國立交通大學

理學院科技與數位學習學程

碩 士 論 文

以 PRMRX 建構法設計暨實作中小學薪資自動化系統

Building an Automatic Payroll Management System for Elementary and Junior High Schools with the PRMRX-based Construction Method

研 究 生：羅仁治

指導教授：陳昌盛　博士

中 華 民 國 一 百 年 六 月

以 PRMRX 建構法設計暨實作中小學薪資自動化系統

Building an Automatic Payroll Management System for Elementary and Junior High Schools with the PRMRX-based Construction Method

研 究 生：羅仁治　　Student：Jen-Chih Lo

指導教授：陳昌盛 博士　　Advisor：Dr. Chang-Sheng Chen

國 立 交 通 大 學

理學院 科技與數位學習學程

碩 士 論 文

A Thesis
Submitted to Degree Program of E-Learning
College of Science
National Chiao Tung University
in partial Fulfillment of the Requirements
for the Degree of
Master
in
Degree Program of E-Learning
June 2011
Hsinchu, Taiwan, Republic of China

中 華 民 國 一 百 年 六 月

以 PRMRX 建構法設計暨實作中小學薪資自動化系統

學生：羅仁治　　指導教授：陳昌盛博士

國立交通大學理學院科技與數位學習學程

摘 要

本論文提出的 PRMRX 建構法，提供一套整合性的開發框架，整合流程設計、資料庫規劃、程式設計、權限控管、資料交換的需求，輔助開發人員，將作業流程轉換為程式邏輯。在本論文中我們另外也建置一套實驗性質的中小學薪資系統，透過此法提供的標準化方法及輔助開發框架，進行不同階段的概念及觀點轉換，具體呈現此法中所有的理論依據與實作流程，驗證此法可行。

PRMRX 建構法，提供明確的方法及步驟，將作業流程轉換為 Petri net 流程圖、資料庫關連圖與程式設計藍圖等，完成系統的設計規劃，最終再一一實作 MVC 程式模組、RBAC 權限控管模組、XML 資料交換模組等，完成系統的建置。尤其，PRMRX 建構法可適用於各種不同的平台及程式語言開發的系統，協助整合流程規劃與程式開發。

關鍵字：Petri net、PRMRX、薪資、MVC、RBAC、XML

Building an Automatic Payroll Management System for Elementary and Junior High Schools with the PRMRX-based Construction Method

student：Jen-Chih Lo　　Advisors：Dr. Chang-Sheng Chen

Degree Program of E-Learning
College of Science
National Chiao Tung University

ABSTRACT

In principle, the PRMRX-based construction algorithm proposed in this thesis provides a systematic development framework for integrating the various requirements of workflow design, database design, program design, and facilitating the implementation of different levels of access control and data exchange of the related systems. For validating the effectiveness, we use it to help design and implement a web payroll system for elementary and junior high schools.

The PRMRX-based construction method provides a set of precise steps to convert among different viewpoints and concepts of specific stages by transferring the workflow process to its corresponding Petri net graph, and then to the creation of the relational database diagram and program blueprints, composed of MVC conceptual modules (including different roles, different tasks and resources, etc.). From the above, we can implement the related MVC program modules, RBAC access control modules and XML data exchange modules. Moreover, as the PRMRX-based construction algorithm provides a standard approach, with some minor modifications, it can be applied to build various web information systems with different platforms and/or program languages to help integrate workflow process design and program development.

Keywords: Petri net, PRMRX, Payroll, MVC, RBAC, XML

誌 謝

這份研究能夠完成，首先要感謝的就是指導教授—陳昌盛教授，在每個禮拜的 meeting 中，提供建議，作腦力激盪，產生不一樣的想法，使論文能夠有更完整的結構，也謝謝教授對於工作上的幫忙，幫助解決系統除錯上的難題，讓工作能夠順利。

接著，謝謝林億霖先生的及時打氣，在最低潮的時候接到他的勉勵卡片，內心的沮喪跟負面情緒，頓時消去不少，雪中送炭的情誼讓人難忘也記憶深刻。

也感謝父母親在生活上的包容，包容我生活上的許多習慣，讓我能夠無後顧之憂地完成論文的寫作。

目　錄

表　目　錄

圖　目　錄

第一章　緒論

1.1 簡介

本論文以中小學薪資作業的自動化為目的,對於系統作業流程的規劃、程式藍圖規劃、資料庫關連圖規劃、系統實作、權限控管及資料交換，提出一套具整合性、彼此環環相扣的方法，達到作業自動化的目的。

本論文設計的薪資系統以 Web 為執行平台，使用者端只要使用瀏覽器連上伺服器，就可以執行應用程式，所有的程式維護工作及資料的存取均由伺服器端負責，使用者不必安裝任何額外的程式或是外掛，就可以執行相關的工作，進行各項薪資作業，這樣的應用環境帶給使用者極大的方便性,薪資作業人員不必攜帶任何資料或安裝任何程式,只要有連網的環境，即可作業，提高操作便利性。

中小學薪資系統牽涉層面廣泛[4]，大多數的系統架構龐大複雜而難以立即了解其整體作業流程，本論文擬導入 Petri net[6][21]流程規劃機制作為程式設計藍圖及資料庫關連圖規劃設計的依據，可大幅簡化系統規劃的困難。一般而言，薪資作業牽涉到金錢的處理，是一項敏感又受關注的作業程序，稍有錯誤，容易引起爭執，因此，相關的作業往往手續嚴謹，深恐引發錯誤，導致爭執，本論文為處理需要精確作業的薪資流程，引入 Petri net，運用 Petri net 需要精確定義，不容許模糊空間的特性，規劃薪資作業流程，避免對流程的誤解，造成作業的錯誤。

以 Petri net 規劃的作業流程圖，釐清了薪資作業中的模糊空間，完整呈現作業的先後順序及相對應的關係，這種精確表達流程的能力，在作為程式撰寫藍圖時，不會造成誤解；Petri net 也具有區分資源及工作的特性，根據這種特性，資源可以轉化成資料庫中的資料表，據以規劃關連式資料庫，工作則轉化成相對應的程式邏輯，讓程式的撰寫有所依循。Petri net 在本論文中所扮演的角色，不但是作業的流程圖，也是程式設計的藍圖。

其次，根據實務經驗觀察，如不經過適當轉換處理，即使 Petri net 流程管理規劃機制已簡化規劃問題，但實作系統程式碼的困難度仍然相當高，後續維護管理程式碼時仍需耗費相當大的心力。因此，本論文擬進一步以 MVC[7][13]模組化的方式，將 Petri net 程式設計藍圖轉換成一個個獨立且較為單純的 MVC 程式模組，各 MVC 程式模組的撰寫彼此獨立，不相干擾，從而降低程式碼開發及後續維護時所需耗費的心力。

再者，稍具規模的學校，薪資作業多由多人協同作業完成，不同職務的負責人(出納、人事等等)於不同時間點，可能分別執行不同的工作，以完成整個作業的流程。本論文為滿足協同作業的需求，除了套用 MVC 模組化的規劃方式將工作分門別類，置放於不同的 MVC 模組內，再進一步將 MVC 程式模組與 RBAC[1][23]的觀念整合，實現以 MVC 為基礎工作單位，以職務角色為判斷依據的權限管理系統，藉由職務角色的判斷，賦予不同角色執行不同 MVC 程式模組的能力，增強系統的安全性，也展現線上協同合作的分工優勢。

其次，薪資流程管理也會牽涉到許多相關的子系統(人事、健勞保、出納等)，通常這些系統建置的時間點不同，參與人員不同，通常差異性很大，往往是異質性的平台系統(例如常見的 Windows、Linux 系統等)。因此如何讓不同異質系統有效率的進行系統間的資料交換並加以整合是非常重要且普遍的課題。本論文透過建置 XML 資料交換中心，進行人事系統與薪資系統的資料交換，讓共享資料可以透過網路傳輸交換，減少複雜的操作程序可能帶來的工作負擔並減少出錯機率。XML 資料交換中心的資料交換機制，可以作為進一步擴展系統彈性並作為異質平台整合的基礎。

本論文所提出的方法結合了 Petri net 流程規劃、RDBMS 規劃、MVC 設計模式、RBAC 權限控管及 XML 資料交換機制，取每個英文字母的首字，簡稱為「PRMRX 建構法」(「PRMRX 建構法」的 P 是 Petri net，第一個 R 指 RDBMS，M 是 MVC，第二個 R 為 RBAC，X 意指 XML。)。更重要的，「PRMRX 建構法」不但適用於薪資系統的開發，只要經過適當的調整及修正，基本上這套方法也可適用於其他常見的應用平台(例如人事系統、採購系統等)、不同種類程式語言的系統開發專案，不論是架構在 PHP 及 MySQL 上的系統，或是採用.net 及 MsSQL 的平台，或是以 JAVA 及 PostgreSQL 進行建構的系統專案均可一體適用，而本論文的實作成果，也驗證了這是一套能付諸實現的有效方法。

1.2 研究動機

中小學薪資作業的處理，繁瑣而又需要以細心的態度有條理的面對。一旦作業疏失，又需要高度的情緒管理，對緊接而來的後續事項進行補救措施，如果能夠導入一套自動化的薪資作業系統，就可以大幅減輕薪資作業相關人員的心理負擔，減少一時粗心大意所犯下的錯誤，有效提高作業效率，強化行政流程的順暢度。

薪資系統的規劃，必須考量到完整精確的流程訂定、程式實作、資料庫的規劃，並融入多人協同作業的需求與異質系統間的資料交換需求，這

些需求整合後，才能據以規劃出能夠確實實作的方法，因此，本論文提出 PRMRX 建構法，整合這些需求，實現這個可能性。

從許多實際的網路系統開發專案中，我們經常可以發現到，這些程式大多是在沒有藍圖規劃的狀況下撰寫，後續的維護修改，通常也沒有可依憑的藍圖，以協助了解整個運作的流程，相關系統的開發通常是以「摸著石頭過河」的方式在發展。在撰寫薪資系統這種與金錢有關的系統時，正確性是相當重要的，如果沿用此種「摸著石頭過河」的方法開發系統，就非常可能引發不可預期的錯誤，造成重大的困擾。因而尋找一套能精確表達流程，並能據以作為程式藍圖的開發方法，就顯得相當重要。

薪資作業的多人協同作業模式，是現實運作的狀況，藍圖的規畫與實作勢必也要符合現實狀況中，多種職務角色，多人協同的作業模式，才能滿足作業的需求。如果能在藍圖規劃之初就考慮這方面的需求，系統的規劃才能更貼近現實情況。

跳離薪資作業系統的層面，以更高的視野審視行政作業的流程，薪資系統除了內部系統的多人作業分工需求外，薪資系統與其他相關系統間的溝通合作也勢不可免，系統與系統之間的資料交換傳輸可以作為系統與系統間溝通的媒介，如果能建立一個統一的資料交換平台，與薪資作業搭配，就能夠實現各系統間的分工合作，整個配套的作業環境才能完整實現。

1.3 研究方法

許多網路資訊系統的開發，通常會經由專家訪談或是整理及分析使用者的經驗，決定功能及實作的項目，但使用者的經驗及訪談的結果，沒有經過適當的轉換及整理，通常難以直接應用於實際的流程規劃及系統開發上，本論文提出一套系統化的 PRMRX 建構法，將現實作業的狀況對映到開發的需求及規劃上，嘗試解決這方面的問題。

表 1-1 為 PRMRX 建構法與常見 OOAD 設計與實作方法的簡單比較表，PRMRX 建構法以流程規劃觀點(Petri net)出發，採用 MVC 框架為基礎進行開發設計，使後續對程式的修改和擴充功能簡化，並且使程式某一部分的重複利用成為可能。除此之外，此模式透過對複雜度的簡化，使程式結構更加直覺，具有容易使用開發的優勢。當然，PRMRX 建構法也可以搭配其他的軟體設計方法，如 RUP[22]，使開發過程更符合需求。

承接上述的分析，PRMRX 建構法以作業流程的觀點出發，結合權限控管及資料交換的需求，將 Petri net 流程圖轉換為程式設計藍圖，這份程式設計藍圖再轉換為 MVC 程式模組及資料庫關連圖，並據以實作 RBAC 權限控

管模組及資料交換模組。

具備 PRMRX 建構法的理論架構、實作工具及環境後，以此為基礎，即能透過 PRMRX 建構法提供的系統化的方法建構中小學薪資系統，驗證此套開發框架適用於系統的開發上，薪資系統開發過程中，PRMRX 架構法的理論與實作一一對應，提供一具體而實際的驗證。

表 1-1：PRMRX 建構法與 OOAD[17]方法比較表[2][3][5]

項次	PRMRX 建構法	OOAD 一般性方法
1.流程規劃	透過 Petri net 技術，從流程觀點出發，方法較直覺，容易理解，易於使用。	從物件導向觀點出發，描述分散的互動群體(系統)，方法比較複雜。
2.用戶與系統互動關聯	結合 Petrinet(可將流程問題以視覺化處理) 和 MVC 技術 ■ 有嚴謹的數學工具可協助驗證流程合理性 ■ 適合描述動態分散式運作的系統，容易與使用者溝通，確認作業流程的正確性。	OOAD 通常比較複雜 ■ 常必須針對一般系統問題拆分成物件類別(class), 狀態(data member) 和行為(functions)等幾個部分，再整理彼此的關聯，較不易與使用者溝通，容易遺漏使用者的需求。
3.開發框架支援	套用 MVC 框架，易於實作系統 ■Model 部份可以整合 OOAD 的設計方法，使用常見 UML 工具進行設計，發揮 OOAD 的優點。	未指定開發框架 ■設計類別彈性雖大，但需要反覆修改，開發期程較久。
4.權限控管機制	以 MVC 框架提供的類別實作 RBAC 權限控管機制，降低設計的複雜度。	未規範權限控管的方式 ■必須自行設計權限控管方式，開發時間較長。
5.跨平台資料交換	指定 XML 作為跨平台資料交換的格式	未指定 ■ 可套用 XML 等相關文法

1.4 章節介紹

本論文提出的 PRMRX 建構法，協助系統分析人員及開發人員以共通的 Petri net 流程圖進行溝通，並提供一套開發框架，將現實作業的工作流程轉換為程式邏輯，其中牽涉到許多的概念與觀點轉換及實作步驟，本節將 PRMRX 建構法所涉及的相關資訊整理如表 1-2，PRMRX 建構法的主要內容，我們將於論文第二章至第五章，依序加以陳述及說明。

本論文的的第二章，介紹 PRMRX 建構法之相關技術，包括 Petri net、MVC、RBAC、XML、Zend Framework、jQuery 等等，第三章說明 PRMRX 建構法的主要內容,第四章為本論文所建置的薪資系統之系統架構及實作環境，第五章以 PRMRX 建構法實作中小學薪資系統，第六章為使用者問卷調查與使用經驗分析，第七章為結論與未來研究方向

表 1-2：論文架構

主題	項目	子項
1.PRMRX 建構法的背景知識	1)Petri net	(1)transition 及 place 交錯配置。 (2)能精確具體地表示工作流程 (3)具有可驗證的特性。 (4)圖形化的語言，直覺簡明。
	2)MVC	(1)分離呈現畫面及程式邏輯。 (2)降低後續的功能擴增及維護的複雜度。
	3)RBAC	(1)以角色為中心，簡化管理複雜度。 (2)以 MVC 模組名稱作為 RBAC 的權限名稱。
	4)XML	(1)與軟硬體無關的中立性。 (2)依據資料意義訂定標籤名稱。
	5)Zend Framework	(1)以 PHP 開發的開發框架 (2)具有大量的函式庫

		(3)提供 MVC 開發框架。
	6)jQuery	(1)瀏覽器端的開發框架。 (2)簡化 javascript 的開發難度。
2.PRMRX 建構法	1)以 Petri net 規劃作業流程	(1)以工作執行及資源配置的觀點規劃。 (2)transition 表示工作，place 表示資源，交錯配置表示工作流程。 (3)使用 Petri net 可驗證的特性，驗證流程的合理性。 (4)作為程式設計藍圖。
	2)永久性資源轉換為資料庫關連圖	(1)從 Petri net 流程圖中的資源存取觀點轉換為資料庫規劃的觀點。 (2)永久性資源轉化為資料庫，暫時性資源則暫存於記憶體。 (3)程式統一存取的資源庫
	3)Petri net 流程轉換為 MVC 模組	(1)以 RBAC 的觀點將流程轉換為 MVC 模組。 (2)MVC 模組內的流程較為單純，相關的資料表及工作數量較少，較容易實作。
	4)程式邏輯及畫面流動封裝在 MVC 模組內	(1)Petri net 流程圖以工作執行及資源配置的觀點規劃，程式邏輯及畫面流動則封裝在 MVC 模組內，減化流程圖的複雜度。 (2)分離流程觀點及程式設計觀點，避免混淆。
	5)RBAC 的權限管控	(1)強化系統安全性。 (2)簡化權限管理的複雜度。 (3)獨立於 Petri net 工作流程之外，以角色、使用者及 MVC 模組名稱的觀點規劃

	6)XML 資料交換中心	(1)整合異質系統。 (2)統一更新資料。
3.系統架構及實作環境	1)Xen Server 的建置	(1)以虛擬化軟體 Xen Server 建置模擬網路環境。 (2)於虛擬化軟體建構的環境下，使用 Ubuntu server、PHP 及 MySQL 分別建置三套作業環境。
	2)軟體工具及管理工具的使用	(1)程式設計工具。 (2)資料庫管理工具。
	3)安全政策及機制的建立	(1)建立防火牆並設定防火牆政策。 (2)網頁程式弱點掃描。
4.以 PRMRX 建構法實作中小學薪資系統	1)以 Petri net 規劃薪資作業流程	(1)分析薪資計算的作業方式，將薪資作業區分為工作及資源二部份。 (2)交錯配置工作及資源，以 Petri net 規劃薪資流程。 (3)以 PIPE 規劃及驗證流程的合理性。
	2)以 MySQL Workbench 規劃關連式資料庫	(1)將 Petri net 流程圖中的永久性資源轉化為資料庫關連圖。 (2)分析作業所需的資料表及欄位。 (3)將資料庫關連圖匯入資料庫中，建立能實際運行的資料庫。
	3)薪資計算 MVC 模組的規劃及實作	(1)以 RBAC 的觀點，將 Petri net 流程轉換為不同角色執行的 MVC 模組。 (2)以 Zend Framework 及 jQuery 實作 MVC 模組。

