案發展的程序已經有很好的影響，並可以提供管理者更多必要性的資訊，讓專案管理者可以檢視其活動的狀況及專案的進度。雖然已經有存在一些不完整的模型來具備一些特性來提供專案管理 做有效率的管理，我們仍將認為一個好的模型應該具有下列的特性：

(1) 一個模型應該能夠提供不同階層的管理者在適當的階層上提供他們相關的專案進度，以達到管理的目的。

(2) 一個模型應該能夠呈現專案執行的進度以及專案是否在訂立的時間排程內完成其活動。

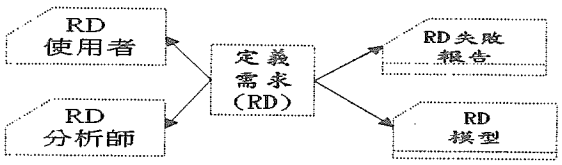
(3) 一個模型應該能夠彈性的支援軟體發展程序之動態的改變(如新增、刪除、重覆其專案活動)，以便指出在設計程序上正確的發展。

(4) 一個模型應該指出在發展程序上任何動態改變的影響(如重新執行、中斷或放棄活動等)。

(5) 一個模型應該能夠充份的讓專案管理者更容易去瞭解及監督其發展的程序。

為了讓這些觀點更具效用，我們採用了物件導向關鍵性的觀念，像是繼承(generalization -specialization)，包含(whole-part)，以及物件明確的位置；物件導向本身的特性可以更容易的反應以上的特性。為了證明此項觀點，我們在本文中提出一個物件模型，稱為OOPM，來提供對於軟體發展程序發展的管理。

OOPM模型提供在不同階層上，專案進度的一些詳細的資訊以及狀態的管理。利用物件導向的特性，類別(如圖1)與物件的使用(如圖2)，再加上利用 Petri-net 符號，以及由上而下的分解技術，提供一個明確的軟體發展程序的階層架構。在模型中，專案活動之物件型態以及存取程序的元件(如資源，生產和狀態報告)在本文中皆有圖形表示；活動的行為描述在物件型態的操作上。由上而下的分解技術是利用物件型態的階層架構來提供有關活動進度的資訊與下層所執行的狀況往上的呈現，表現不同階層的細節狀態。在物件導向的技術上，物件型態的繼承與包含的架構，來表示其 classification 以及 aggregation 相關的活動/元件的型態，呈現出不同的階層架構 這可以讓模型表現出每個階層架構活動/元件的抽象表示；專案管理者可圖 1



類別圖

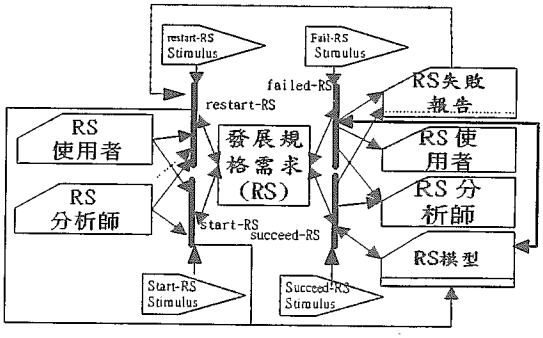


圖 2 物件圖

以詢問及監督他們想要知道的專案執行進度及狀況之相關細節。最重要的是，我們簡化了分析的圖形，凡是有子活動的上層活動只表示出物件之間的關聯性來表示其活動的進行，而不用其 Petri Net 複雜的分析過程。這是因為上一層活動是反應出下一層的活動的執行狀況，故我們捨棄了對上層複雜的表示，而改用簡易的關係圖來表示其活動的進行，這不但可以簡化分析的活動，更可呈現出較清楚的架構。對於這些實際的應用，模型的建構是可以支持軟體發展程序的階層架構以及從下層的執行狀況往上累聚至上層之專案進度的相關資訊。

