1. 簡介

近年來，國內資訊產業一直是我國外貿主力所示，但是在硬體方面，因產業硬體技術門檻的降低，國外大廠的競爭以及大陸急起直追，產品價值愈來愈低落，使得獲利已經大不如前。過去國內產業發展主要投資於低附加價值的硬體電子產業。但是，在具有高附加價值軟體與服務方面，國內投資比例為 20%，遠低於美國 65%及日本 55%。

根據研究顯示結果，未來資訊產業中軟體發展是未來的趨勢。軟體及相關服務產品及資訊家電(IA)比例也不斷的成長。但是，軟體長久以來較少在國內受注目。近年來，由於印度、大陸等國的競爭，若不加強國內軟體競爭力，國內資訊產業將會失去主導優勢。

面對軟體需求的不斷增加，所帶來的相對問題，除了需要豐富且充足的人力資源外，另外必須考慮的是軟體是否具有品質？軟體品質的好壞，絕大部分取決於軟體開發過程。在世界趨勢中，目前軟體品質的標準是以美國卡內基美隆大學的軟體工程學院(SEI)訂立的能力成熟度模型 Capability Maturity Model(CMM)為評量指標。目前全球約有 700 餘個公司及組織的單位通過 CMM 認証。其中最高的四、五兩級，全球各自有 73 與 67 個單位獲得，多數集中在美國及印度。中國大陸也相當積極。2000 年中國國務院發佈「鼓勵軟體產業和積體電路產業發展的若干政策」中，明文規定對軟體出口型企業 CMM 認證費用予以適當支援。其大陸國內若干企業也已有獲得 CMMI 最高等級五的認證。

過去，國內軟體產業普遍被視為艱苦行業。廠商規模偏小，且經營及獲利不易，大部份套裝軟體業者均以發展中文相關處理工具(如倚天中文系統)及以文化區隔為藩籬的產品(如華康字形、貿易財會等本土化商用軟體)爭取有限之內需市場，缺乏國際級旗艦產品，產品外銷亦鮮少，而系統整合業者及專業服務業者除金融業軟體因國內自行開發較早，尚可與國外廠商一較長短外，其餘皆只能在國外廠商爭食之夾縫中，以低廉之價錢承接技術層次較低之工作。

為提升整體產業產值、產業規模、產品發展能力、研發投入規模與實力、外銷能力、大型資訊系統承接能力則必須要有國際標準之引導與認證或是軟體流程作業模型準則，如 ISO 9000 系列、CMMI 等等，才可取得國內外之軟硬體廠商之信任，進一步與國際級開發廠商相互匹敵。

我們試著利用管理學上的 **SWOT**(**S**trength, **W**eakness, **O**pportunity and **T**hreat)分析，分析台灣軟體產業外在環境的機會與威脅；及內部的優勢與劣勢。我們可以得知，由於台灣軟體開發的可再使用性很低，因此，台灣軟體產業如要進軍

國際，則必須要有國際標準輔助、相關軟體工程方法以及相關工具之導入，組織團體才具其競爭力。因此，軟體工程的觀念在學、業界愈來愈受到重視，許多軟體開發流程技術如雨後春筍般的被大量提出，例如 **RUP**(Rational Unified Process)、**OPEN**(Object-orient Process Environment and Notation)、**XP**(eXtreme Process)、**PRIME(**Toward Process-Integrated Modeling Environments**)**等等。

表 1 軟體產業環境 SWOT 之分析

優勢(Strength) ■ 新技術的接受度高。 ■ 政府將軟體列為策略性發展之產業，並予以各種輔導、支援計畫。 ■ 資本市場發達，利於籌措資金，不需仰賴外資。 ■ 勇於創新，創造自我品牌。	**機會(Opportunity)** ■ 民間具有強烈的創新精神。 ■ 科技專案計畫已投入大量資源。 ■ 大量的電子數位產品輸出。 ■ 國外軟體委外訂單的釋出。
弱勢(Weakness) ■ 軟體開發可再使用性低。 ■ 高額的人力成本。 ■ 國內市場規模小。 ■ 缺乏世界級領導性產品。 ■ 軟體開發組織規模小。 ■ 客觀軟體評量標準未被採用。 ■ 土法煉鋼的軟體開發。	**威脅(Threat)** ■ 大陸、印度及韓國軟體產業的快速成長。 ■ 高階研發上的人才仍顯不足。 ■ 國內其他產業外移，內在需求逐漸縮小。 ■ 世界投資環境惡化，企業領導人投資意願降低。