	4)RBAC 權限控管模組的實作	(1)規畫 RBAC 的資料庫關連圖。 (2)以角色、使用者及 MVC 模組三者的關係規劃運作流程。 (3)實作權限控管模組。
	5)XML 資料交換模組及 XML 資料交換中心的實作	(1)確認 XML 資料交換的格式。 (2)以程式邏輯的觀點規畫運作流程。 (3)分別實作薪資系統的 XML 資料交換模組及 XML 資料交換中心。

第二章　背景知識

本論文的第二章，介紹 PRMRX 建構法背後牽涉的相關背景知識，包括 Petri net、MVC、RBAC、XML、Zend Framework、jQuery 等。其中，Petri net、MVC、RBAC 及 XML 為 PRMRX 建構法提供理論基礎，Zend Framwork 及 jQuery 則是 PRMRX 建構法的實作工具，將工作流程轉化為實際上線的自動化程式。

2.1 Petri net

系統分析通常需要繪製流程圖，Petri net 有許多視覺化的工具可以進行流程驗證[6][21]，透過 Petri net 流程圖的協助，可以很容易釐清流程的狀態，一旦驗證流程不合理，可以進行修正；另一方面，Petri net 可以精確描述狀態的特性，例如在檢視薪資作業流程規劃時，能精準地標定問題點，修正問題。

Petri net 中，各種圖形符號的組合方式及 token 的轉移都有明確嚴謹，定義清楚的規範，圖形化的方式易於溝通，加上嚴謹的定義，避免溝通時產生誤解，非常適合需要精確定義作業流程的薪資作業，以 Petri net 定義的薪資作業流程，不會有因人而異的作業流程解釋方式，減低溝通成本，減少錯誤的發生機率。

2.1.1 Petri net 是描述流程且定義清楚的圖形化語言

Petri net 是由 transition、place、arc 及 token 四種符號組成的圖形化語言[21][25]。transition 通常表達一個動作，常用表示法為矩形；place 通常表達狀態或是資源(本論文以 place 表達資源)，常用表示法是空心的圓形；arc 是帶箭號的線段，表示動作的流向；token 表達主要流程變化到某一瞬間的狀態，通常是以 place 的符號加上圓點表示。這四種圖形構成完整的 Petri net 圖形，表 2-1 是 Petri net 的圖形符號及符號名稱的對照表。

表 2-1：Petri net 圖形符號、符號名稱與擺放位置對照表

符號名稱	圖形符號	擺放位置
Place	◯	與 transition 相連
Transition	▌	與 place 相連
Arc	→	連接 place 及 transition
Token	⊙	Place(◯)裡頭的圓點(・)

Petri net 的四種圖形符號，擁有固定的組合規則，place 與 transition 間使用 arc 相連接，至於 place 與 place 間無法使用 arc 相連接，且 transition 與 transition 間也無法使用 arc 連接，transition 與 place 一定要使用 arc 交錯相連。至於 token 則擺放在 place 裡頭，在 place 之間進行移轉，表示 Petri net 圖形狀態的改變，借由移轉 token，我們可以了解流程的執行狀態。

圖 2-1 是一個 Petri net 說明圖形，使用 arc，將 place 與 transition 交錯連接，token 則擺放在命名為 P0 的 place 裡頭。

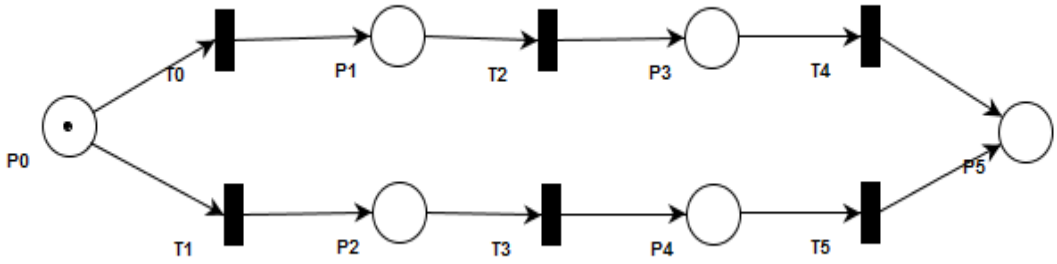


圖 2-1：Petri net 圖形

Petri net 圖形中，所有指向該 transition 的 place，稱為該 transition 的 input place，由該 transition 往外指的 place，則稱為該 transition 的 output place。例如圖 2-1 中的 P0 就是 transition T0 的 input place，P1 則為 transition T0 的 output place[6]。

Petri net 以 token 的分佈狀況表示各種不同的狀態，token 的移轉方式說明，請參閱 Appendix A1。

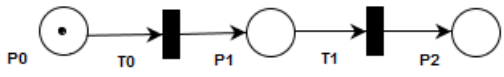
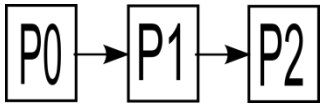
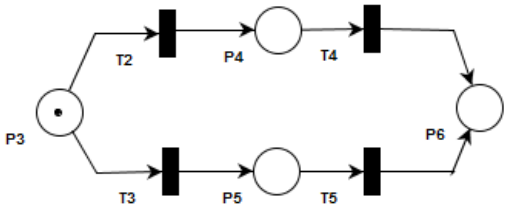
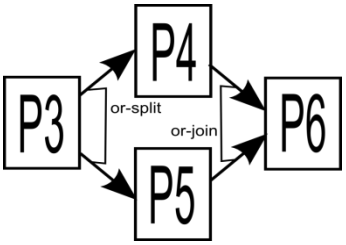
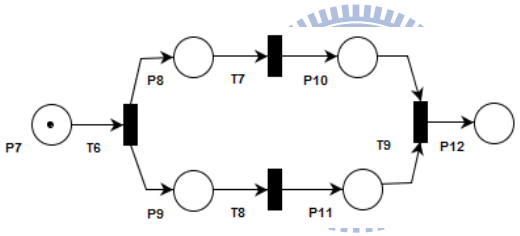
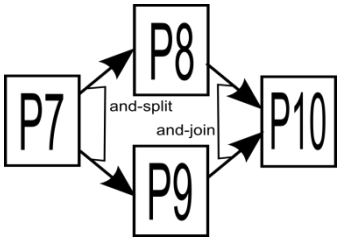
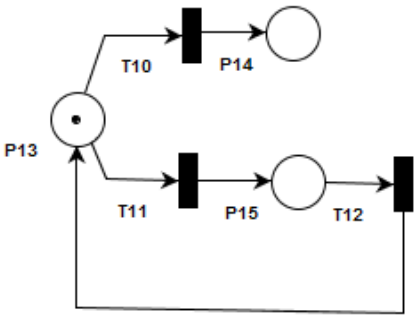
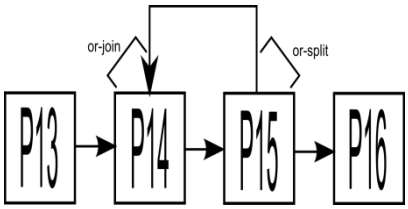
2.1.2 Petri net 可以清楚呈現常見的作業流程模式

使用 Petri net 繪製的作業流程圖，可以從 token 的移轉及分佈狀態，精確地表達流程的走向及狀態，而這正是薪資作業流程所非常需要的，透過視覺化的 Petri net 流程圖，比較能夠釐清作業中容易出現差錯的部份，進而修正流程，使流程的運作合乎邏輯，消弭錯誤流程帶來的不良影響。

表 2-2 是四種常見的作業流程模式，包括循序模式、判斷模式、同步模式及迴圈模式。運用在作業流程中時，為了清楚描述作業模式，會以圖像的方式描述，表 2-2 中我們比較按照 Petri net 規則製作的流程圖和直覺方式製作的流程圖，由於 Petri net 定義清楚的特性，不會造成誤解，

而能精確描述作業的流程。

表 2-2：作業流程模式描述方式比較表

作業流程模式	Petri net 的描述方式	直覺的描述方式
循序模式		
判斷模式		
同步模式		
迴圈模式		

底下我們就分別介紹表 2-2 中的四種作業模式

(1)循序模式：表 2-2 中，第二欄以 Petri net 描述循序模式的作業流程中，token 的移動順序是 P0、再來是 P1，最後 token 停留在 P2，結束循序的作業流程，藉由 token 的分佈，可以清楚流程執行的狀況，整個流程的狀態

也可以呈現，另一方面直覺的描述方式(第三欄)，就無法知曉循序流程的執行狀態，只能描述執行的先後順序，而無法呈現執行的狀態。

(2)判斷模式：以 Petri net 描述的判斷模式作業流程，token 的移轉只能選擇二條路徑之一，一條從 P3 到 P4，最後停在 P6，另一條路徑，從 P3 出發，經 P5，最後到 P6，由 token 的移轉方式，可以清楚知道判斷模式的流程，只能選一條路徑執行工作，而由 token 的移轉方式，也可以清楚知道選擇哪條路徑執行工作；另一方面，直覺式的描述方式，必須加上文字說明，才可以明白規劃流程的設計者真正的意圖，如果溝通不良，容易造成誤解。

(3)同步模式：根據 Petri net 圖形的定義，token 會選擇二條路徑，同時進行 token 的移轉，一條路徑是 P7 到 P8，再到 P10，另一條是 P7 到 P9，再到 P11，token 同時在這二條路徑移轉，最後 token 停留在 P12，流程結束，由 token 的移轉，可以明白這是同步模式的作業流程，而直覺的描述方式，如果不加上文字說明，容易與判斷模式的作業流程圖混淆，Petri net 在這種情況下是比較理想的描述方式。

(4)迴圈模式：Petri net 圖形中的 token 移轉，必須選擇一條路徑，一條是 P13 到 P14，流程結束，另一條路徑是 P13 到 P15，再回到 P13，這條路徑是不斷重覆的迴圈，從 token 的移轉，可以清楚流程的走向；另一方面，直覺的描述方式，很難清楚知道，流程是從 P13 到 P14，再到 P15，最後到 P16，還是選擇 P13 到 P14 到 P15，再到 P14 的迴圈，這容易造成誤會，不易釐清流程的執行方式。

2.1.3 Petri net 圖形與 UML 活動圖的比較

UML 活動圖與 Petri net 都是以流程導向的觀點，使用圖形化的方式描述流程的運作，Petri net 能夠以 token 的分佈精確地描述流程地狀態，確認流程中各項狀態是否能夠抵達，而對流程進行精確地驗證。UML 活動圖的圖形符號，雖然數量較多，但卻很難精確地描述流程的狀態，再加以驗證，我們底下會以實際例子(圖 2-2 是 Petri net，圖 2-3 是 UML)加以對照說明。

圖 2-2 是以 Petri net 描述的流程，圖中有四個 place(分別是 P0-P3)，以 token 描述目前流程狀態的方法是(1,0,0,0)，1 代表 P0 有一個 token，接下來的 0，分別依序表示 P1、P2 及 P3 的 token 數量都是 0，如果圖 2-2 中的 T0 被 fired，圖 2-2 的流程狀態就會轉換為(0,1,0,0)，而 T0 被 fired 後，會消耗一個 token（原先 P0 的 token）；假如 T1 無法同時被 fired，流程的狀態無法轉換為(0,1,1,0)，可以推論，token 無法被移轉到 P3，達

到(0,0,0,1)的狀態。

另一方面，假定圖 2-2 中的 T1 被 fired 後，會到達(0,0,1,0)的狀態，但是，由於 T1 被 fired 後會消耗掉一個 token，一樣無法到達(0,1,1,0)的狀態，token 一樣無法移轉到 P3，轉換為(0,0,0,1)的狀態。簡言之，如果流程的目的地是要達到(0,0,0,1)的狀態，T0 及 T1 必須同時被 fired，但圖 2-2 所顯示的流程無法達到目的，我們就可以驗證出圖 2-2 的流程有問題，必需要加以修正

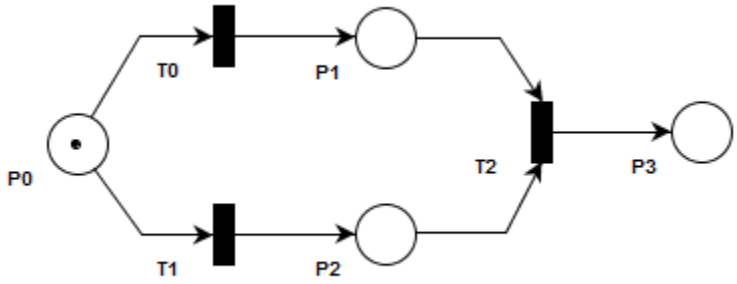


圖 2-2：Petri net 的驗證

圖 2-3 的 UML 活動圖流程是無法到達結束點的，因為分岔結點後的動作狀態 2 及動作狀態 3 無法被同時執行，不同於 Petri net 可以用 token 移轉精確地描述流程狀態的改變，但 UML 活動圖無法精確地描述流程的狀態，只能以文字或口語說明，流程會停留在動作狀態 2 或是動作狀態 3，就無法繼續進行，驗證較困難，如果問題比較複雜(圖形較複雜)，很容易疏忽掉有問題的流程，而沒有加以修正。

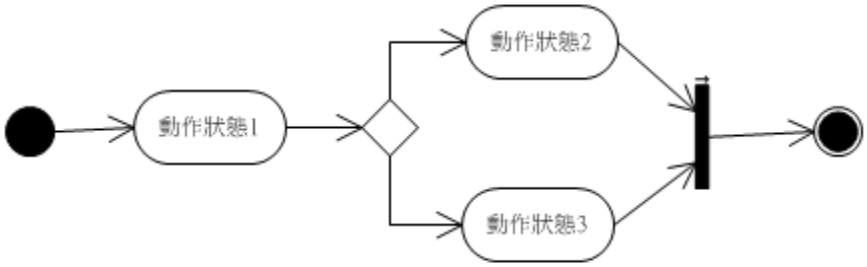


圖 2-3：UML 活動圖的驗證

由以上的對照比較，我們可以發現 Petri net 有許多工具可以進行流程驗證，在流程的繪製上可以釐清流程的狀態，一旦驗證流程不合理，可

以很快的修正，比 UML 活動圖更適合用來處理類似本論文所要探討薪資系統的流程規劃上。

2.2 MVC 設計模式

MVC 是 Model、View 及 Controller 三個英文字的首字母縮寫，MVC 設計模式是一種軟體設計的觀念，如圖 2-4，將程式碼分為 Model、View 及 Controller 三部份:與使用者互動及呈現外觀的程式碼,放在View的部份，資料存取及作業邏輯的程式碼則集中到 Model 部份，Controller 負責回應使用者的需求，組合要求的 Model 及 View，將結果回傳給使用者。

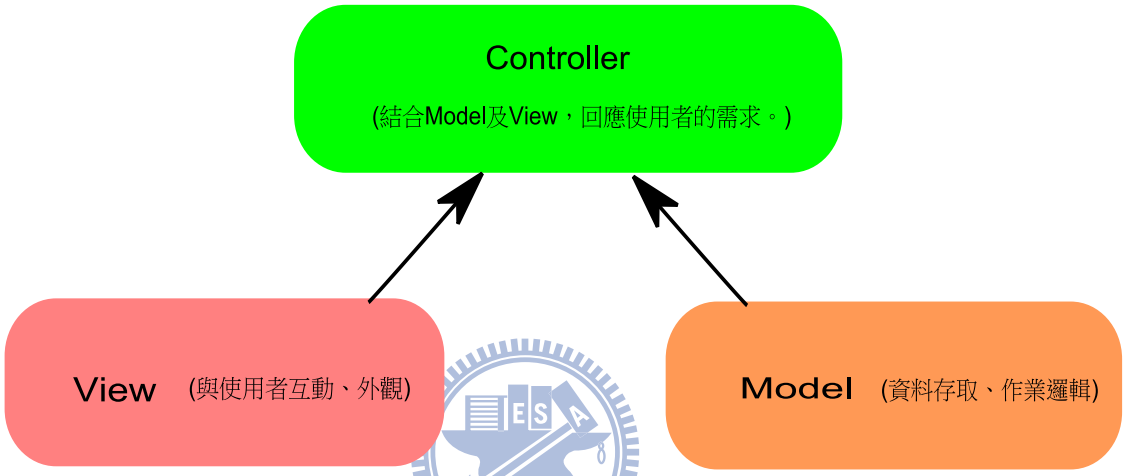


圖 2-4：MVC 程式碼分工圖

程式碼依據功能區分為三個部份，有利於分工作業的進行，美工設計專長人員專注於 View 的部份，設計互動性高，具有良好使用經驗的介面，程式設計師則投注心力於 Model 部份程式邏輯的開發，這樣的方式可以個別分頭進行工作，不會互相干擾作業的進行，待開發完成後，再整合成功能完整的程式。

2.2.1 模組化的 MVC 程式碼減輕維護的負擔。

本論文將薪資作業流程中切割出的單項工作，以數項工作一組，群組化成一個 MVC 模組的方式，將工作轉化成一個個獨立的 MVC 程式模組，所有 MVC 模組的開發方法相同，目錄結構相同，如果要進行 MVC 模組的分工，由多人分頭開發，也不會造成整合上的問題，美工設計人員及程式設計師也可以獨自進行份內的工作，再整合完成程式功能，減少了專案的溝通及維護的負擔。