3. 案例說明

這個章節我們將以一個案例來說明 OOPM 的理論架構，OOPM採用Petri-Net以及由上而下的設計理念來做流程的分析；使用 Petri-Net 來表示出每個活動的執行情況，並以由上而下的分解技術的設計理念來簡化其分析的架構，我們最後以系統發展過程中的分析需求階段來做案例的說明(有關 OOPM 相關的正式定義請參照[16])。OOPM 包括了一個類別圖和一個物件圖，在介紹這兩個圖之前，我們將對此二圖做一簡單的定義說明(如圖 3 及圖 4)。在圖 3 中，我們以虛線的圖形來表現類別的定義，包含了對活動、資源及(成功與失敗)報告的類別表示。至於實線箭頭則表示活動與資源/報告之間的關係，粗線箭頭表示活動之間執行的順序；圖 4 是OOPM 對於專案所使用到的(活動、資源、報告)物件所做的描述，其中活動的事件 (|)，相當於 Petri-Net 定義中的 transition；而改變物件的狀態 (<->)，所指的是活動的事件發生後，對相關物件內容做的結果更新；而物件使用，但不消耗 (-->)，所代表的是在情況允許之下，物件可同時提供二個或二個以上活動事件的發生所使用。

圖 5 及圖 6，顯示了定義分析需求(以下簡稱 RA)活動及其相關資源/報告的類別與各類別之間的關係。從圖 5 中，可看出活動及其組成的子活動之間的階層關係，也可以瞭解各(子)活動與其相關的資源/報告之間的互動情形。至於從圖 6 中，則可以瞭解各類別在不同階層所形成的分類/組成架構。因此，從第一層的類別對應到圖 5 中的第一層類別，我們可以清楚知道 RA 動作所需要的資源。及其執行時將會產生的文件(成功或失敗)；同樣的，圖 6 中的第二層與第三層的類別為對應到圖 5 中的第二層與第三層的類別；特別在圖 6 中，RA 失敗報告有一 DS (Disjoint Subtype)的表示[16]，其所代表的意思是 RA 失敗報告的產生只有可能是因為 RD 失敗報告或是 RS 失敗報告或是 SUM 失敗報告的產生而造成。這主要是回應一個真實的情況；只要某一組成的子活動失敗，則上一層中的活動也進入失敗狀況，而產生一個失敗報告。在圖 5 與圖 6 的例子中，利用三層的類別定義來表達分析需求(以下簡稱 RA)這樣的活動：第一層為 RA；第二層為完成 RA 所需進行的子活動，包含：

發展定義需求(以下簡稱 RD)，發展規格需求(以下簡稱 RS)，發展使用者手冊(以下簡稱 SUM)；第三層為完成 RD 所需進行的子活動，分為發展 Context 圖形(以下簡稱 SCD)，發展 USE-CASE 模型(以下簡稱 SUCM)，以及發展介面模型(以下簡稱 SIM)。而在第二層 RD 所完成的報告 (成功的報告) 之輸出粗線所代表的是必須等 RD 之動作完成之後才能開始進行 RS 以及 SUM 的活動；也就是說，必須等到 RD 在第三層的子活動(SCD 圖形、SUCM 模型、SIM 模型)完成後，才能進行第二層的後續活動。由此表示出其活動執行的優先順序。