每一種開發流程都具有其特色及適合的專案大小類型，而許多對於軟體品質的衡量，也逐漸採用相關標準所規範的準則，作為評量的依據。但是，不論軟體開發單位欲導入的是何種開發流程技術，其中對於開發流程技術的繁雜規則、條例以及限制，卻往往讓企業望之卻步。深究其原因，軟體開發人員缺乏經驗，同時缺乏相關方法、技術、工具與流程控制，讓企業不知從何入門，是最大之隱憂。因此，如何在引入軟體流程作業模型準則時，融入其中以畢其功於一役，並整合組織資源，發揮組織內部最大效益，是軟體工程導入一大重點。相關電腦輔助工具(CASE Tool)則是另一個導入重點。雖然軟體工程方面的應用工具更是多不可數，每一種工具或方法論往往只著重支援整體軟體開發程序的某部份，無法涵蓋整體軟體發展過程，例如 Rational Rose 只著重於設計階段之輔助開發。因此，統整各類型工具並使其相互溝通利用成為系統開發環境中是在導入軟體工程中的一大挑戰。

為能使軟體開發整體效益有效地提昇，陸陸續續有多種標準化的模型

(Models)與技術(Technologies)被提出以解決各階段的問題，例如，物件導向分析與設計觀念之於軟體架構分析與規劃；標準化模型語言(UML)之於系統整體架構、設計、狀態控制、與測試分析；設計樣板(Design Patterns)之於問題定位與有效設計模式的導入；元件再使用(Reusable Components)之於回收應用高效益高認同之軟體與資料模組或片段；標準化的導入之於軟體作業流程準則等等。它們試圖在所針對的領域中提出整合性的標準化，也達到了相當的成果。

然而，綜觀整個軟體開發流程的各個階段與並檢討其間的連貫關係，各階段發展的標準、方法步驟以及應用工具，卻因為缺乏彼此之間協調與整合的考量機制存在，致使軟體開發者在每一階段內部，必須以人工方式檢視並協調整合來自不同標準、方法及應用工具之來源資料。也因不同設計之間的統合、細部分工等作業程序，使得軟體開發與研究者無法以一個一致性、協同性的開發方式進行軟體系統的開發作業，而使得整個軟體開發成本居高不下。