將所有的需求及功能轉化成一個個的 MVC 模組時，所有 MVC 模組的

Model、View 及 Controller 程式檔案的目錄結構安排，都是相同的，不用花時間熟悉程式檔案的規劃及安排，同一個專案的所有 MVC 程式模組，程式檔的安排都遵循一定的規則，當需要修改程式碼或功能時，能夠很容易找出是哪一個 MVC 模組需要修改，並找出要修正的程式碼。

當有新的需求，需要新增功能，只需複製現有的 MVC 模組架構，專注於程式邏輯及使用者介面的程式碼撰寫，以現有的 MVC 基礎架構進行功能及需求的建構，就能與原有的 MVC 程式模組整合，而不是將程式碼散亂地分散在各目錄或檔案裡頭，毫無章法，造成維護及整合的負擔。

2.2.2 眾多成熟的 MVC 框架可供運用

MVC 設計模式只是一套軟體開發的觀念，要落實到實際的專案，就必須有一套 MVC 開發框架，作為開發的輔助工具，MVC 開發框架的功能就像是建地的鷹架與模板一般，加速專案的開發，確保專案的品質，提供程式設計師穩定且容易整合的架構，進行專案的開發。

一套良善的 MVC 開發框架，要耗費大量的心力，本論文的專案開發以 Web 環境作為開發平台，須要一套成熟穩定的開發框架作為輔助，而一套成熟穩定的 MVC 開發框架必然要付出大量的時間進行思考、分析及實作，如果開發框架常變動，上層的 MVC 模式程式碼必然也要隨之變動，抵消了 MVC 模組化的優勢，自行開發的 MVC 框架，初期的穩定性不高，網路上已有其他高品質的 MVC 框架，不須另外耗費時間開發。

本論文使用 PHP 程式語言進行專案的開發，PHP 上的 MVC 開發框架眾多，CakePHP、CodeIgnitor、Symfony 及 Zend Framework 都是成熟穩定的 MVC 開發框架，本論文選用 Zend Framework 作為 MVC 的開發框架，所有程式碼都以穩定成熟的 MVC 架構為基礎，以 Web 作為執行環境，進行薪資系統的開發。

2.3 Zend Framework

Zend Framework[27]是由 PHP[18]程式語言撰寫，使用物件導向方式開發的開發框架，Zend Framework 類別間的藕合性低，彈性變換的組合性高，可以任意替換其中的類別，改用自行開發的類別或是其他社群開發的類別，這種任意組合變換的彈性，讓 Zend Framework 很容易融入專案當中，專案中可以完全採用 Zend Framewok 定義的 MVC 架構，進行開發，或是只採用部份的 Zend Framework 函式庫，採漸近地方式，逐漸發揮 Zend Framework 框架的完整功能。

2.3.1 Zend Framework 提供架構完整的 MVC 框架

Zend Framwork 的 MVC 開發框架分為 Controller、Model 及 View 三大類別，其中以 Controller 架構最為複雜，負責整合 Model 及 View，再將結果傳送給使用者，Controller 物件的名稱，開頭都是 Zend_Controller，是整個 MVC 框架的核心類別，所負的任務最為龐雜，牽址的類別數量繁多，也是 MVC 框架的靈魂所在。

2.3.1.1 Controller 類別

使用者在瀏覽器上輸入網址，送出瀏覽要求後，Controller 類別即負責使用者整個瀏覽歷程的流程控制，所有的使用者要求，不論網址為何，只要是同一專案的要求，都會導向 Controller 類別，由 Controller 類別集中控制瀏覽的歷程，這種中央集權式地控制方式，可以由 Controller 類別統一調度所需的所有類別，這種軟體設計觀念，就是所謂的 Front Controller 設計模式。Controller 的詳細說明請參閱 Appendix A2。

2.3.1.2 Model 類別

Zend Framework 的 MVC 框架中的 Model 類別由使用者自訂，Controller 中可引入使用者自訂的 Model 類別，非常地有彈性，而不是硬性規定使用方式，讓程式設計師有發揮的空間，Model 類別放置工作運作及資料存取的邏輯，程式設計師的工作核心就是在 Model 類別裡頭撰寫適當的邏輯，完成工作任務。

2.3.1.3 View 類別

MVC 框架中預設使用的 View 類別是 Zend_View，由 Controller 類別整合 Model 及 View，輸出結果，使用者並不一定要使用 Zend_View 作為 MVC 框架中的 View，可以選用其他的套件，如 Smarty，替換預設的 View，Zend Framework 具有強大的彈性，當中的類別都可以互相替換或改寫，滿足開發的需求。

2.3.2 Zend Framework 提供豐富完整的函式庫

MVC 開發框架提供程式開發的架構，解決程式開發上維護的難題，但程式開發的需求是多面向的，有許多的開發需求需要被滿足，為提供完整的解決辦法，Zend Framework 有數量龐大的函式庫可以滿足開發上的各種需求。

Zend Framework 的 Zend_Auth、Zend_Acl 可以滿足開發上，認證及授

權的開發需求，針對資料過濾的開發需求，Zend_Filter、Zend_Validate 可以解決這方面的問題，數量龐大的各種函式庫可以滿足程式開發上各種不同的需求。

Zend Framework 函式庫可以搭配 MVC 框架使用，也能單獨抽離，應用在不同的個別專案裡，外部的函式庫也能引入 MVC 框架中使用，不受束縛，Zend Framework 函式庫具有任意組合的高度使用彈性。

2.4 XML

XML 是 eXtensible Markup Language 的英文字母縮寫，是一種標籤語言，XML 的內容都是純文字內容，不論程式語言或是軟體都易於讀取，不需特殊製作的客製化軟體，或是特別設計的程式，就可以讀取 XML 的內容，這種特性讓 XML 文件內容易於解讀，降低 XML 文件的流通障礙。本論文使用 XML 作為異質系統交換資料的共通資料格式。

XML 的標籤是成對的，標籤必須成對出現，將資料包夾在成對的標籤中，一對標籤包夾一項資料，以標籤分隔每項資料，而標籤的名稱則由 XML 文件的建立者自行訂定，由標籤名稱，即可推斷資料的用途及意義，讀取 XML 文件內容很容易即由文件內容明白 XML 文件代表的意義。

2.4.1 XML 依據資料意義設定標籤名稱

標籤語言依據不同的目的及用途，而被發展及應用，不同的標籤語言所具有的特性及應用範圍也因此相異，依據需求選用適當的標籤語言，才能發揮標籤語言的功能，滿足程式開發及使用上的需求。

Web 程式開發上，常會使用 HTML 標籤語言，不同的 HTML 標籤代表不同的顯示結構，這些顯示結構，可能是段落、條列顯示、表單或是表格等方式呈現，HTML 標籤依據顯示結構的不同，也就有相對應的固定標籤名稱，由於顯示結構的數量有限，HTML 的數量也有限制，HTML 此種呈現資料的特性，應用範圍被侷限於顯示資料的結構。這種特性使 HTML 標籤的變動性高，同一項資料，可能因為排版的需求，而不斷變更 HTML 標籤。

XML 標籤的數量並無限制，也沒有預設的標籤名稱，而是由建立 XML 文件的使用者，針對資料的用途及意義，使用合宜的標籤數量，自行訂定適當的標籤名稱，此種特性，使 XML 標籤，能夠由標籤名稱即可明白資料的意義及用途，而標籤名稱根據資料的意義訂定，也使 XML 標籤名稱的穩定性高，變動性少。

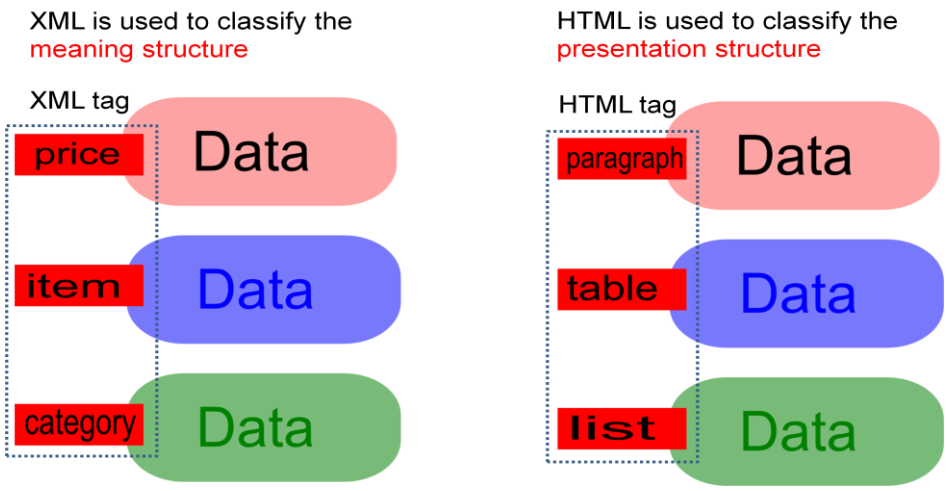


圖 2-5：HTML 及 XML 標籤的資料分類方式比較圖

圖 2-5 針對 XML 及 HTML 標籤資料的分類方式進行比較，以資料的意義分類作為分類依據時，資料被區分成價格、品項及類別三類，分別套上 price、item 及 category 的 XML 標籤名稱，以資料呈現結構作為分類方式時，就將資料分別套上 HTML 標籤的段落( )、表格(<table></table>)及條列的標籤(
)，以不同的呈現結構顯示資料。

由於 XML 標籤依據資料的意義，由使用者自訂標籤名稱，因此 XML 標籤的穩定性較高，較少變動，也容易由標籤名稱推斷資料的意義，不會因為需要更動顯示結構，而更動標籤名稱，或更動標籤名稱時，造成對資料內容的誤解，有資料交換需求時，XML 標籤語言穩定的特性，不會因網頁結構的變動，更動標籤名稱，影響資料交換的進行，非常適合作為異質 Web 系統間的資料交換語言。

2.4.2 XML 具有跨軟硬體平台的中立性

XML 文件內容由純文字組成，純文字內容不論是在哪種平台下都非常容易讀取，不必受限於特定的軟硬體才能讀取內容，這種特性使 XML 文件能夠很容易的在不同平台間傳遞資訊內容，消弭資料傳遞的障礙。

假設使用特定的資料庫儲存資訊內容，由於資料都是存在資料庫中，必須與資料庫建立連線，熟悉建立連線及存取資料的方法，而每種資料庫的連線方法不同，必須花費時間明白不同資料庫的連線方法，才能存取資料，一旦要整合不同的資料庫內容，就得使用不同的方法建立不同的資料庫連線，再將內容整合，這種方式與資料庫平台的相關性太緊密，使用門檻高。

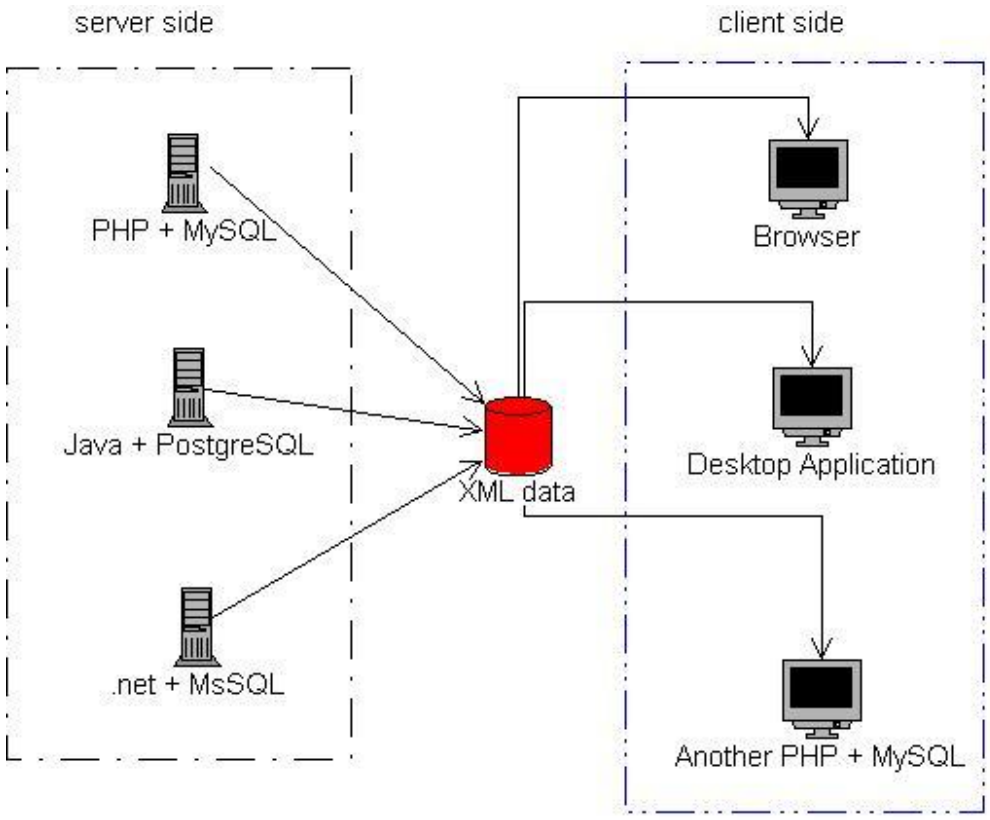


圖 2-6：XML 資料的平台中立特性圖

不同的開發團發擅長的程式開發語言及資料庫種類並不相同，圖 2-6 中，應用程式搭配不同的資料庫，進行開發，分別使用 PHP 程式語言存取 MySQL 資料庫、Java 程式語言配合 PostgreSQL 資料庫、.net 程式語言與 MsSQL 資料庫配合，三套使用不同程式語言及資料庫開發的應用程式，使用 XML 文件格式，將資料以 XML 文件的型式，傳送到 XML 資料中心，由於 XML 具有跨平台中立性的特性，XML 資料中心取得三套應用程式的資料方法都是相同的，不須連接不同的資料庫，考慮開發平台的異質性，而 XML 資料中心的資料，可以供桌上型軟體、瀏覽器或是其他應用程式存取，而存取的方法具有一致性，減少資料交換的阻礙。

本論文運用 XML 具有與軟硬體平台無關的中立特性[26]，建構 XML 資料交換中心，透過 XML 資料交換中心，進行人事系統及薪資系統的資料交換，不須考量底層實作的程式語言或是資料庫種類，而能讓各平台及應用程式，使用一致的方式，各自運用平台的程式語言或軟體，存取異質系統的資料，達成異質系統整合的目的。

2.5 以角色為基礎的存取控制(Role-Based Access Control)

以角色為基礎的存取控制機制[23]，是以角色作為權限控管的中心，所有權限的分派及控管均透過角色進行配置，使用者不直接被付予權限，而是被指定為特定的角色，由使用者被分配的角色決定權限。圖 2-7 中，角色被付予 Permission_1、Permission_2 及 Permission_3 的權限，使用者 User_1 被指派為這個指定的角色，因此使用者 User_1 擁有 Permission_1、Permission_2 及 Permission_3 的權限，同理，使用者 User_2 及 User_3 因被指派為同樣的角色，而擁有相同的權限。

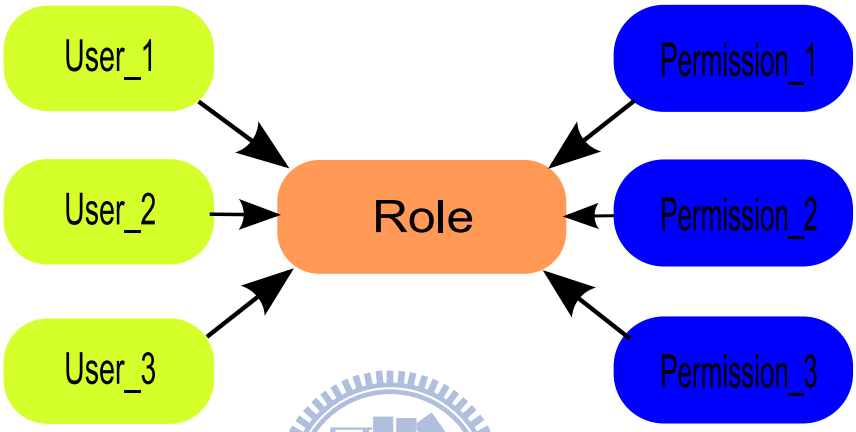


圖 2-7：RBAC 說明圖

2.5.1 RBAC 簡化管理的複雜度

以 RBAC 的方式分配權限給使用者時，不須將權限與使用者一一配對，而是直接指派一個角色給使用者[15]，使用者即可擁有該角色的權限，當更動角色的權限時，所有被指派為該角色的使用者權限，也隨之更動，毋須一一更動每位使用者的權限，簡化管理上的複雜度，降低操作上的難度。

2.5.2 RBAC 的權限名稱可自訂

RBAC 的權限名稱可以是檔案名稱、目錄名稱，或是特定的行為，如讀取、寫入等，完全由使用者自訂，這樣的彈性，很適合用於本論文的權限控管系統中，本論文以 MVC 模組名稱作為權限名稱，當擴增 MVC 模組時，權限名稱也能隨之擴增，強化系統的彈性。

2.6 jQuery(簡化 JavaScript 的開發難度)

jQuery[12]是使用 JavaScript 程式語言開發的開發框架，以各式瀏覽器作為執行的平台，開發的目的是為了簡化前端網頁程式的開發工作，滿足快速變動的開發需求。