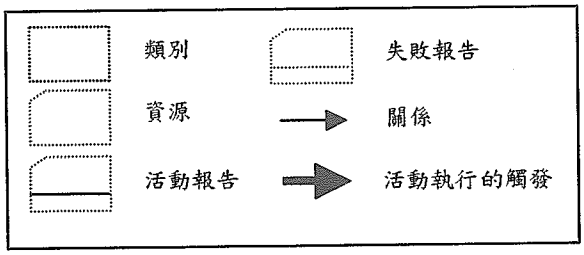


圖 3 OOPM 模型的類別圖型符號定義

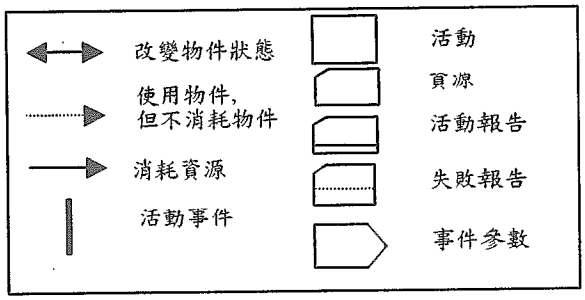


圖 4 OOPM 模型的物件圖型符號定義

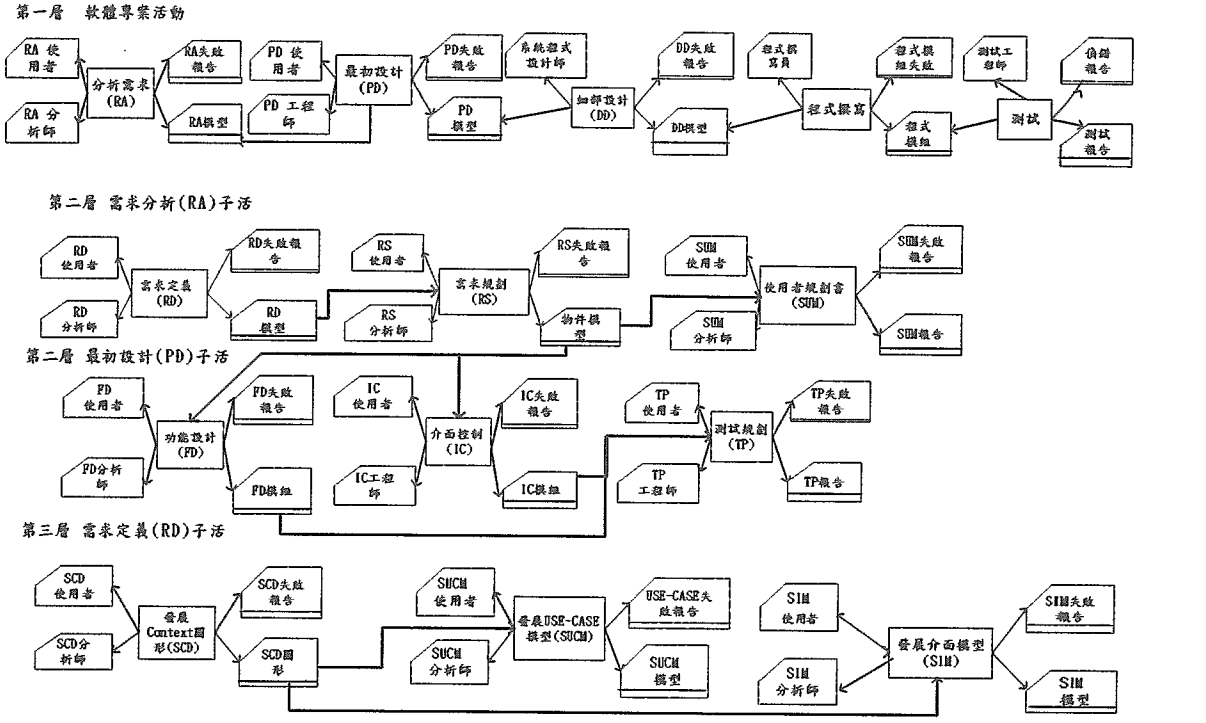


圖 5 分析需求/初步設計的類別圖

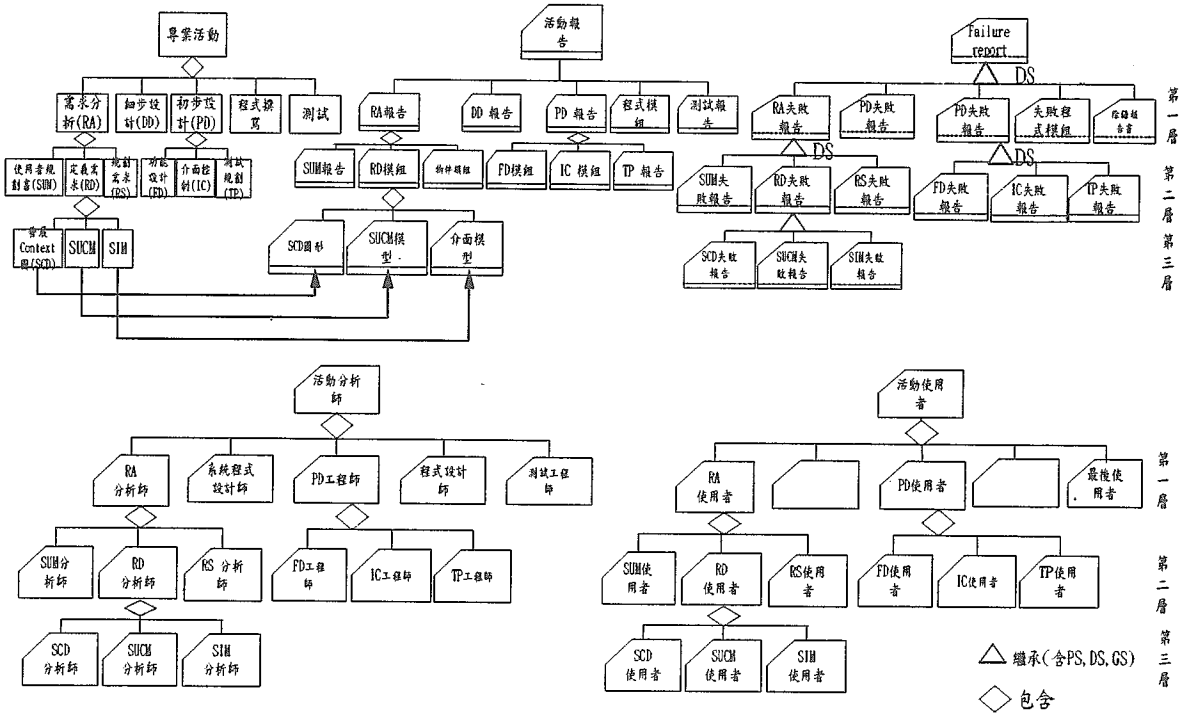


圖 7 物件圖是根據圖 5 類別圖的定義而產生的 RA 物件

圖。比較特別的是，在圖 7 的最後一層中，我們使用了 Petri-net 圖形的方式來表達所有活動可能會發生的事件與其它相關物件的存取關係。至於其他階層，由於上一層的活動執行結果完全是由下一層的子活動的執行狀況來決定，所以上一層並不需要使用 Petri-Net 圖形來表示其活動之事件進行，而只需呈現出目前活動由下一層所推導出來的執行狀態。因此，如果上一層的活動是由許多子活動所組成，我們就只將此上一層的活動的相關物件畫出。

依據在圖 7 的最後一層中所顯示，要完成 SCD 活動必須先將所需之資源：SCD 使用者以及 SCD 分析師找齊之後，才能啟動 SCD 的動作（start-SCD 事件的發生），此一活動被啟動後就會產生一個 SCD 圖形物件來等待成功訊息的傳達；若活動成功完成，則此一 SCD 圖形的狀態即改為成功的狀態(Succeed-SCD 事件的發生)。若是執行結果為失敗的動作，則會觸發 failed-SCD 來產生一個 SCD 失敗報告。專案管理人員可依據此失敗報告檢討其失敗原因，並重新再觸發執行的動作(restart-SCD)。

若以第一層的 RA 而言，要完成此項動作所需要的資源分別是 RA 使用者以及 RA 分析師;若執行此 RA 後，可能產生的報告包括失敗報告或是活動執行成功的 RA 報告。而在執行完成後，所再產生的 RA 使用者以及 RA 分析師代表是活動執行後，資源可重新的再被使用 甚至一個使用者或是分析師可能會同時扮演兩種不同的角色。