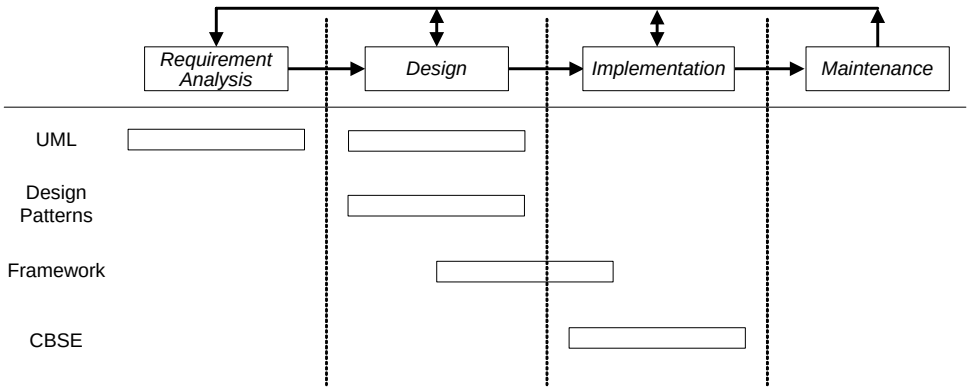

圖 1、The Relationship of Some Software Paradigms and the Software Life Cycle

本計劃延續之前的研究體系與成果，成功地架構了一個以 XML 為基礎的正規化整合式軟體系統文件模型(Formal Model for System Document Integration, **FM/SDI**)，它可被用來協調、整合一般軟體系統規範與系統發展過程中，由一群不同團隊，於各個不同開發歷程按照任務需求，依循不同軟體標準所完成的軟體系統子模組。以這個整合模型為基礎，我們將得以於後續正式的應用性研究計畫中，並期待透過更多的實案整合與應用，以 FM/SDI 這個整合式系統模型的延伸設計，透過 XML 的結構與連結描述特性，串連大部分業界實際於開發過程中設計出的抽象與實做模型，而達成一個正規化、整合性、應用性、且具備延伸性的軟體系統文件規範，期以協助解決長久以來存在軟體系統整合發展與維護作業上的問題。

2. 相關技術研究

物件導向技術及 UML

物件導向是軟體工程領域的一項重要的技術，物件導向的觀點在於將現實世界人、事、物等許多概念的靜態資料及動態行為表示成為物件，而這些概念之間的互動也會對映至物件之間的交互關係。一個物件往往被定義成為一個角色，擁有各自的特徵與行為，而所謂的商業模型(Business Model)就是將某個領域的資訊定義成個別的物件。利用物件導向技術可以使軟體的開發更快、成本更低、及得到較高的品質[5]。物件導向分析與設計(OOA/D) [1]基礎於物件導向的概念成為了軟體開發方法的主流。

Unified Modeling Language (UML)[3]是用來將一個軟體系統描述成視覺化軟體模型的標準表示法。UML 提供了許多既定的圖示用來描述軟體開發過程不同階段的抽象概念，例如使用案例圖(Use case diagram)、類別圖(Class diagram)、合作圖(Collaboration diagram)等等。利用這些圖示法可以讓使用者很容易的了解軟體的分析與設計過程。也因此 UML 已成為物件導向軟體開發的一項重要的技術。但 UML 的使用往往過於彈性而缺乏整合性，不同的 diagram 有不同的適用範圍，但不同的 diagram 卻缺乏一個串連的機制，在系統分析規劃時一旦需要跨不同 diagram，就有賴程式設計師以人工方式轉換、驗證，導致需求分析與系統設計的模型容易產生不完整及不一致性。

XML

XML [6]是由 World Wide Web Consortium (W3C)所制定的標準，一種延伸式標記語言，具有擴展性(Extensibility)、結構性(Structure)、描述性(Description)、確認性(Validation)等特性。同時 XML 具有跨平台的功能，對於不同的作業系統、硬體設備、應用軟體、多元的輸入模式，開發者可以自行制定符合己身需求的標記(tag)，做結構性的描述，促使相同的一份文件呈現不同的規格，適用於不同的軟體，符合不同的設備、滿足多重的輸入方式。

XUM (XML-based Unified Model)

XUM[7]是一個利用 XML 為基礎，將軟體生命週期中，不同階段產出物加以串連、整合，XUM 印證了不同型態的軟體模型、程式碼能有效的加以整合成為

具相同表示格式，卻又不失各軟體模型原本特質，並且透過 XML 的特性，可以呈現出多重不同樣式。本計畫主要就是以 XUM 的研究為基礎，進一步將系統文件加以正規化、整合成為一個一致的模式，並具有支援後續相關處理的表現模型。

3. 正規化整合式軟體系統文件模型

3.1 模型系統說明

透過 XML-based 系統文件整合架構，我們實現將系統文件以一致的表示方法呈現，透過 XML Schema 的規範，我們訂定了一個正規化整合式軟體系統文件模型(FM/SDI)結構如下圖所示：