雖然現今的瀏覽器均支援 JavaScript，但各瀏覽器上內建的 JavaScript 對於網頁元素的操作方式，仍有差異，單純使用 JavaScript，會造成同一套程式，須撰寫適用於不同版本的 JavaScript 程式，增加程式碼撰寫的負擔，而 jQuery 提供的開發框架提供一致的網頁元素操作方式，解決跨瀏覽器程式相容性的問題，使用 jQuery 撰寫的程式，能夠適用於不同的瀏覽器平台，不必重覆開發適用於不同瀏覽器的程式碼，使用 jQuery 開發的程式天生就具有跨越不同瀏覽器平台的基因。

jQuery 以 CSS 語法，選取網頁元素，再進行相關操作，操作邏輯非常直覺易懂，與 JavaScript 相比，以更簡明易懂的方式，選取網頁元素，程式碼更簡潔，以更少的程式碼，就可以撰寫功能相同的程式[12]。

第三章　PRMRX 建構法

PRMRX 建構法的架構及實作牽涉到許多的概念及觀點的轉換，本論文第二章已針對相關的背景知識提出關連性的介紹及探討，第三章將進一步針對 PRMRX 建構法的整體架構及轉換的演算法提供一套完整的理論架構說明。本章將 PRMRX 建構法中的個別觀點，如 Petri net 流程圖規劃、程式藍圖規劃、資料庫關連圖規劃、RBAC 機制的建立及 XML 資料交換的方法，作一系列系統性的說明，並以實際例子為說明對象，實際應用 PRMRX 建構法進行建構，將這些觀念分節詳細說明並整理出相關的演算法，實際轉化成相關的案例，讓 PRMRX 建構法的實作流程清晰呈現。

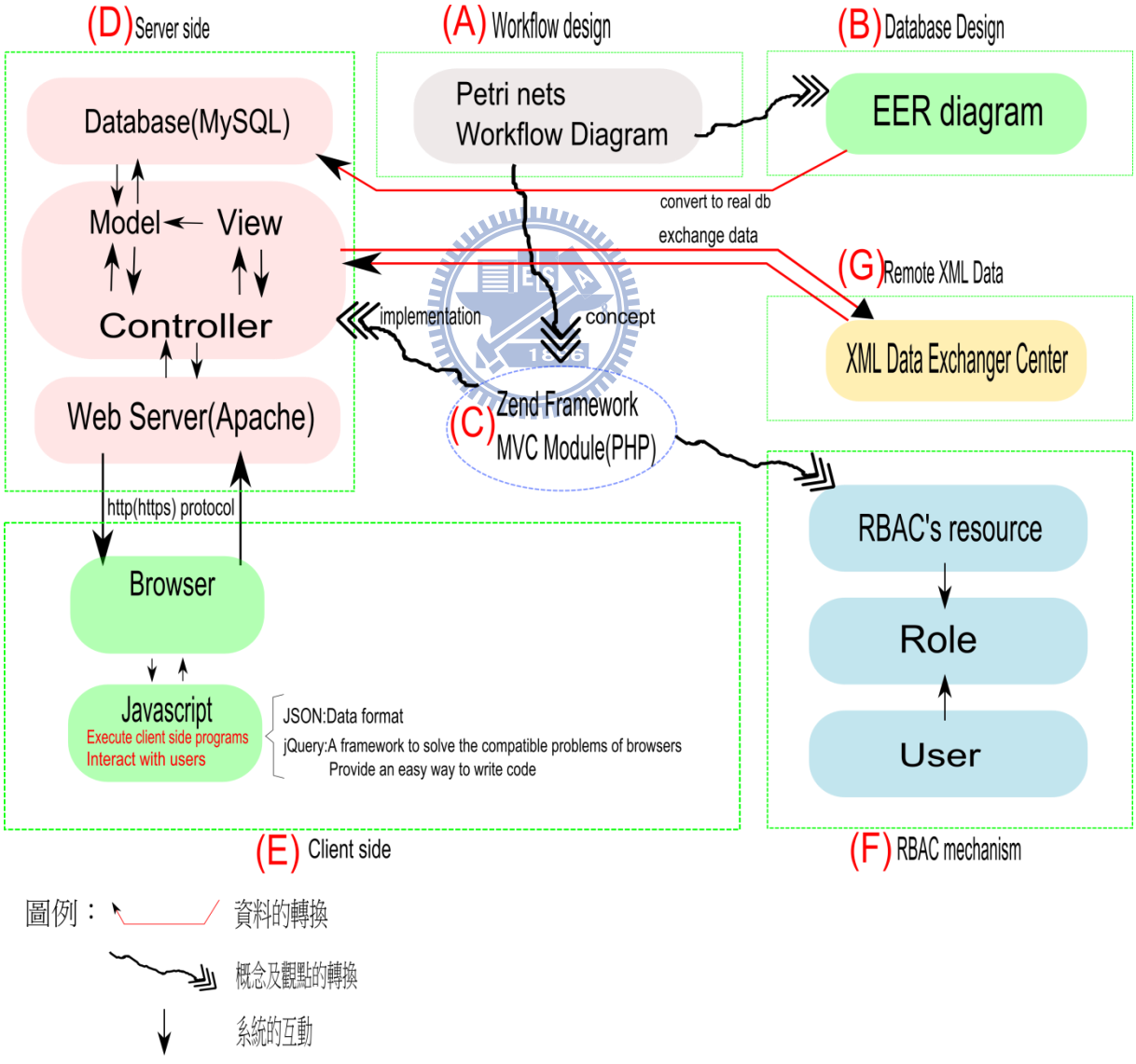


圖 3-1：PRMRX 建構法整體設計圖

3.1 系統架構

本小節探討的主題為 PRMRX 建構法的系統架構與演算法，主要內容以整體觀點出發，將 PRMRX 建構法中的 Petri net 流程圖、RDBMS 觀點、MVC 程式模組、RBAC 權限控管機制及 XML 資料交換中心的轉換流程及相互之間的關連作說明，並且在 3.1.7 這一小節完整介紹 PRMRX 建構法的關鍵概念及實作步驟（本節只作概念性介紹，3.2~3.7 節將詳列細節）。

3.1.1 Petri net 觀點(圖 3-1 中的 A)

PRMRX 建構法的設計流程及整體架構如圖 3-1，起始作業以資源配置及工作執行的觀點規劃作業流程，將工作流程切割為一個個細項的工作及工作所需的資源，再以 Petri net 圖形中的 transition 及 place 分別表示工作及資源，適當配置 transition 及 place 的先後順序及位置，將作業流程轉化為清晰、定義清楚的 Petri net 作業流程圖。

一般而言，Petri net 作業流程圖規劃完成後，尚無法確認流程能夠正確無誤的運作，為驗證流程的合理性，本論文提出的 PRMRX 建構法利用 Petri net 圖形能夠精確表達各種狀態的特性，透過圖中 token 分佈狀態，確認 Petri net 流程圖各種狀態的合理性，逐一修正不合理的狀態，直至通過驗證。通過驗證後的 Petri net 流程圖，為後續 PRMRX 建構法的規劃依據及根本基礎。

相較於以物件導向(OOAD[17])的觀點進行規劃，採用流程觀點規劃的優勢[2][3][5]為：

(1)容易理解

(2)容易與使用者溝通

(3)儘早發掘遺漏的需求，避免需求暴增

(4)發掘需求之間的關係，避免需求不一致

3.1.2 資料庫觀點(圖 3-1 中的 B)

圖 3-1 中，Petri net 工作流程圖規劃完成後，接下來透過資料庫觀點規劃資料庫關連圖。

實務上，為有效率的存取系統所需要的資料以及所產生的資料，通常在系統設計時，都會透過資料庫來存取資料，透過標準化的 SQL 語法即能

存取大量資料，簡化存取資料的方式。一般而言，Petri net 流程圖(如圖 2-1)中的 place 表示工作流程中工作所需的資源，或是工作完成後產生的各種數據及資源，利用資料庫的觀點規劃相關的資源存取，能夠符合程式撰寫的需求。

為更進一步的簡化系統設計，因此我們將各式各樣的資料以一致性的方式，安排放入資料庫中，統一存取資源。因此這時我們將觀點轉換為資料庫規劃的觀點，將 Petri net 中的 place 所代表的資源轉化為相關的資料庫關連圖，再轉換為真實的資料庫結構，使 PRMRX 建構法實作的程式邏輯能有一個統一存取資源的方式。

3.1.3 角色的觀點(圖 3-1 中的 C)

複雜的作業流程中，多人分工以分攤流程中的不同工作，為一常態，PRMRX 建構法為滿足多人協同作業及資訊安全的需求，以 RBAC 中的角色觀點將 Petri net 作業流程圖中 place 及 transition 代表的資源及工作群組化為許多獨立的區塊，每一區塊分別代表不同角色負責執行的工作，這些區塊即為 PRMRX 建構法中概念上的 Zend Framework MVC 程式模組，已經劃分 MVC 程式模組的 Petri net 作業流程圖，即為 PRMRX 建構法的程式設計藍圖。

套用現有的 MVC 框架與使用 OOD 從頭進行設計的方式比較，具有以下優點：

(1)使用已開發完成的框架，不必從頭開始，容易實作系統

(2)與 OOD 設計方式相同，具有容易維護修改並重用的優點[13]

(3)OOD 設計方式可整合進 MVC 當中的 Model，發揮 OOD 設計的優勢

3.1.4 程式邏輯及畫面流動觀點(圖 3-1 中的 D 及 E)

PRMRX 建構法程式設計藍圖中的 MVC 程式模組中包含了 Petri net 流程圖的 place 及 transition，place 與資料庫對應，transition 代表的工作則由程式設計師轉化成程式邏輯，分別實作 MVC 程式模組的 Model、View 及 Controller 程式碼，本論文的實作部份以 Zend Framework 程式開發框架開發，因此使用 Zend Framework 實作 MVC 的程式碼，MVC 的 View 部份則使用 jQuery 開發框架開發與使用者互動的程式碼。本論文提出的 PRMRX 建構法並不限定開發的程式語言及程式開發框架，這些開發語言及程式框架均可替換。

MVC 程式模組是 PRMRX 建構法中系統實作的最小單元，程式設計師負責將程式模組中的工作及存取資源轉化為程式邏輯，完成系統功能的設計，MVC 程式模組與程式語言無關，可以適用於各類平台及程式語言。

3.1.5 RBAC 權限控管模組(圖 3-1 中的 F)

PRMRX 建構法中的 RBAC 權限控管機制與作業流程的關連度不大，因此，獨立於 Petri net 作業流程設計之外，實作一套獨立於流程規劃之外的 RBAC 權限控管模組。RBAC 權限控管模組，將 MVC 模組名稱轉換為 RBAC 的權限名稱，以角色、使用者及權限名稱的觀點，規劃 RBAC 權限控管模組的資料表，實作 RBAC 權限控管模組，以便控管 Petri net 作業流程中的作業權限，實現多人協同作業的需求並兼顧系統多人使用時的安全性。

3.1.6 XML 資料交換中心(圖 3-1 中的 G)

Petri net 流程圖中規劃的各 MVC 程式模組及 RBAC 權限控管模組實作完成後，系統為單獨運作，無法與異質系統整合，為能夠順利整合異質系統間的資料交換，PRMRX 架構法提出建置 XML 資料交換中心的作法，可以解決這方面的困擾，藉由 XML 資料格式跨平台的資料交換能力，實作一獨立的 MVC 模組，與 XML 資料交換中心進行資料交換，藉以整合異質系統。

3.1.7 PRMRX 建構法演算法

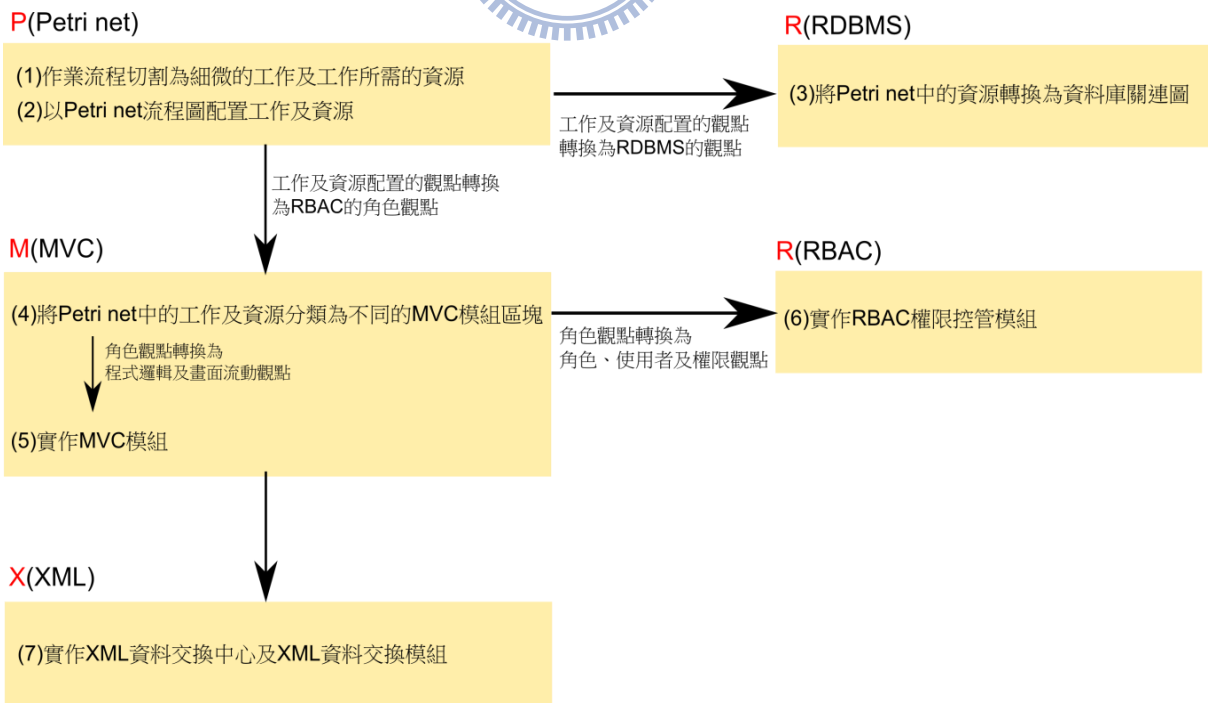


圖 3-2：PRMRX 建構法演算法

PRMRX 建構法提供的框架，將流程轉變為確實可上線的程式邏輯，彼此環環相扣，互有關連，消弭空有作業流程圖，卻無法據以實作，或是有強大的實作方法，卻沒有適當的程式藍圖作為依循，造成實作上沒有方向性的問題[8]。本論文提出的 PRMRX 建構法可以有效消除這兩大類的困擾，並進一步解決權限控管及異質系統整合的問題。

PRMRX 建構法涉及不同的概念及觀點的轉換，這些轉換的演算法整理如圖 3-2，PRMRX 建構法的轉換步驟如下：

(1)以資源配置及工作執行的觀點將工作流程切割為細小的工作及工作所需的資源。(工作執行及資源配置的觀點)

(2)使用 Petri net 繪製工作流程中工作及資源交互配置的方式。(工作執行及資源配置的觀點)

(3)以資料庫規劃的觀點，將 Petri net 流程圖中的資源轉換為資料庫關連圖，再轉換為實際的資料庫結構。(RDBMS 觀點)

(4)以 RBAC 的角色觀點，將 Petri net 流程圖中的工作及資源分類為不同角色應該執行的工作區塊，再將工作區塊轉換為 MVC 模組。(RBAC 的角色觀點)

(5)以程式邏輯及畫面流動的觀點實作 MVC 模組的功能。(程式邏輯及畫面流動的觀點)

(6)將 MVC 模組名稱轉換為權限名稱，以角色、使用者及權限的觀點實作 RBAC 權限控管模組。(角色、使用者及權限名稱的觀點)

(7)實作 XML 資料交換中心及 XML 資料交換模組。(程式邏輯觀點)

PRMRX 建構法緊密結合 Petri net 流程圖、關連式資料庫、MVC 程式模組、RBAC 權限控管及 XML 資料交換機制的觀點，提供明確的步驟，轉換各種概念及觀點，使規劃及實作中，原本彼此關係鬆散的各個階段彼此環環相扣，互有關連，避免各自為政的狀況發生。

整合流程設計、程式設計藍圖、資料庫規劃、系統權限控管機制及異質系統資料交換需求的 PRMRX 建構法，提供一套標準化的方法及輔助用的框架，將作業流程，轉化為實際可上線運作的程式。

為簡化說明，本章以簡化的正職教師薪資計算流程為例，以圖 3-3 簡潔的作業流程進行說明，使 PRMRX 建構法易於理解並呈現完整的面貌。

3.2 Petri net 作業流程圖的規劃

PRMRX 建構法中的 Petri net 流程圖的規劃步驟整理如下：

(1)以資源配置及工作執行的觀點將工作切分為細小的工作及工作所需的資源。

(2)以 PRMRX 建構法的流程觀點，交錯配置工作及資源，以反應工作及資源的因果關係。

(3)以 Petri net 圖形中的 transition 代表執行的工作，place 表示工作所需的資源，規劃 transition 及 place 交替出現的順序，完成工作流程的規劃。

(4)以 token 的分佈狀態驗證流程的合理性並加以修正。

一般而言，網路資訊系統流程自動化的藍圖設計，涉及資源的配置、工作的執行、畫面的流動及程式的邏輯，這些複雜的因素，全部呈現在設計藍圖中，會變成難以理解，不易解讀。使用 Petri net 圖形，以 place 表示資源，而 transition 代表工作，運用 place 與 transition 交替出現的特性，規劃工作(以 transition 表示)與資源(以 place 表示)交錯出現的因果關係，適當安排工作及資源出現的順序及相互之間的關係，即可完成流程的規劃。