專案計劃的進行可包括資源(Resource)、活動(Activity)、及報表(Report)等相關物件所組成，這些物件本身所可能發生的相關狀態，將於下面做詳細的說明與定義。

活動會發生有下列幾種狀態：

1. 活動尚未進行
2. 活動進行中
3. 活動因某些原因暫時中斷(如專案人員離職)
4. 活動執行失敗
5. 活動已完成
6. 活動已完成，但已超過排程時間，可能造成時間的延遲

在不同層次之間的狀態推導，可藉由其相關活動來取得，以圖七為例說明如下：

1. 第二層 RD 活動的完成是由第三層 SCD、SUCM 與 SIM 活動的進行完成（包含關係），只要其中有任何一個活動尚未完成，則不能視 RD 完成。

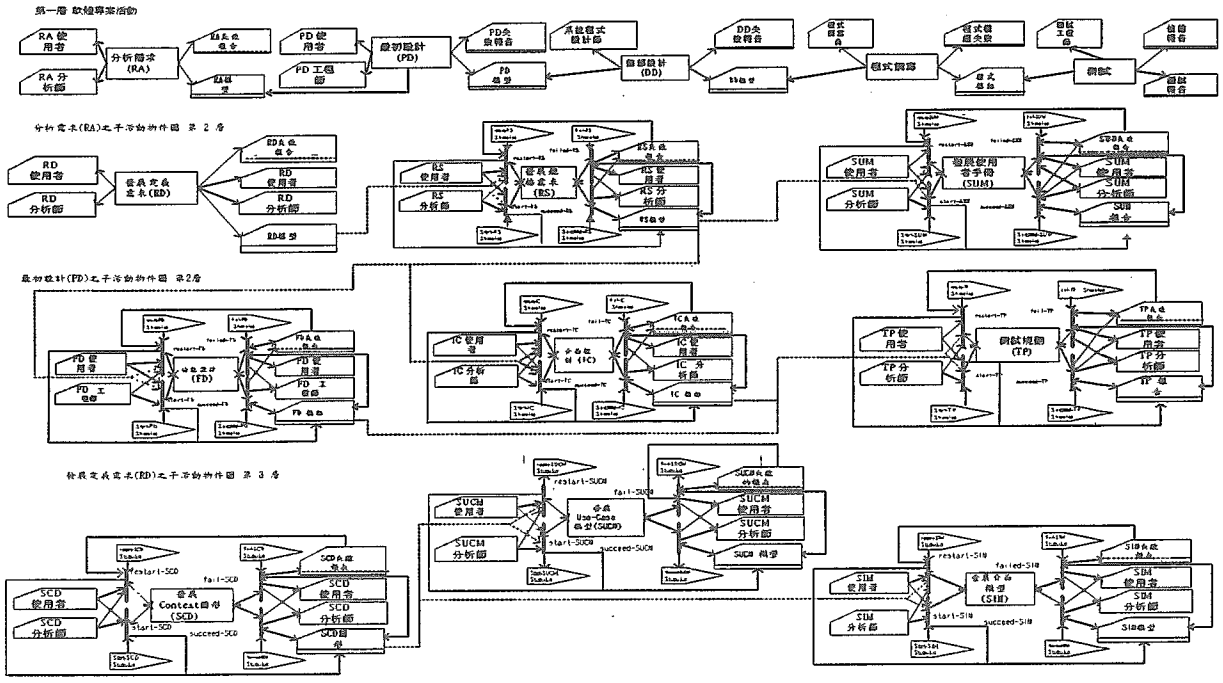


圖 7 分析需求/初步設計 物件圖

2. 同理在第三層上，若所有活動都尚未開始進行，則第二層的 RD 將會呈現活動尚未進行的狀況。
3. 在第三層上，若有任何一個活動已在進行中，則第二層的 RD 活動將會顯示目前活動正在進行的狀態。
4. 在第三層上，若有任何一個活動失敗，則第二層的 RD 會立即反應出失敗的狀態。
5. 在第三層上，若三個活動進行中有任何一個活動在執行的過程中，因某些因素而造成活動的暫時中斷進行（如專案人員的離職），連帶的反應到上層第二層 RD 活動進行也會造成暫時無法繼續執行。
6. 若第三層上，三個活動皆已完成則表示第二層 RD 的活動亦完成（已完成且通過審核）；但是，若完成的時間已超過原先排程上的預計時間，則第二層 RD 的活動表示會呈現完成，但會展現延遲的狀況，專案人員可依此對活動進行時程的調整。