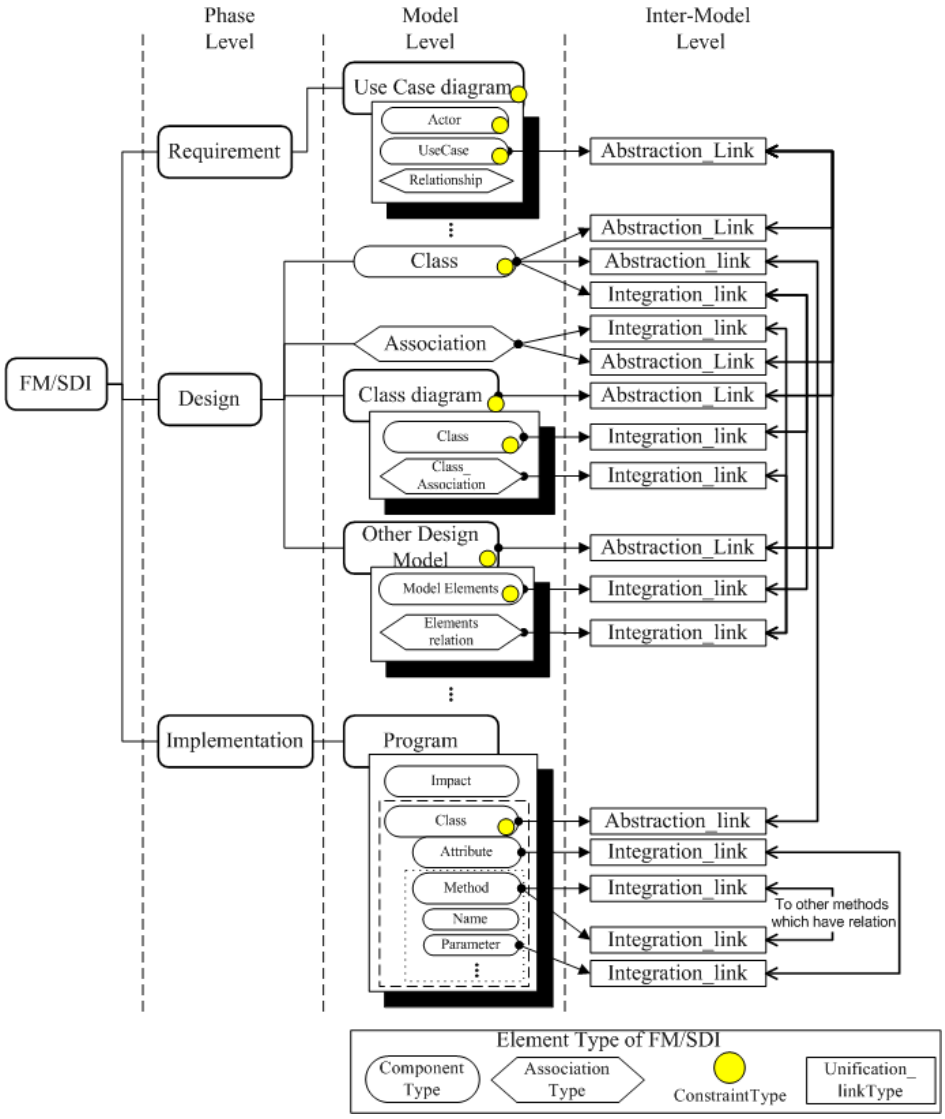

圖 2、正規化整合式軟體系統文件模型(FM/SDI)結構

在這個文件結構中，主要定義了四種基本元素(primitive)：ComponentType、AssociationType、ConstraintType 與 UnificationType。其中，ComponentType 是用來描述原本系統文件中的塑模元件(Modeling elements)中必要的語義要素以及相關塑模所需的元件資訊；AssociationType 則是用來描述塑模元件間的關係及其相關資訊，ConstraintType 是用來描述塑模元素的條件限制如先決條件(precondition)、後續條件(postcondition)等，根據述語描述性質，可以區分成結構性(Structural) 述語、行為性(Behavioral) 述語以及一般性(General) 述語三類；UnificationType 則是用來連接不同模型觀點中，相同或衍生元素間的關係，其中根據連接對象不同，UificationType 衍生出兩種不同的連結：抽象連接(Abstraction Link)，用以連接不同階段中有關連的元件；以及整合連結(Integration link)，用以連接在同一設計階段中，不同塑模觀點中相對應的元件 。這四種基本元素定義了整合式軟體系統程序文件結構所需的基本元素，在整合式軟體系統程序文件結構中所有的物件都是由這四種基本元素衍生而來。