本論文提出的 PRMRX 建構法運用資源配置及工作執行的觀點，規劃作業流程圖，使整個流程的設計簡明易懂，而程式邏輯及畫面的變化則封裝在 MVC 程式模組裡頭，避免干擾流程的設計。一旦程式邏輯或是畫面流動的方式有所變動，能夠限縮影響範圍，以 MVC 容易修改維護的特點，快速反應修改需求，而不牽動流程的變動，影響整個作業流程的規劃。這就像房子的基礎架構是不常變動的，但內部裝潢及隔間卻常隨房主喜好，經常變動，PRMRX 建構法套用相同的觀點，進行流程及 MVC 模組的規劃，運用穩定不常變動的資源配置及工作執行觀點規劃流程，而較常變動的程式邏輯及畫面流動則封裝在 MVC 模組裡。

另一方面，Petri net 流程圖的驗證是非常重要的，不正確的藍圖將導致不正確的實作，失去 Petri net 流程圖的應用價值。至於驗證的方法，將於底下舉實例說明。

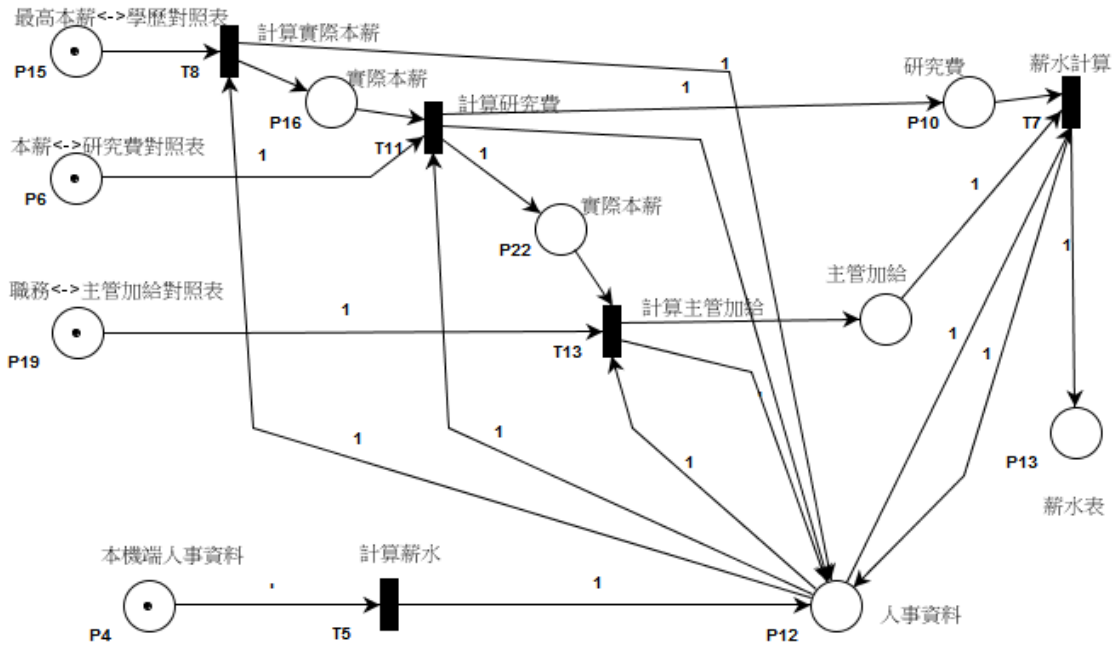


圖 3-3：以資源配置及工作執行的觀點規劃作業流程說明圖

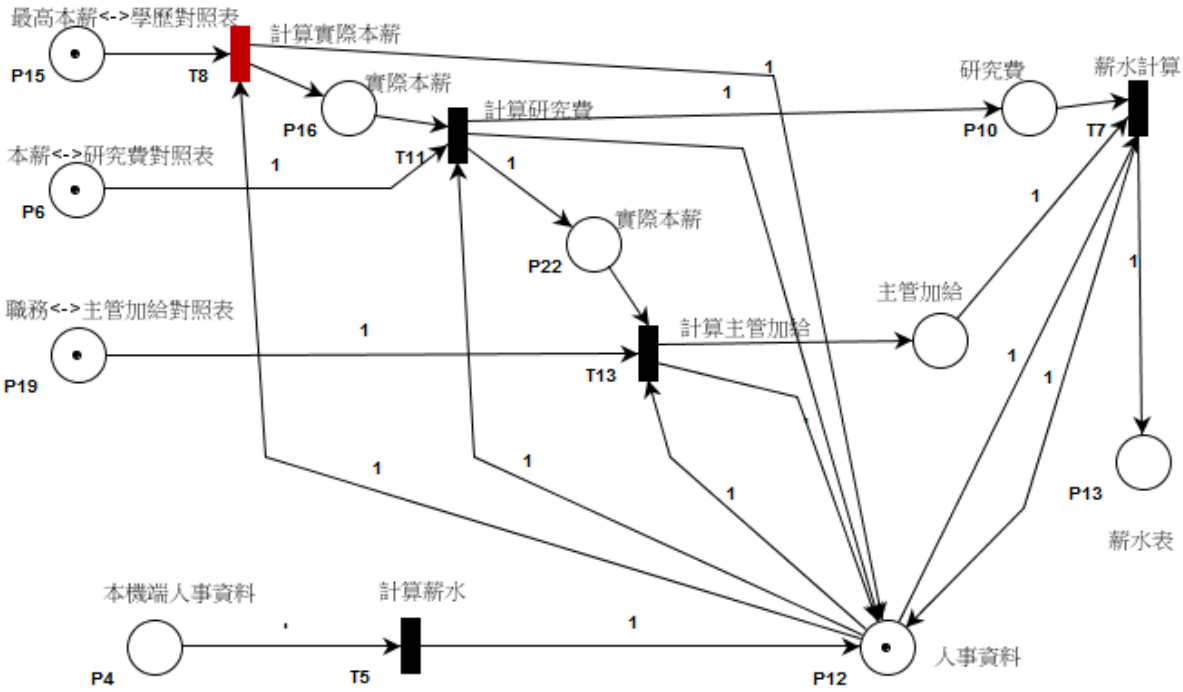


圖 3-4：流程驗證說明圖（1/5）

圖 3-3 以工作執行及資源配置的觀點規劃正職教師研究費及主管加給的薪資計算流程圖。將工作細分為計算實際本薪、計算研究費、計算主管加給、計算薪水及薪水計算五項工作，並依據工作的需要，配置各項資源，使用工作(以 transition 代表)及資源(以 place 代表)交錯配置的方式規劃作業流程。本論文以 token 代表資源是否準備妥當，若 place 內有 token 存在，則代表該 place 代表的資源已備妥，若否即表示缺乏該項資源，若一個 transition 的所有 input place 都有 token 存在，表示該 transition 代表的工作可被執行，一旦執行，token 即會轉移到所有的 out place，藉著 token 的分佈狀態，即可驗證流程的合理性，作為修正流程的依據。

圖 3-4 到圖 3-8 以 token 的移轉狀態，驗證圖 3-3 中的流程為正確合理的，同樣的方法，可以驗證其餘的薪資流程是正確的，確認整個流程沒有任何問題。

圖 3-4 中，token 停留在 P12 及 P15 中，T8 是 enabled 的狀況，T8 被 fired 後，token 移轉到 P16 中，轉變為圖 3-5 的狀態，圖 3-5 中的 T11 被 fired 後，token 移轉到 P22 及 P10 中，轉變為圖 3-6 的狀態，token 最後會停留在 P12 及 P13 中，轉變為圖 3-8 的狀態。

同樣的方式可以驗證其餘運用 PRMRX 建構法規劃的流程，如果發現流程有錯誤，可以修正流程，再進行驗證，直到流程正確沒有謬誤。

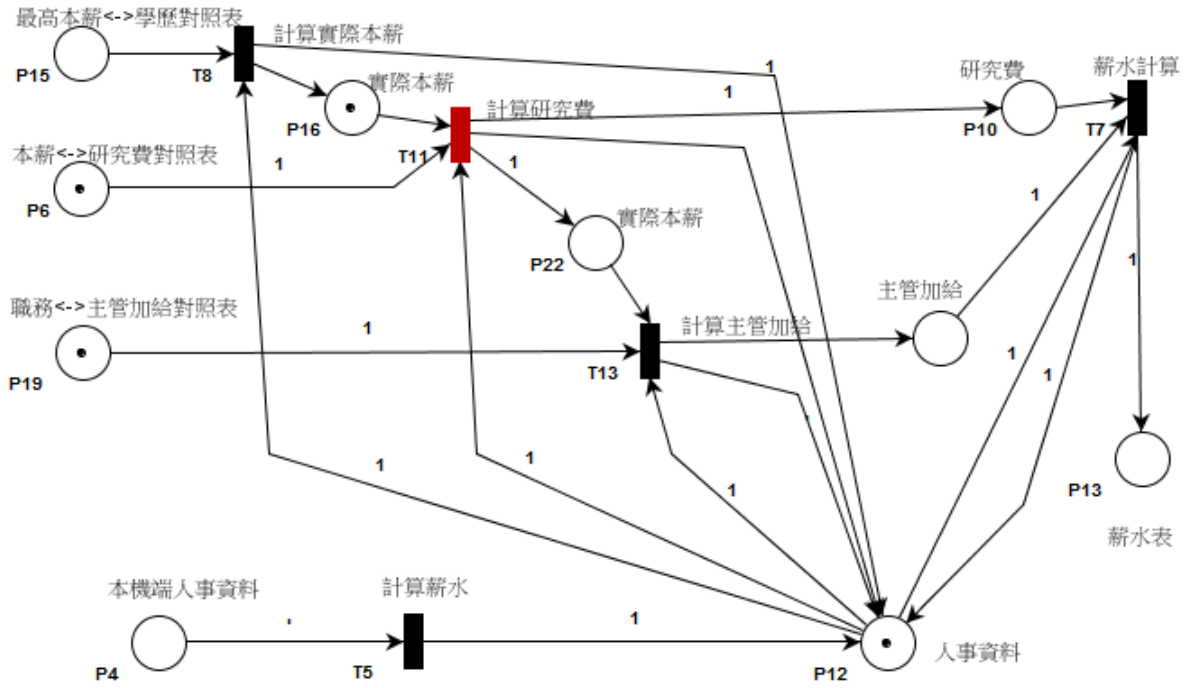


圖 3-5：流程驗證說明圖（2/5）

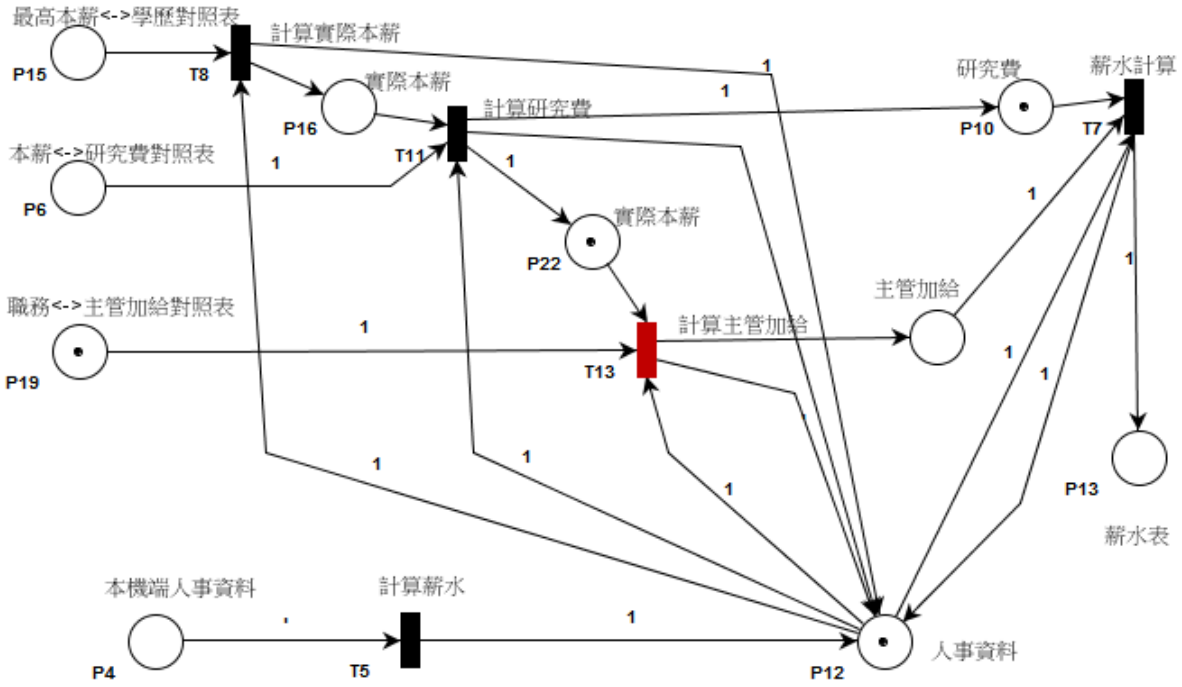


圖 3-6：流程驗證說明圖（3/5）

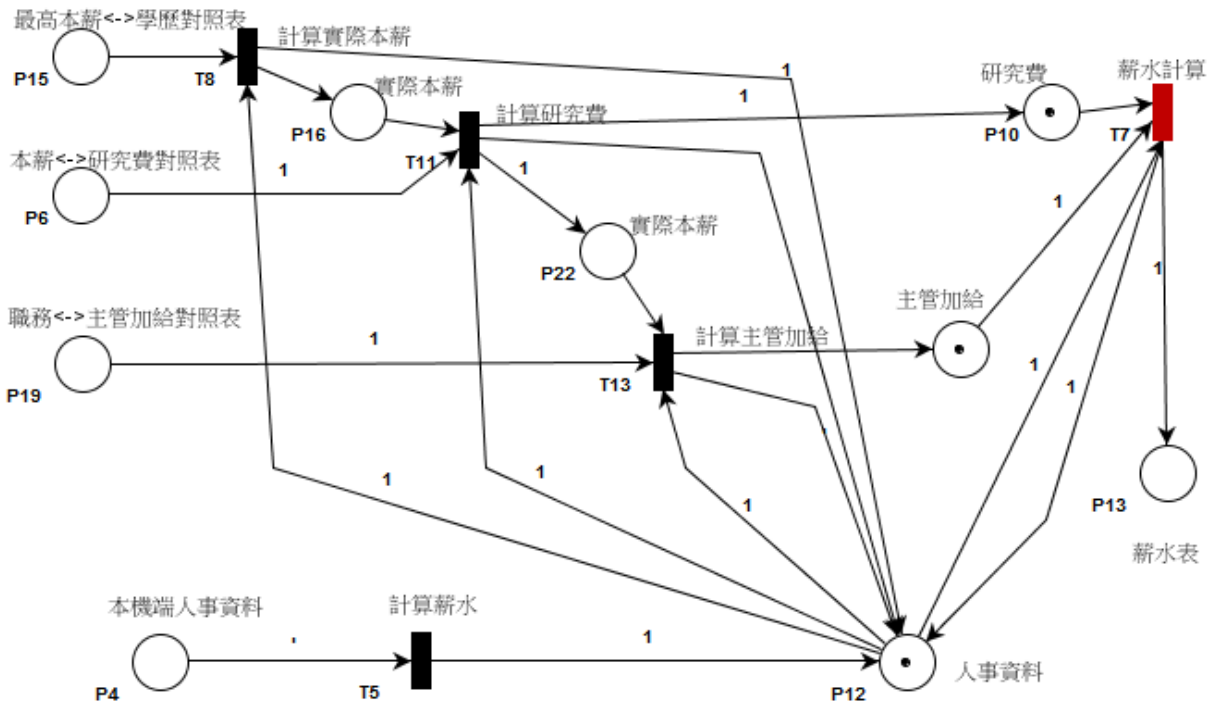


圖 3-7：流程驗證說明圖（4/5）

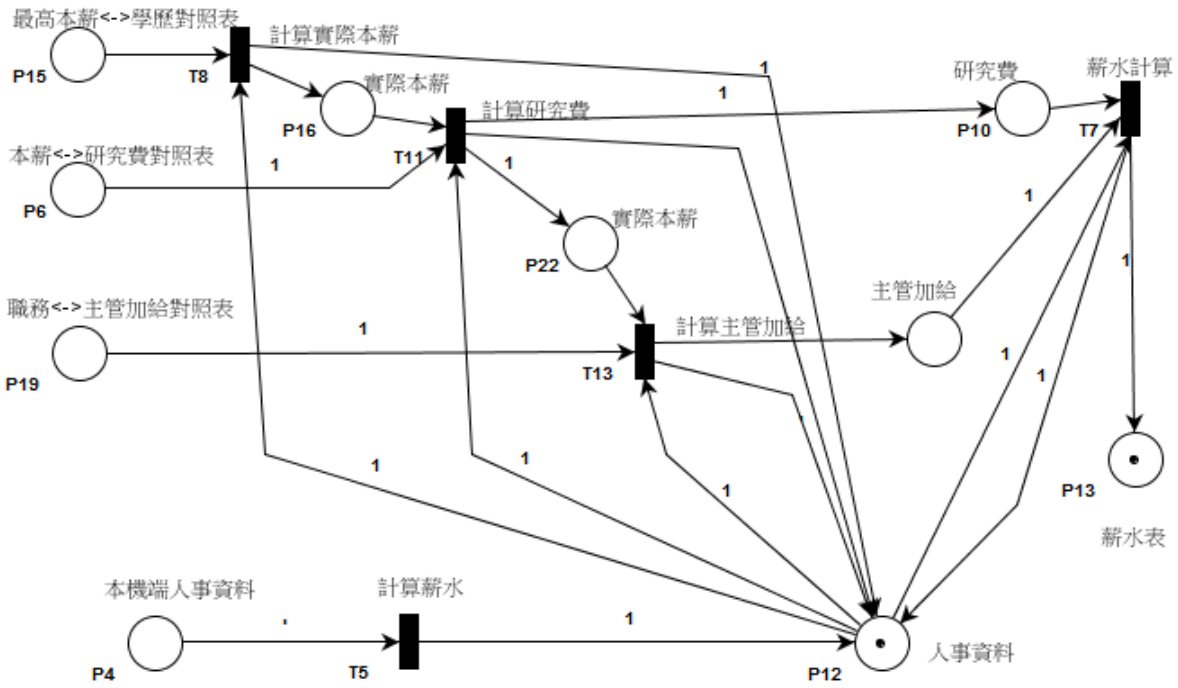


圖 3-8：流程驗證說明圖（5/5）

3.3 資料庫的規劃

PRMRX 建構法的資料庫規劃步驟：

(1)將 Petri net 流程圖中的資源區分為永久性資源及暫時性資源。

(2)將永久性資源轉換為資料表及資料表中的欄位，並以資料庫規劃的觀點設計資料表的結構及資料表之間的關連性，完成資料庫關連圖的實作。

(3)轉換資料庫關連圖為實際的資料庫結構。

PRMRX 建構法以 Petri net 規劃作業自動化的流程圖，place 代表的資源，通常區分為二類，一類是永久性資源，另一類是暫時性資源。永久性資源儲存在檔案、資料庫等永久性存放的儲存媒介中，暫時性資源則儲存在記憶體等暫時性儲存的媒介中。