資源(Resource)的狀態有幾種表示方法：

1. 資源之準備尚未進行(如尚未進行使用者或分析師的尋找)
2. 資源之準備已在進行中(如使用者或分析師已進行尋找相關的資源)
3. 資源失敗（如找不到所需的資源）
4. 資源之已完成，且已通過審核

在資源(Resource)的表示上，會有下列的情形發生：

1. 在第二層 RD 之進行在使用者(RD 使用者)方面所需要的包括：SCD 使用者、SUCM 使用者及 SIM 使用者等三位使用者。以 RD 分析師來說包括：SCD 分析師、SUCM 分析師及 SIM 分析師。若所有所需的使用者及分析師都已準備完成，則 RD 使用者及 RD 分析師皆會呈現完成的狀態（包含關係）。
2. 使用者及分析師在不同的時候所扮演的角色不同，而在同一時候不同分析師的角色可能由一個分析師來同時扮演，相對的使用者也一樣，端視系統的需求而定。
3. 以第二層的 RD 來做解說：RD 需要 RD 使用者及 RD 分析師，當此資源已全部準備好，則需求定義之相關活動即可進行，且進行中 RD 使用者 及 RD 分析師將會專注該活動進行，但在活動執行結束後，RD 使用者及 RD 分析師會恢復成為原先的資源，所代表的是此資源(使用者與分析師)可再被其他的活動所使用，當然所扮演的角色會因活動的不同而有所不同。
4. 以第二層的 RD 為例，若 RD 使用者中任一個所需的使用者已開始進行準備，所以 RD 使用者狀況顯示會由原先的尚未進行轉變成已在進行。

報表(Report)的關係（成功報表或失敗的報表）有幾種表示方法：

1. 尚未進行任何的報表相關的活動被執行，而觸發物件的產生(正在被產生)
2. 報表之進行卻因某些原因暫時中斷
3. 報表已完成，尚未審查
4. 報表已完成，但未通過審核
5. 報表已完成，且已通過審核

在報表表示上，DS 有以下的狀況產生：

1. 在第二層上，若報表已完成，且通過審查，則狀況顯示會轉變成已完成且審查通過；若已完成，但因不符合需求而未通過審查，則狀況會轉變成已完成但未通過審查。
2. 在活動執行過程中，因某些因素而需暫停執行，故報表之產生也因此中斷，故會由正在被產生轉變成暫時中斷。

4. OOPM 的貢獻

OOPM 模型定義了 DesignNet 以及 PM-net 的特性，並延伸了他們在軟體發展程序上的優點：

(1) OOPM 利用了物件導向的技術，在發展的程序上對於動態的改變能夠更具彈性；並利用物件導向的特性，對於一個發展的程序能夠更容易去瞭解及維護。

(2) 以 PM-Net 來說，OOPM 利用了由上而下的分解技術去提供專案經理在不同的階層上有關專案進度的資訊；以 DesignNet 來說，OOPM 使用了繼承與包含的圖形來表示出不同的階層架構。因此，OOPM 加強了 DesignNet 與 PM-net 的特性，並提出在每個階層對活動/元件適當的表示方法。

(3) 相較於 DesignNet 以及 PM-net，OOPM 可以指出活動執行時的相關資訊以及將活動執行的資訊由下往上傳達至最上層。這可以使專案管理者及時地安排每個階層的活動排程。