下表為一般常用的軟體塑模元素對應到正規化整合式軟體系統文件模型中所呈現的語法架構：

表 2、塑模元素與 FM/SDI 對應關係

Models /Standards	***Model elements***	***FM/SDI element Representations***
UML Use Case diagram	Actor	<Actor>
	Use Case	<Usecase>
	Association	<association_type>association </association_type>
	Generalization	<association_type>generalization </association_type>
	Extend	<association_type> extend </association_type>
	Include	<association_type>include </association_type>
UML Class, Collaboration, Sequence diagram	Class	<Class>
	Attribute	<Attribute>
	Operation	<Method>
	Interface	<Interface>
	Parameter	<Parameter>
	Association	<association_type>association </association_type>
	Composition	<association_type>composition </association_type>
	Generalization	<association_type>generalization </association_type>
	Dependency	</association_type>dependency </association_type>
	Message	<Message>

其詳細的語法架構的 Schema 如附錄所示。

3.2 範例

舉例來說，下圖 3 是一個圖書館借書系統部分的類別圖，轉換成正規化整合式軟體系統文件模型的話，其結構與內容如圖 4a、圖 4b、圖 4c 所示。

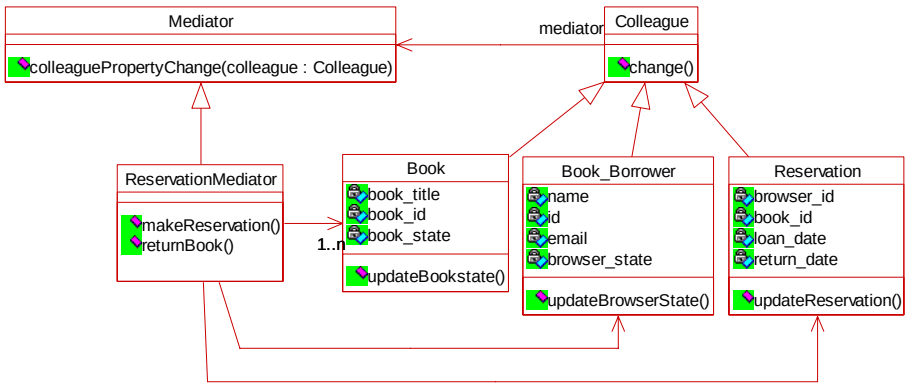

圖 3、圖書館借書系統類別圖(部分)

```
     <design_Class>
        <name>Mediator</name>
        <Integration_link xlink:label="D_Mediator" xlink:title="Class
of Mediator"/>
        <Abstraction_link xlink:from="D_Mediator"
xlink:to="S_Mediator"/>
        <method attribute="public"> colleaguePropertyChange
(colleague:Colleague)" </method>
     </design_Class>
     … …
     </design_Class>
```

圖 4a、類別(classes)的整合式軟體系統程序文件表示

```
     <Association>
          <ass_from>Mediator</ass_form>
          <ass_to multi=”1..n”> ReservationMediator</ass_to>
            <Intgration_link
          xlink:label="Mediator_ReservationMediator"
          xlink:title="Association: Mediator_ReservationMediator"/>
     </Association>
… …
     </Association>
```

圖 4b、關連(associations)的正規化整合式軟體系統文件模型表示

```
<Class_Diagram>
   <class>Mediator</class>
        <Integration_link xlink:href="D_Mediator"/>
   <class>ReservationMediator</class>
   … …
   <class>Book</class>
   … …
   <Class_ass>
       <ass_from>Mediator</ass_from>
       <ass_to multi="1"> Reservation Mediator</ass_to>
        <association_type>generalization</association_type>
      <Integration_link xlink:title="Mediator_ReservationMediator"
         xlink:lable=" Association of Mediator_ ReservationMediator"
         xlink:href="Mediator_ReservationMediator"
         xlink:from="D_Mediator " xlink:to="D_ ReservationMediator"
         />
   <class_ass>
       <ass_from>ReservationMediator</ass_from>
       <ass_to multu=”0..n>Reservation</ass_to>
       <association_type>dependency</association_type>
   … …
</Class_Diagram>
```