永久性資源，以資料庫規劃的觀點，依據工作的需求及執行的結果，轉化為資料表及相關的資料表欄位，建立資料表之間的關連性，據此繪製資料庫關連圖。

例如圖 3-9 作業流程中的永久性資源「人事資料」必須包含教師的學歷及本薪，計算實際本薪時才能根據「人事資料」中的學歷與本薪及「最高本薪<->學歷對照表」計算出實際本薪。

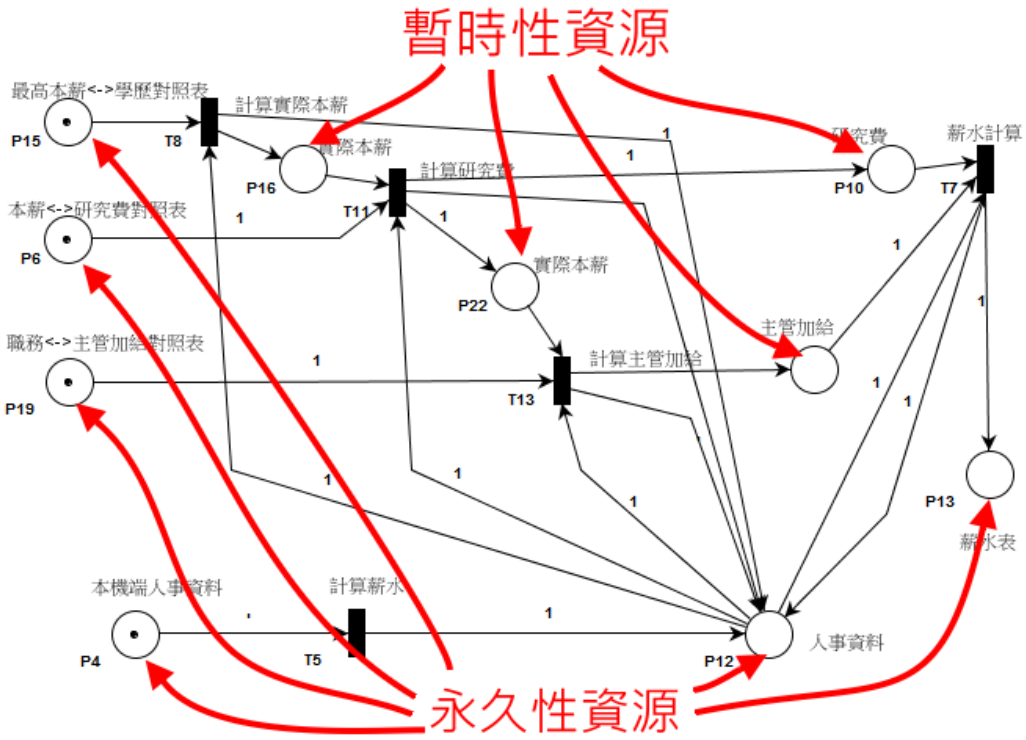


圖 3-9：資源轉化為資料庫關連圖說明圖

暫時性資源儲存在記憶體等非永久性儲存媒體中，程式設計師依據需求，設計相對應的儲存結構，由程式存取資料。

永久性資源一一轉變為相對應的資料表及欄位,建立資料庫的關連圖，再將資料庫關連圖轉化成實際的資料庫結構。

3.4 MVC 模組的規劃

PRMRX 建構法的 MVC 模組規劃步驟：

(1)以 RBAC 的角色觀點，將流程執行時的相關角色列出。

(2)將 Petri net 流程圖中與特定角色相關的 transition 及 place 設定為相同的群組。此時群組化的區塊為一個個概念上的 MVC 程式模組。

(3)規劃出 MVC 程式模組的 Petri net 流程圖即為 PRMRX 建構法的程式設計藍圖。

Petri net 工作流程圖如果較為複雜，牽涉到的工作及資源繁多，單獨一人通常難以負荷全部的工作，此時就須以分工合作，多人協同作業的方式完成整個作業程序。

PRMRX 建構法以 RBAC 權限控管的角色觀點，將流程中涉及的工作角色列出，依據角色的分工，將流程圖中的 place 及 transition 群組化為一個個範圍較小的區塊，由不同角色負責不同區塊的工作，解決分工合作的問題。

流程圖中群組化後的一塊塊區塊,除了代表不同角色執行的工作區塊，也是概念上的 Zend Framework MVC 程式模組，由不同角色負責各個 MVC 程式模組的執行。規劃出 MVC 程式模組的 Petri net 流程圖即為 PRMRX 建構法的程式設計藍圖，為後續 MVC 程式模組的實作依據。MVC 程式模組並不限定使用某一特定程式語言或平台，可以運用其他程式語言或程式開發框架替換本論文使用的 Zend Framework 及 jQuery。

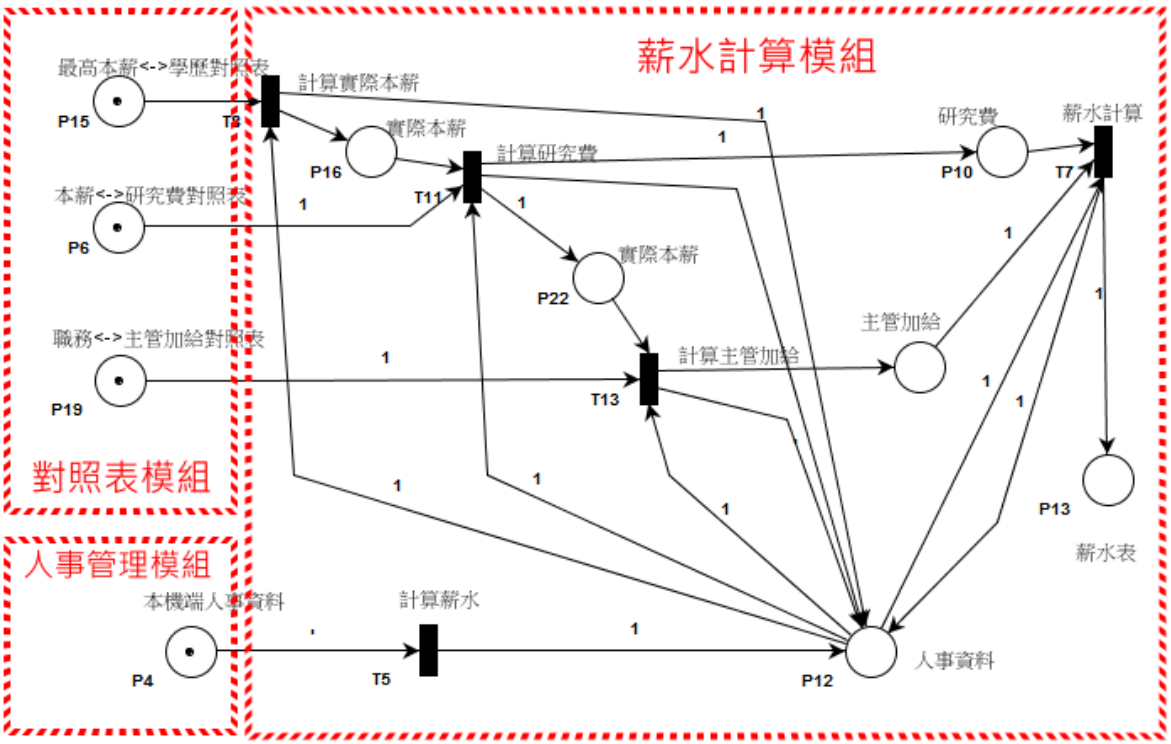


圖 3-10：Petri net 流程圖轉化為 MVC 程式模組說明圖

舉例而言，圖 3-10（從圖 3-3 轉化而來）說明將 Petri net 流程圖轉化為 MVC 程式模組的方法，首先，將作業流程中涉及的角色(出納、人事)列出，再依據角色的分工，群組化流程中的工作及資源。各角色負責的模組區塊如下：

(1)出納：薪水計算模組

(2)人事：對照表模組、人事管理模組

圖 3-10 即為一份 PRMRX 建構法的程式設計藍圖，整份藍圖由三個 MVC 程式模組組成，程式開發人員依據此份藍圖分別設計三個 MVC 程式模組，完成系統的設計。

3.5 MVC 模組的實作

MVC 模組實作步驟：

(1)列出模組、工作與資源的對應關係。

(2)決定模組的功能及畫面。

(3)實際實作。

表 3-1：MVC 模組轉化為程式邏輯說明表

模組名稱	功能	存取資源
薪水計算模組	(1)計算薪水 (2)瀏覽薪水	「薪水表」 「人事資料」
對照表模組	(1)「最高本薪<->學歷對照表」的新增、修改、刪除 (2)「本薪<->研究費對照表」的新增、修改、刪除 (3)「職務<->主管加給對照表」的新增、修改、刪除	「最高本薪<->學歷對照表」 「本薪<->研究費對照表」 「職務<->主管加給對照表」
人事管理模組	(1)「本機端人事資料」的新增、修改、刪除	「本機端人事資料」

程式邏輯及畫面流動並未出現在 PRMRX 建構法的程式設計藍圖中(已規劃 MVC 程式模組的 Petri net 流程圖)，而是由程式設計師依據藍圖，將一個個 MVC 程式模組，運用 Zend Framework MVC 開發框架，轉化為程式及畫面。Zend Framework MVC 開發框架，將藍圖中的 MVC 程式模組再分為 Model、View 及 Controller 三個區塊，程式邏輯放在 Model 及 Controller 中，由伺服器執行，負責工作的執行，而畫面則放在 View 中，以分工方式快速反應程式邏輯及畫面變動的需求。

View 中除了畫面外，也包含與使用者互動的程式碼，本論文利用 jQuery 作為瀏覽器端的程式開發輔助工具，開發與使用者互動的程式，讓使用者從伺服器上下載後，在瀏覽器中執行。

表 3-1（從圖 3-10 轉化而來）為各個 MVC 程式模組的功能及存取的資源。實作時，根據流程圖決定模組的功能及畫面，將相關的工作轉化成程式邏輯及畫面，完成程式的撰寫。

3.6 RBAC 權限控管模組

RBAC 權限控管模組的實作步驟如下：

(1)模組名稱轉換為權限名稱。

(2)規劃 RBAC 權限控管資料表。

(3)實作 RBAC 權限控管模組。

Petri net 流程圖中的 MVC 程式模組，由不同角色執行模組內的工作，藉此分工並分別管制 MVC 程式模組的執行權限，權限管控作業與原本的作業流程關連度低。因此，獨立於流程規劃之外，另外實作權限控管模組，確保多人協同作業的安全性。

表 3-2：RBAC 權限控管說明表

使用者名稱	角色名稱	權限名稱
王甲	出納	薪水計算模組、對照表模組
王乙	人事	人事管理模組

PRMRX 建構法將 MVC 程式模組名稱轉換為權限名稱，規劃權限控管的資料表結構，實際實作 RBAC 權限控管 MVC 程式模組，管控作業流程的執行。

表 3-2 依據圖 3-10，將 MVC 模組名稱轉換為權限名稱，再以表列方式列出角色、使用者及權限的對照表。依據表 3-2 規劃的資料庫關連圖及程式邏輯，實作 RBAC 權限控管模組，即可藉以控管流程中相對應的使用者能夠執行的 MVC 模組。

3.7 以 XML 資料交換中心整合異質系統

本論文以 XML 資料交換中心，作為不同的作業流程、不同的系統平台、實作方法相異的異質系統的資料交換中心，整合不同的異質系統。

XML 資料交換中心作為資料交換的集散中心，只負責資料的儲存及交換，不作任何的運算，能夠使用任何種類的資料庫存放資料，搭配使用的程式語言也不限定任何特定的程式語言，絕大多數程式語言均內建 XML 的 API，能夠輕易實作 XML 資料交換中心的程式邏輯。

底下將資料交換的方式分為同步與非同步，分別說明二者的資料交換方式及特性。

3.7.1 以非同步方式與 XML 資料交換中心交換資料

非同步方式使用人工管理方式，實作一獨立的 MVC 程式模組，由管理者負責操作，將資料庫中的資料與 XML 資料交換中心進行資料的交換，由於資料庫中的資料是由其他 MVC 模組運作得出的結果，並非立即更新的資料，因此 XML 資料交換中心的資料與現存系統中的資料並不是同步更新的。

本論文的薪資系統資料更新頻率低，因此以非同步的方式實作 XML 資料的人事資料交換機制，進行資料的更新。

3.7.2 以同步方式與 XML 資料交換中心交換資料

同步方式交換資料，是在各 MVC 模組內，實作與 XML 資料交換中心交換資料的程式邏輯，MVC 模組內的程式邏輯，一旦更新資料，也連帶更新 XML 資料交換中心的資料，達到資料同步化的目的，每個 MVC 模組均進行實作，實作上不容易集中控管資料交換的程式邏輯，實作上也較非同步方式複雜。

第四章　系統架構及實作環境

本章主要內容在於說明本論文所建構的薪資系統之建置環境及開發時使用的軟硬體工具。另一方面，為了避免網路上層出不窮的資安事件危害系統的安全性，我們也針對本論文設計的 Web 薪資系統，提出一些防護的方法。

4.1 建置環境

本系統執行的環境分成三套系統，其中二套為薪資系統，另外一套為 XML 資料交換中心。實際上，受限於實體機器資源的限制(只有一台實體伺服器)，本論文所有的應用系統均建置於虛擬化軟體 Xen Server 的基礎上，由於目前的硬體系統效能優異，且多數中小學的教職員工數不多，這樣的實驗配置足以模擬多數的學校作業需求。另一方面，採用虛擬主機技術也能加快系統的建置速度，提高維護的方便性，新增作業環境時，不再需要提供新的硬體，重覆安裝軟體，只需新增虛擬機即可。

虛擬化軟體使用 Xen Sever[10]的免費版本建置，圖 4-1 是實際建置的 Xen Server 執行效能圖。Xen Server 安裝在單台伺服器上，其上分別建置了三套獨立的 Ubuntu[24]作業系統，這三套系統的 ip 分別為 140.113.2.221、140.113.2.226 及 140.113.2.227，以一台硬體伺服器，即可建置傳統上必須使用三台伺服器才能建置的作業環境。

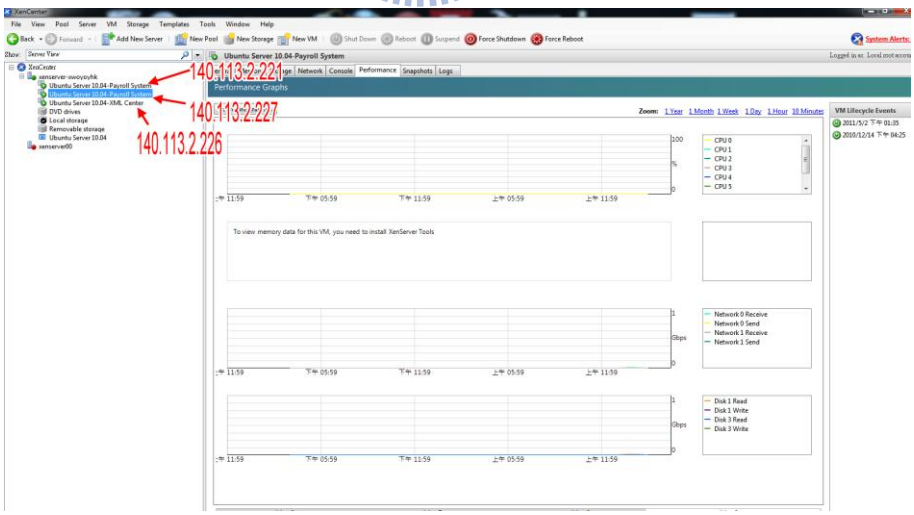


圖 4-1：Xen Server 執行效能圖

如圖 4-2 所示，Xen Server 安裝在一台實體機器上，本論文所建置的三套系統則建置在 Xen Server 上。在 Web 的執行環境下，Client 端只要有瀏覽器，不需安裝額外的程式，透過網路，即可連結 Server 端的薪資系統