(4) DesignNet 以及 PM-net，對於不同的活動種類存取程序元件有不同的表示法（如同：一個輸出的節點到 place 可能意謂著一個 place 中 token 的更新或是產生一個新的 token 進入 place 中）。相對地，OOPM 利用以及延伸了 Petri net 符號，提供了對於活動/元件較正式的定義以及他們的相依性。這可以讓專案管理者很容易去瞭解及監督程序的發展。

5. 結論

我們將物件導向模型應用在專案的管理與控制上。這是利用了物件導向的特性，來定義以及加強 DesignNet 以及 PM-net 模型，以提供軟體發展程序一個階層的表示。這幾個好處幫助專案管理者去瞭解以及對程序的描述，更重要的是去獲得及監督他們所想要知道的程序狀況及進度。

物件導向已經是一個很重要的觀念，並已經愈來愈受到電腦軟體工業的注意。物件導向技術的特性可以讓軟體系統很容易被瞭解、維護以及再使用。一個軟體發展程序是一個設計的程序，並且是一個展開的程序，對於專案模型與控制上，使用物件導向的技術相信是一個合理的方法。我們的模型也致力於在這項工作的表現。現在一個專案管理的環境都建立在較完整的模型上，這也包含了符合本文在軟體發展程序的設計上圖形的表示，本文的設計將會利用 embedded 資料庫以及軟體的開發來存取資料以及活動的物件。透過應用程式的實作可以幫助專案管理者在軟體發展程序的敘述以及監督，由於本研究尚在進行中，至於實作的部份將在未來的文章中向各位報告。

最後，本論文之發展是經由為國科會專案計畫(NSC 88-2213-E-324-006）所支援，

參考文獻

[1] E. Davis, Ed., *Project Management: Techniques, Applications, and Managerial Issues*, Industrial Engineering and Management Press, 1983.

[2] E. Horowitz and S. Sahni, *Fundamentals of Data Structures in Pascal*, Computer Science Press, 1985.

[3] J. Wiest and F. Levy, *A Management Guide to PERT/CPM*, Prentice-Hall, 1977.

[4] J. Peterson, "Petri Nets", ACM Computer Surveys, Vol.9,No.3,Sep. 1977, pp.223-252.

[5] J. Peterson, *Petri Net Theory and The Modeling of Systems*, Prentice-Hall, 1981.

[6] E. Yiannis and L. Thomas, "Specification and Analysis of Parallel/Distributed Software and Systems by Petri Nets with Transition Enabling Function", *IEEE Transaction on Software Engineering*, Vol.18,No.3,March 1992, pp.252-261.

[7] V. Cerf, "Multiprocessors, Semaphores and A Graph Model of Computation", *Ph.D. Dissertation*, Computer Science Department, UCLA, 1972.

[8] L. Belady and C. Evangelisti, "System Partitioning and Its Measure", *The Journal of Systems and Software*,Vol.2, 1981,pp.23-29.

[9] Y. Wand and R. Weber, "A Model of Systems Decomposition", *Proc. Of Int'l Conf. on Information Systems*, Boston, Dec.4-6, 1989,pp.41-51.

[10] L.Liu and E. Horowitz, "A Formal Model for Software Project Management", *IEEE Transaction on Software Engineering*, Vol. 15,No.10,Oct.1989, pp.1280-1293.

[11] K.Lee and I.Lu, "PM -Net: A Software Project Management Representation Model", *The Journal of Information and Software Technology*, Vol.36, No.5,1994,pp.295-308.

[12] P.Winston, *Artificial Intelligence*, 2nd Ed., Addision-Wesley,1984.

[13] T.DeMarco, *Structured Analysis and System Specification*, Yourdon Press, 1978.

[14] C.Gane and T.Sarson, *Structured Systems Analysis: Tools and Techniques*, Prentice -Hall, 1976.

[15] P.Coad and E.Yourdon, *Object-Oriented Analysis*, Prentice-Hall, 1991.

[16] J.Lin and C.Yeh, "An object-oriented Formal Model for Software Project Management",Proc. Of IEEE 6th

Asia-Pacific Software Engineering Conference (APSEC), Takamatsu, Japan, Dec.1999.