圖 4c、類別圖(class diagram)的正規化整合式軟體系統文件模型表示

圖 4 中，三個子圖分別代表類別 class、類別間的關連及類別圖(class diagram)相關描述資訊以 FM/SDI 的表示所呈現的內容。將類別資訊、類別關連與塑模圖分離之主要目的在於相同之類別資訊可能會在多個不同的塑模圖之中出現，一方面不宜將類別資訊與塑模圖合併，再者這樣也可以達到資訊共用及後續的類別整合追蹤。在整個正規化整合式軟體系統文件模型存在許多這類的元件資訊，這些元件資訊會在許多不同的塑模圖中出現，這些可能具有不同名稱或特質但實際上是相同物件的元素，我們就利用<Integration_link> 與 < Abstraction_link> 將這些元素加以串連，以便後續的處理中，當程式設計師對於塑模圖中做修改時，可以輕易的將所有可能被影響的物件、塑模圖乃至程式碼表列出來，提醒程式設計師。

4. 持續研究計畫之工作內容

基於以上這個具有正面意義的階段成果，我們在正持續將 FM/SDI 的模型實際導入，並串連大部分業界於開發過程中所採用與設計出的抽象及實做模型，而達成一個正規化、整合性、應用性的系統文件規範，以協助解決長久以來存在軟體系統整合發展與維護作業上的問題。為了達成這個目的，以下我們列舉幾項接

下來的主要具體研究目標與工作項目：

整合式軟體塑模發展環境

透過整合式軟體系統程序文件表示模型，已經可以將軟體生命週期中所需要用到的需求規格、軟體塑模文件、程式碼適當地結合，然而對程式設計師而言，若沒有適合輔助工具，要撰寫 XML 格式的系統程序文件，也是相當不容易。但是透過現有程式語言處理 XML 的函數套件，如 Java 的 JAXB 等，使得實作一個可以視覺化撰寫系統規格、系統塑模、程式碼開發、並儲存符合整合式軟體系統程序文件的整合開發環境可以被實現。在本計畫的後續研究中，將實作此一整合開發環境，同時作為本計畫中其他相關研究子題的實驗平台。

本整合式軟體塑模發展環境預計將以 JAVA 為開發語言，並依循 N-tier 架構其主要可分為四個階層，分別為使用者介面層(GUI presentation layer)、控制層(control modules)、內容層(content layer)及資料儲存層(XUM data layer)。

使用者介面層提供了圖形化之介面如 FM/SDI browser 提供使用者瀏覽所有模型及其 unification link。Model viewers 提供 UML 標準如使用案例圖(Use case diagram)、類別圖(Class diagram) 等模型的顯示。

控制層是本系統的核心部分，此部份可分為以下模組:

- 基礎功能模組(Foundation modules)

基礎功能模組提供了整合模型基本的功能，包含整合模型內容管理(Content tree manager)、統一關聯管理(Unification relation manager)等。

- 追蹤性支援模組(Tractability support modules)

追蹤性支援模組主要提供了模型的可追蹤性，如漣漪效應的追蹤(Ripple effect tracer)。在使用者修改軟體模型的過程中，系統將隨時紀錄這些修改的模型，並追蹤其直接或間接關聯的模型，輔助使用者作軟體的維護。

- 自動化支援模組(Automation support modules)

自動化支援模組提供了軟體開發的自動化機制包含了支援需求分析至系統設計自動化的需求擷取模組(Requirement extractor)與系統設計至實作自動化的程式碼分析/建立模組(Code analysis/building modules)。

- 驗證支援模組(Validation support modules)

驗證支援模組主要是提供了模型驗證的機制,FM/SDI 裡預計整合常見的軟體規範，如整合了 UML 與設計樣版，由於許多軟體規範往往具有架構或屬性上的條件限制，例如設計樣版是一種規範性的軟體設計架構，因此當 UML 整合了設計樣版之後，系統設計人員就不能夠在任意的使用 UML 建置設計模型，而必須遵循於設計樣版的規範。因此本系統提供了驗證的機制。在使用者修改設計模型的過程中驗證使用者是否違反了設計樣版等特殊軟體規範的限制。