及 XML 資料交換中心，只要實體機器的運算能力足夠，即可擴增虛擬機器的數量，增加服務的能力。

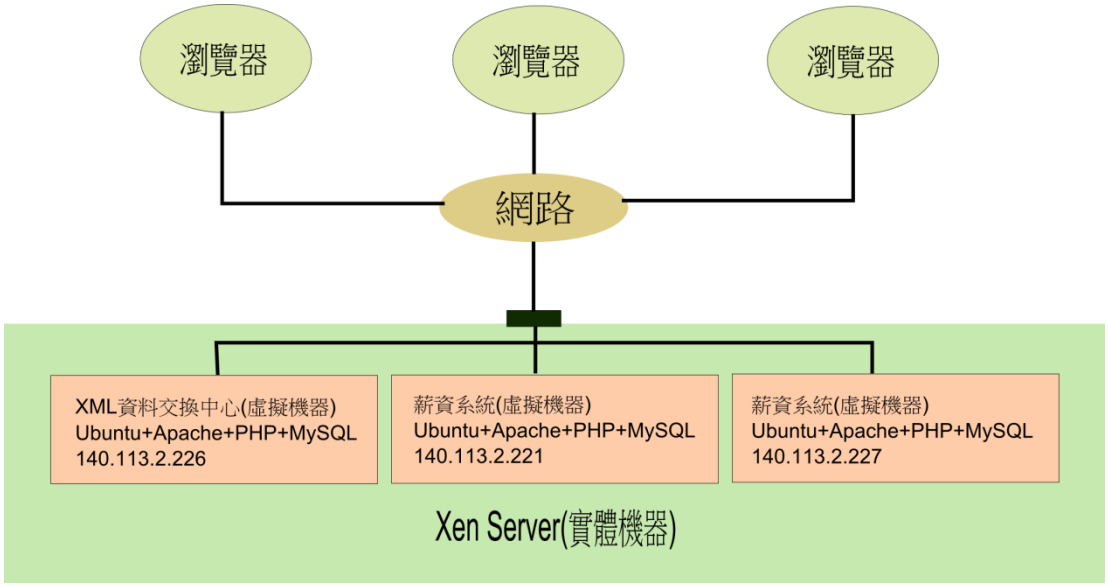


圖 4-2：實體環境建置圖

表 4-1：系統建置環境

建置項目	採用設備或軟體	版本或規格
硬體主機	IBM Server	2 x Intel Xeon CPU E5310 @ 1.6GHz 4GB RAM
虛擬化環境	Xen Server	5.6
作業系統	Ubuntu Server	10.04
網頁伺服器	Apache[9]	2.2
程式語言	PHP	5.3
資料庫伺服器	MySQL[14]	5.1
程式開發框架	Zend Framework	1.11
程式開發框架	jQuery	1.4

系統環境如表 4-1，虛擬機器上安裝的作業系統均為 Ubuntu Server，程式執行的環境為 Apache、PHP 及 MySQL，程式開發框架為 Zend Framework 及 jQuery，整體作業環境均由開放原始碼組成，系統擴增或增加使用人數時，不需擔心授權或法律問題，有利於後續的系統功能擴展，減少開發上的負擔。實際上系統使用的各軟硬體的細節及版本如表 4-1 所示。

4.2 軟體工具的使用

本論文建置系統所使用的軟體工具如表 4-2，分為四類：Petri net 流程圖規劃工具、程式設計開發編輯器、資料庫管理工具及開發環境工具。每種工具在不同的面向協助系統的設計及開發，提供更快速方便的方法建置系統。細節請參照表 4-2。

表 4-2：軟體工具列表

軟體類別	軟體名稱	用途
Petri net 流程圖規劃工具	Platform Independent Petri net Editor 3.0[20]	繪製並驗證 Petri net 工作流程圖。
資料庫管理工具	MySQL Workbench 5.2 CE	規劃 MySQL 資料庫關連圖並將資料庫結構匯入 MySQL 資料庫中。
資料庫管理工具	Navicat Lite 9.1.5	連接 MySQL 資料庫，並作資料結構的更改及資料的新增、修改、刪除。
程式設計工具	NetBeans 7.0	PHP 程式的整合開發環境，提供除錯及專案管理的功能。
開發環境工具	XAMPP	在 Windows 平台上提供 Apache、MySQL 及 PHP 的執行環境，增加開發上的便利性。

4.3 安全機制的建立

本論文建置的系統，在安全性的考量上，如圖 4-3 使用 Ubuntu Server 內建的 netfilter 防火牆功能，以 iptables 工具，設定防火牆規則[11]，禁止不當的存取，防火牆政策是除了允許的連入連線外，其餘的一律禁止。

第一道防線是防火牆，再來就是網頁應用程式的防護，網頁弱點沒辦法依靠防火牆防護，在程式撰寫時，除了留意安全性問題，以人工檢閱程式碼，避免安全漏洞外，另外使用 Paros[19]軟體，掃描網頁程式是否存有 SQL injection 及 XSS 等漏洞[16]，避免系統遭受此種類型的攻擊。

薪資系統內處理的資料牽涉到個人的隱私資料，為保護這些敏感性資料，避免資料竊取及外洩，本論文如圖 4-3 以 ssl 的方式實作，加解密這些資料，確保資料於網路傳輸的安全。

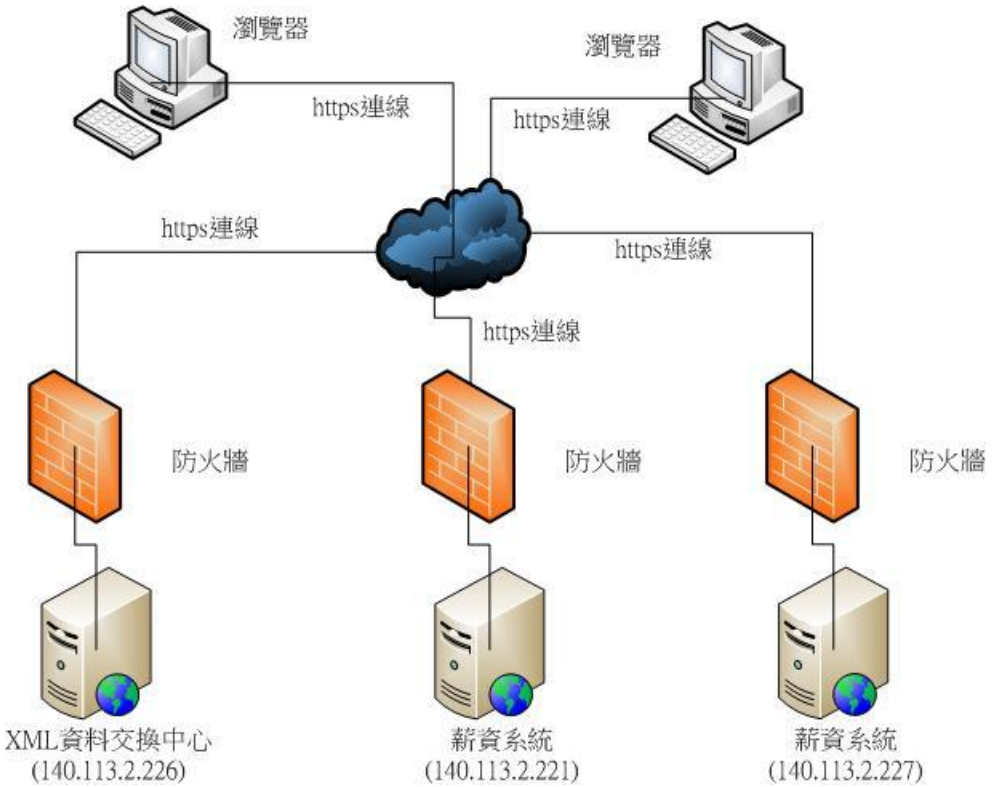


圖 4-3：系統的安全機制

另外，為了預防 robot 程式攻擊，本系統還特別增加數字驗證機制。個別用戶以特定帳號/密碼登錄時候，系統除了檢查帳號和密碼必須正確無誤外，用戶還必須正確輸入畫面上提示的數字驗證碼（如圖 4-4 所示），

三者比對正確無誤，才可以登錄使用。這個做法對阻絕 robot 程式攻擊，相當有效。

圖 4-4: 本論文所實作的中小學薪資實驗系統(登入畫面)

第五章　以 PRMRX 建構法建置中小學薪資系統

本章主要內容為以 PRMRX 建構法實作中小學薪資系統，驗證這套建構法的可行性，在複雜的薪資計算流程中，實際應用第三章的系統架構及整理出的演算法，轉換各種觀點，實作一套實際以 PRMRX 建構法建構的薪資系統：首先將薪資計算流程轉變成 Petri net 流程圖，再進一步，依次轉化為資料庫關連圖、MVC 模組、RBAC 權限控管模組，並實作 XML 資料交換中心。

薪資系統內處理的資料牽涉到個人的隱私資料(本章所舉校名、人名及欄位資料等均為虛擬的)，為保護這些敏感性資料，避免資料竊取及外洩，本論文以 ssl 的方式實作，加解密這些資料，確保資料於網路傳輸的安全。

5.1 規劃中小學薪資系統的 Petri net 流程圖

Petri net 作業流程圖描述的實際問題若過於複雜，有時會再進一步細分為二個以上的階段，分別繪製二張以上的流程圖，以利於解讀，避免 Petri net 流程圖過於複雜，造成判讀的障礙。例如本節以正職教師的薪資計算為例，可分成第一階段(圖 5-1)及第二階段(圖 5-2)，以相同的 place 出現在欲串接的二張流程圖中，將二張 Petri net 流程圖連貫起來。

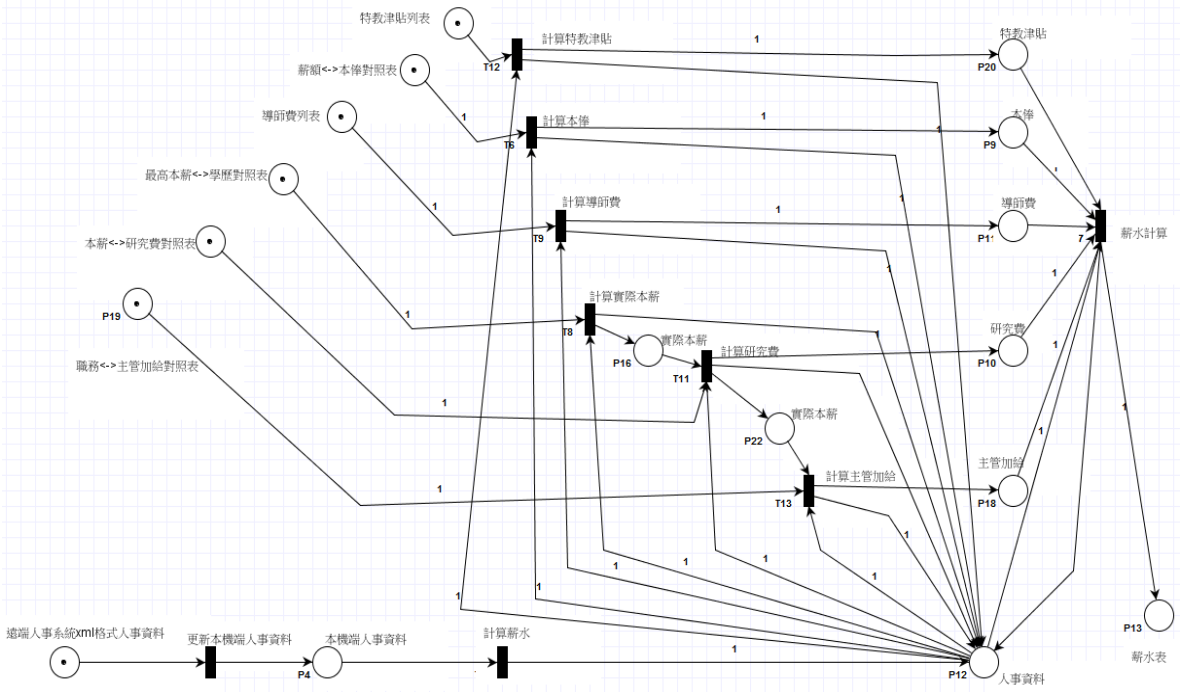


圖 5-1(第一階段)：正職教師的薪水計算作業流程圖(1/2)

如圖 5-1，首先根據 Petri net 流程圖規劃步驟，將正職教師的薪水計

算流程切分為更細微的工作（以 transition 代表）及工作所需的資源（以 place 代表），交替配置 place 及 transition 以完成正職教師的薪水計算流程圖。

圖 5-1 中的薪資流程的第一階段正確執行完畢後，人事資料(p12)及薪水表(p13)資源各留有一個 token，可供圖 5-2 中的工作流程運用。圖 5-1 及圖 5-2 以人事資料及薪水表資源作為流程銜接的接口，完整呈現正職教師計算薪水清冊的流程，計算出的薪水清冊再作為後續流程的存取資源，以供後續作業流程運用。

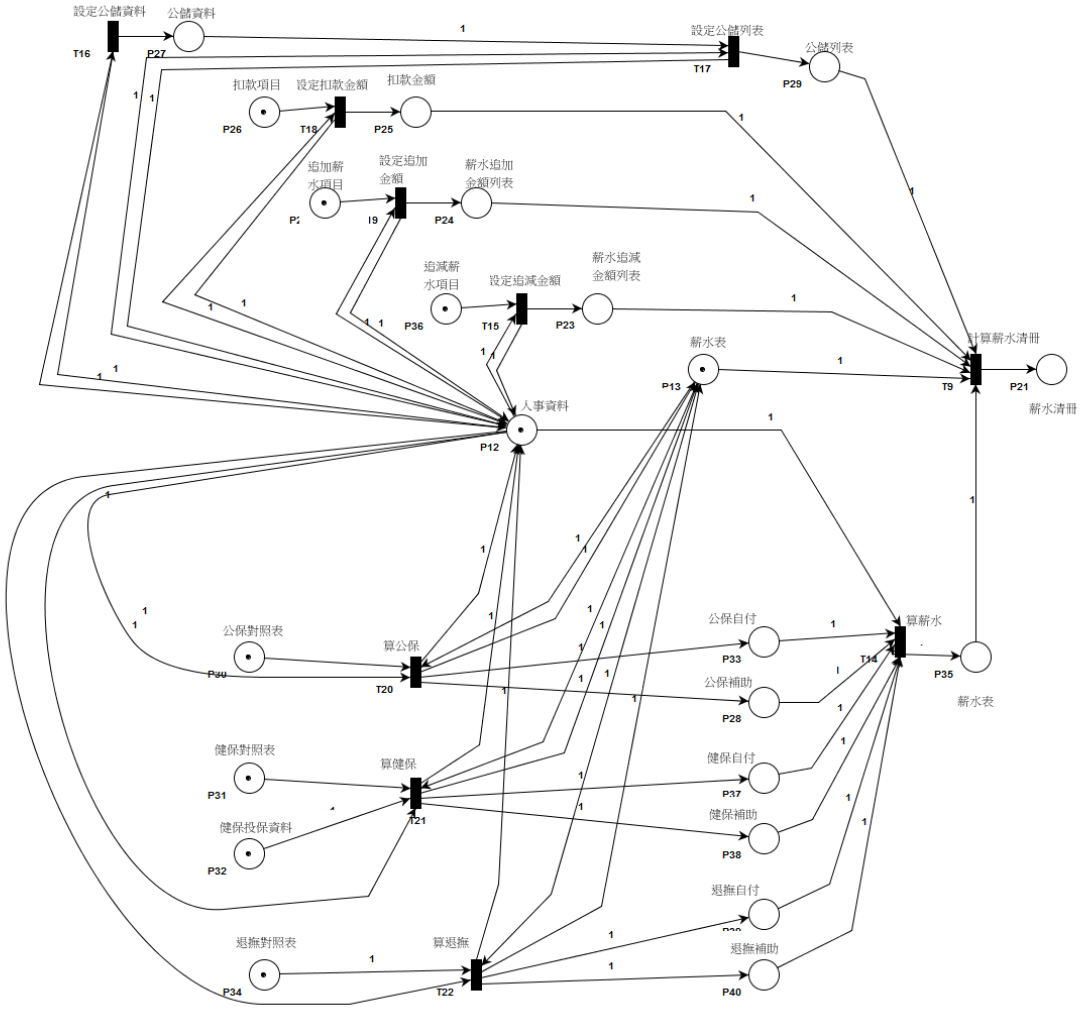


圖 5-2(第二階段)：正職教師的薪水計算作業流程圖(2/2)

- 關於薪水計算的 Petri net 流程圖的驗證請參閱 Appendix A3，Appendix A3 裡頭有詳細的說明。

5.2 規劃資料庫關連圖

使用資料庫關連圖與 Petri net 流程圖互相比對，在撰寫程式時，容易釐清整體運作的關連，Petri net 流程圖可以幫助了解程式運作邏輯，資料庫關連圖顯示程式運作後，資料的流向及存取範圍，Petri net 流程圖與資料庫關連圖互相搭配運用，能夠作為撰寫程式碼的基礎藍圖。

Petri net 作業流程圖中的 place 代表作業流程中所需的資源，這些資源區分為暫時性及永久性資源。

(1) 暫時性資源只供緊接而來的工作使用，工作使用完畢後，即會消失，不影響後續工作進行；暫時性資源暫存在記憶體或是其他暫時性的儲存媒體中。

(2) 永久性資源，具有永久存放的特性，可供各項工作流程使用，使用彈性較大，應用較廣泛，永久性資源統一存放在資料庫當中，利用關連式資料庫的特點，進行規劃，便於程式進行一致性的存取。

代表永久性資源的 place，可被規劃為獨立的資料表，這些資料表，存放工作運作後的結果，或是供其他工作取用，作為工作的資源。基本上，一個資料表代表一項永久性的資源，但實務上有些應用，由於永久性資源可長期存在，因此，一個資料表有時可以代表多個永久性資源，表示提供的資源可供不同的工作使用。