內容層以物件樹狀結構的方式對映至 XML 中的模型資訊，系統的其他模組只要透過管理此內容樹就可以達成間接存取 XML 的機制，以免除複雜的文字解析動作。

資料儲存層以 XML 的格式將統一模型儲存於檔案中。

軟體文件驗證

在上述的這個整合式軟體塑模發展環境的發展與 FM/SDI 的延續研究中，軟體文件及其組成單元彼此與統合關係之間的一致性檢查與驗證,是非常關鍵的核心考量之一。

目前大部分的軟體塑模方法，都是採用視覺化圖形的表示方式，然而有些細部的條件，卻無法完全用視覺化方式來表達，而這些條件往往表達了塑模文件中，一些行為上、結構上的特徵，這些特徵在模型間檢查時，卻可以扮演重要角色，常用的塑模方法中，其實也都有這些加註條件的機制，然而一方面現有的塑模輔助工具，並沒有特別強調這方面的特性，而且一般的塑模方法有其涵蓋範圍的限制，並無法兼顧軟體開發過程中不同的需求，因此這方面的塑模限制語言並未被程式設計師善加運用。在先前的整合式軟體系統程序文件表示模型中，我們保留了加入條件限制的機制(即前一章所提之 ConstraintType 衍生之條件元素)，以便徹底運用各種塑模方式的特性，例如 UML 中的 OCL 可以將原本視覺化的塑模圖形加入先決條件、後置條件等限制，即可套用適當的一致性檢查機制，如派翠網路(Petri Net)等，可進行模型內、模型間的一致檢查，加上合式軟體系統程序文件表示模型本身已經容納各種常用塑模方式的表示結構，使得以 FM/SDI 為基礎的塑模一致性檢查，可以跨越塑模方法的限制，檢查出不同塑模模型間不一致的現象，減少程式設計師因使用人工驗證可能造成遺漏。

下圖是模型一致性檢查派翠網路圖，當程式設計師需要驗證或檢查模型中，或不同模型間是否一致時，系統會從 FM/SDI 中取得相關模型的資訊及條件轉換成派翠網路中所需的 place、transaction，以及每個 place 的先決條件、後續條件

與限制，而每條路徑也會有其條件限制。利用派翠網路的特性，當 token 從 Start 部分 fire 之後，如果所有的 place 其條件都符合，最後 token 將可以抵達 End 處，若有一個或數個 place 有條件不符，token 無法達到 End，那就在表此模型中的元素不一致，此時可檢查 token 最後卡在何處，就可以知道哪個元素可能與其他元素間無法匹配。

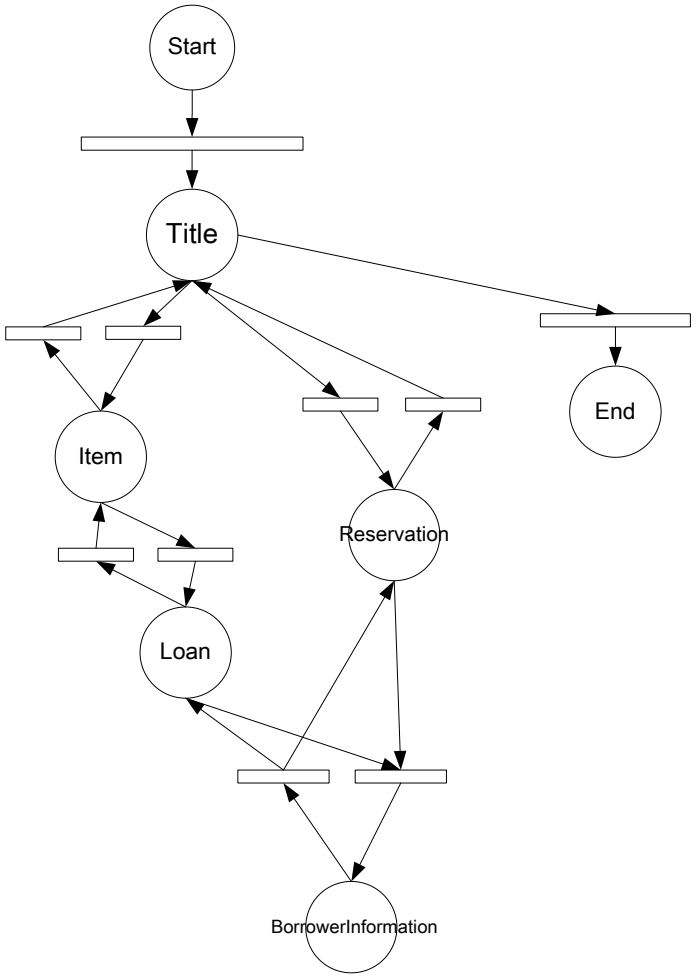

圖 5、模型一致性檢查派翠網路圖

5. 總結

軟體工業將是我國在未來國際社會競爭力不可或缺的一環，如何提高軟體品質與產能，將是我國能否在世界軟體工業中佔有舉足輕重地位的重要因素。本系列性的軟體文件整合研究計畫若能繼續推展延伸，其成果將可對於目前軟體開發過程中軟體品質不穩定、高成本、高人力等亟待突破的瓶頸，將提供有效的解決之道，並可在軟體技術研究的競爭潮流中，取得相當正面的突破。

另外，目前國內軟體產業中的主流人力，多是過去十年所培養出來的。由於這些人員在學校中尚未學習到這些新的軟體技術規範，進入業界後也因為缺乏適合的輔助工具，或多或少使得這些軟體產業的主流中堅對於引入新軟體規範卻步不前。近年來資訊教育在我國大專以上的成長明顯，尤其在導入並普及至技職教育中，由猶如雨後春筍出現的各類資訊、資工、資管、資傳、資網等相關所、系、科，更是一股正當蓬勃發展的氣象。資訊人員如何學習新的軟體趨勢技術，迎接來自大陸、印度、乃至於全世界軟體技術挑戰，而成為未來國內軟體產業中主要的開發與產值的人力來源，也將是一個嚴肅而急迫的議題。因此，如何在學校中，讓學生學習到新的軟體規範，讓他們樂於在實務中依循這些軟體規範，將成為未來國內軟體產業是否能進入到新一階段的關鍵。

本計畫希望將過去在軟體開發過程中斷斷續續的軟體塑模方法以及軟體規範做一整合，並開發出讓使用者容易使用的整合開發環境，同時透過整合在 FM/SDI 中的資訊，對於軟體文件一致性加以檢查，提高系統文件的品質，進一步有效地提升軟體系統全生命週期的軟體品質與產能效益。

參考資料

[1] Booch, G. (1994). *Object-oriented analysis and design with applications* 2nd ed. Redwood City, Calif. : Benjamin/Cummings Pub. Co., 3-25.

[2] Deitel, H., Deitel, P., Nieto, T., Lin, T., & Sadhu, P. (2001). *XML how to program*. Upper Saddle River, NJ : Prentice Hall.

[3] Object Management Group. (2001, August). *OMG unified modeling language specification*. Version 1.4, Retrieved July 16, 2001 from http://www.omg.org/technology/documents/recent/omg_modeling.htm

[4] Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (1995). *Design patterns: elements of reusable object-oriented software*. Reading, MA.: Addison-Wesley.

[5] Wirfs-Brock, R. J., & Johnson, R. E. (1990). Surveying Current Research in Object-Oriented Design. *Communications of the ACM*, 33(9),105-124.

[6] World Wide Web Consortium (W3C), *Extensible Markup Language* (*XML*), Retrieved July 16, 2003 from http://www.w3.org/XML/

[7] Lu C.W., Chu C.W., Chang C.H., Lian W.D., Yang D.L. (2003). Integrating Divers Paradigms in Evolution and Maintenance by an XML-based Unified Model , *Journal of Software Maintenance and Evolution*