資料表的欄位由工作實際所需的項目決定，例如人事資料這項永久性資源，可作為研究費及導師費計算的資源，而計算研究費需要人事資料中的學歷項目，計算導師費則需要人事資料中的是否兼任導師這個項目，所以，人事資料的資料表就必須包含學歷及是否兼任導師的欄位，至於其他欄位的需求，也是由相同方法推導而得，資料表及資料表的欄位都是由工作所需決定，工作的執行主導資料表及欄位的規劃。

以圖 5-3 為例說明，人事資料此項資源屬於永久性資源，因此轉換為「人事資料表」時，考慮到計算薪水，需要有姓名、身份證字號、學歷、俸額等資料才能進行薪水的計算，因此人事資料表的設計即必須包含有對應的欄位資料(id、name、level、及 degree 等)，以利後續的程式邏輯撰寫。

另一方面，最高本薪與學歷有關，因此需要「最高本薪與學歷對照表」包含最高本薪(level_top)與學歷(degree)二個欄位，以便計算實際本薪。

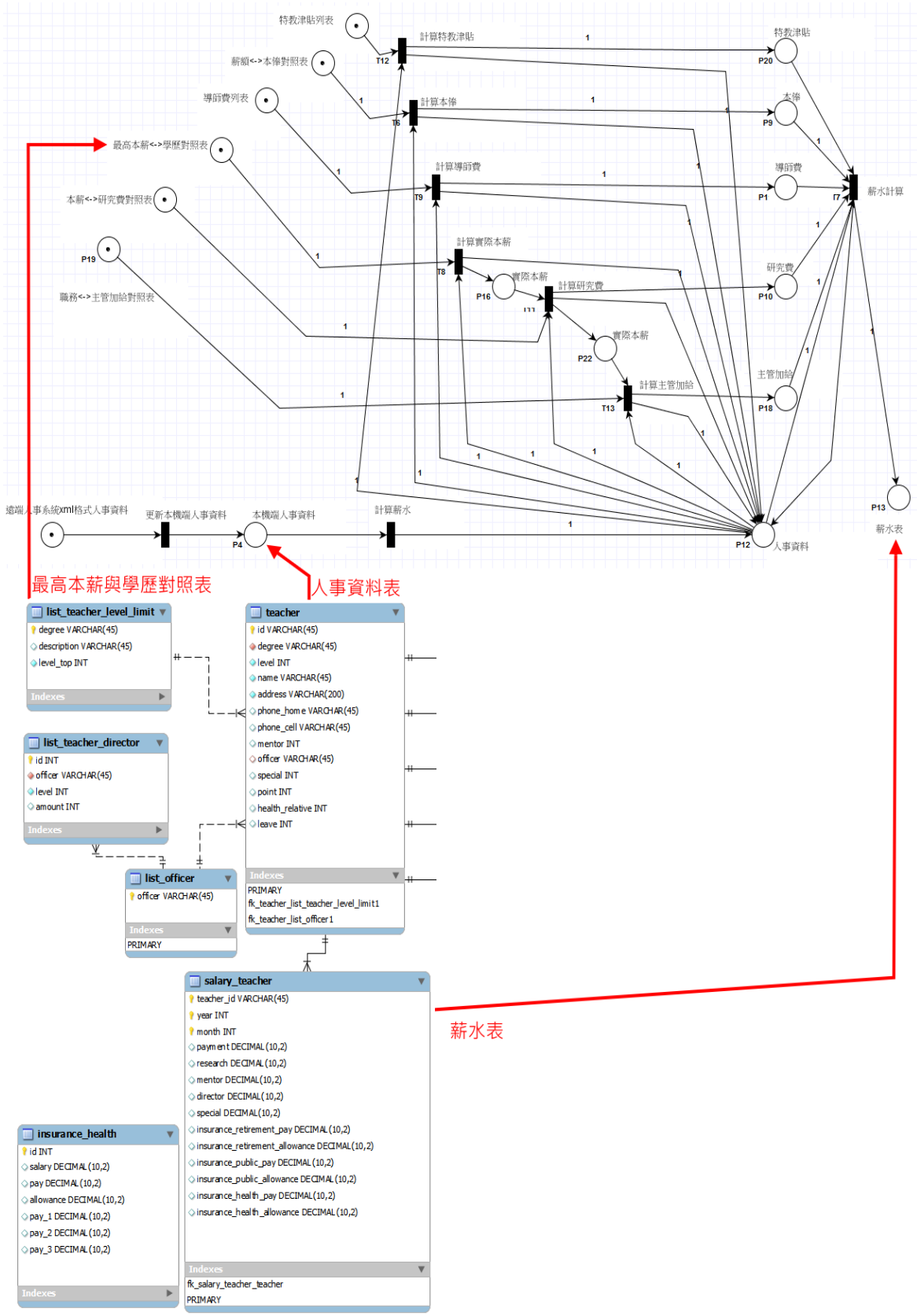


圖 5-3：Petri net 流程圖中的 place 轉換為資料庫關連圖

圖 5-4 是完整的正職教師薪資計算資料庫關連圖，依據圖 5-1（第一階段流程）及圖 5-2（第二階段流程）的 Petri net 流程圖規劃，使用 MySQL Workbench 繪製，繪製完成後，可以直接匯入 MySQL 資料庫，建立系統的資料庫結構。

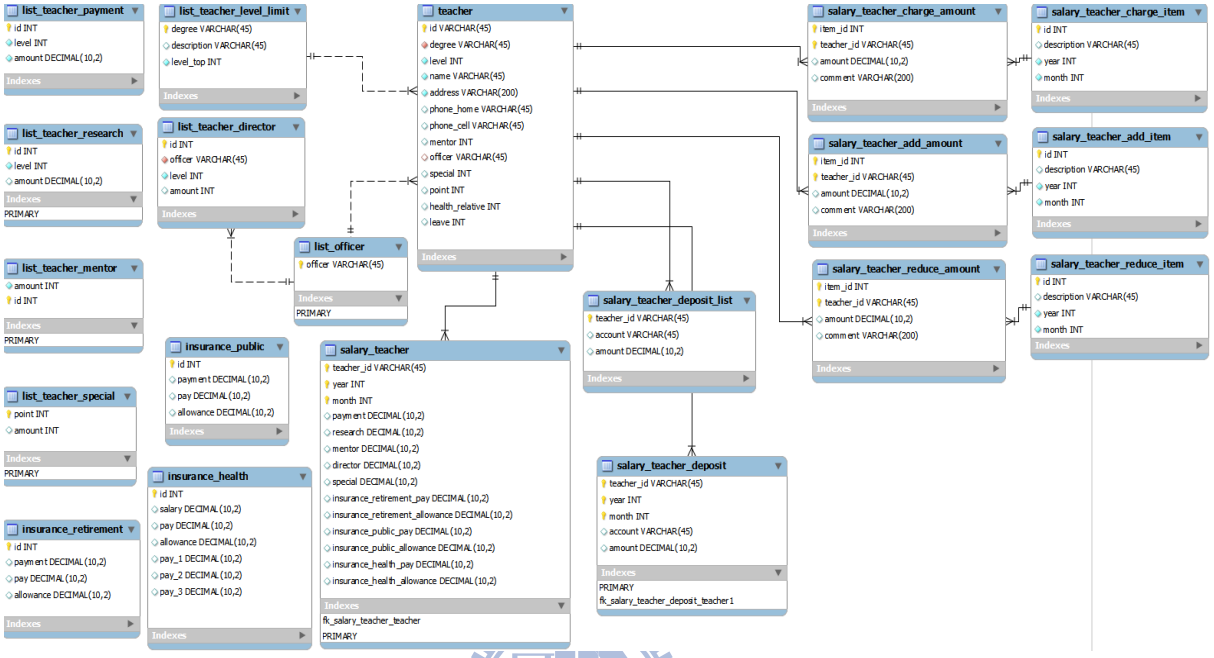


圖 5-4：資料庫關連圖

5.3 規劃及實作 MVC 模組

實務上，通常難以直接看圖實作，將複雜的薪資作業 Petri net 流程圖轉換為計算薪資的程式，且程式碼的分工撰寫也不容易。另一方面，目前在系統設計上常會導入 MVC 模組化的開發概念，以提升系統開發的效率，適當結合這二項工具，我們可以將 Petri net 流程圖轉換為一個個較小的概念上的 MVC 實作單元，能夠減低 MVC 模組內流程的複雜度，較易將 MVC 模組內的工作內容轉換為程式邏輯，有利於自動化程式的實作。

本論文分別以人事角色、出納人員角色、幹事角色的觀點轉換 Petri net 流程圖為概念上的 MVC 模組，概念上的 MVC 模組一一對應到適當的角色，對應的角色負責模組內工作流程的執行。本論文所提出的 PRMRX 建構法將 Petri net 流程圖以 RBAC 中的角色觀點進行轉換分割，轉換成概念上的 MVC 模組，其中的 transition 及 place 數目較整體 Petri net 流程圖少，容易將 transition 代表的工作內容轉變成程式邏輯，降低程式撰寫上的複雜度。

採用這種方法的好處：(1)概念上的 MVC 模組在模組內的工作告一段落

時，相關的資源會存放在資料表內，以供其餘的 MVC 模組存取，MVC 模組間以資料表結合，進行互動合作；(2)MVC 模組的程式邏輯相關性低，一旦 MVC 模組內的邏輯更動，只會更動與模組相關的資料表內容，較不會牽動其餘的 MVC 模組內的程式邏輯，降低 MVC 模組間的程式藕合性；(3)MVC 模組間的藕合性低，彼此間的獨立性高，容易進行分工作業，MVC 模組及資料庫關連圖規劃完成後，即可分派 MVC 模組予不同人員，分頭開發，加快開發進度。

5.3.1 MVC 模組的規劃

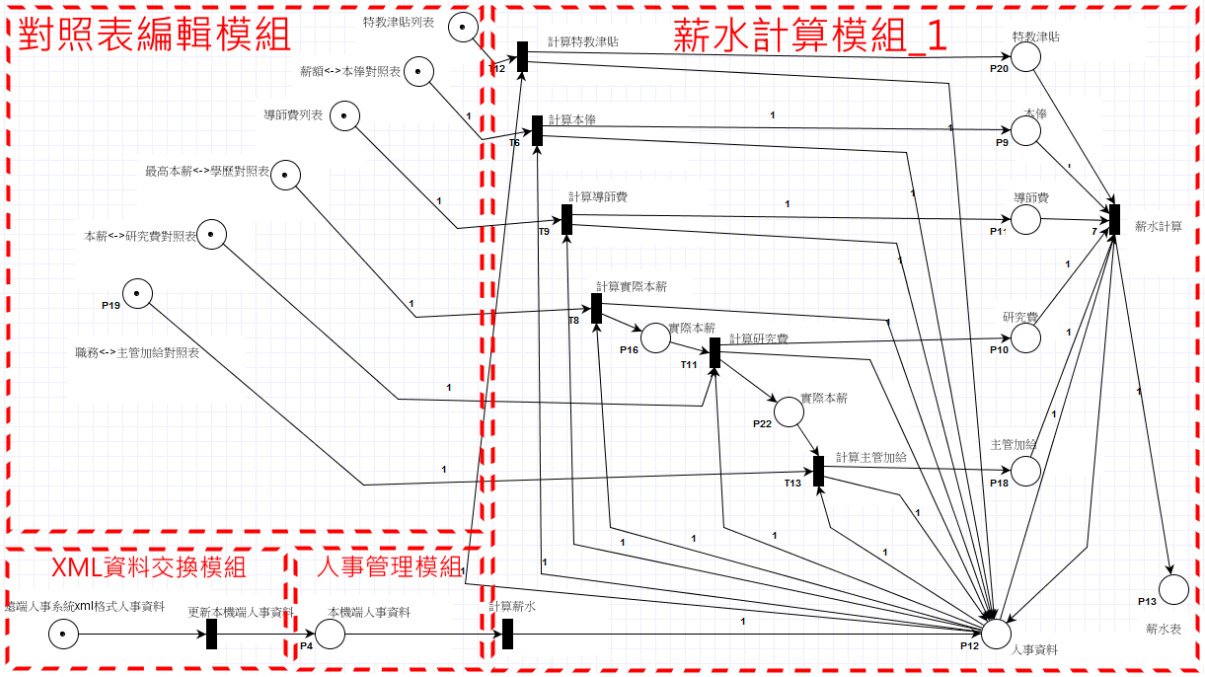


圖 5-5：正職教師的薪資計算 MVC 模組規劃(1/2)

表 5-1：正職及長期代理教師的薪資計算 MVC 模組及職務角色對應表

職務角色	MVC 模組名稱
人事	人事管理模組、退撫模組、XML 資料交換模組
出納	對照表編輯模組 薪水計算模組(薪水計算模組_1 及薪水計算模組_2) 扣款模組、追加薪水模組、追減薪水模組
幹事	公保模組、健保模組

本論文分別以人事角色、出納人員角色、幹事角色的觀點轉換Petri net流程圖為概念上的MVC模組，概念上的MVC模組一一對應到適當的角色，對應的角色負責模組內工作流程的執行。

在規劃正職教師薪資計算的Petri net流程圖的系統實作時，我們可以進一步區分為三個部份(各部份的細項工作內容如表5-1所列)，分別由三種不同的角色觀點加以執行。如表5-1、圖5-5及圖5-6，我們分別將人事角色、出納人員角色、幹事角色所對應的工作觀點轉換為MVC模組後，各角色分別負責不同模組所規範的工作。薪水計算模組_1及薪水計算模組_2為同一個模組(因MVC模組圖形較複雜)，因同一個模組分別在二張Petri net流程圖中，故以數字區分，表示此MVC模組跨越多張Petri net流程圖。

各職務角色及MVC模組的對應如表5-1， MVC模組代表了不同角色執行的工作內容，也是實作的單元，各MVC模組實作完成後，即可組成完整的自動化流程。

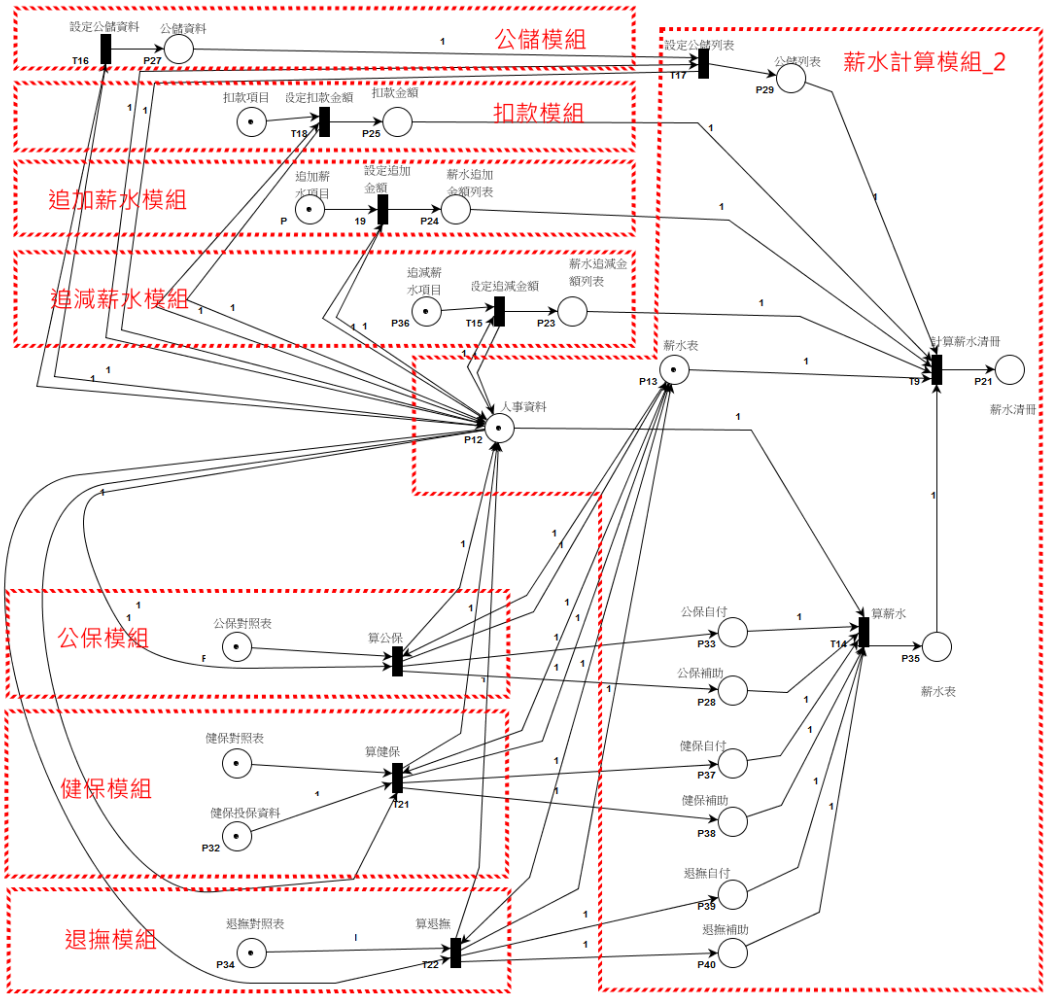


圖5-6：正職教師的薪資計算MVC模組規劃(2/2)

5.3.2 MVC 模組的實作

本論文中各 MVC 模組的實作根據 5.3.1 小節所描述的方式執行，如圖 5-10、5-11、5-12 及 5-13 為有關正職教師薪資計算的範例(包括人事管理模組、優惠存款設定模組、薪水計算模組、對照表編輯模組等的部份畫面)。MVC 模組的實作即根據 Petri net 流程圖及資料庫關連圖的規劃，將工作及存取資源的順序轉化為程式邏輯，完成程式的實作。

本論文以 Zend Framework 的 MVC 框架分別實作 Controller、Model 及 View 的部份，資料庫相關類別則使用 Zend Framework 開發的 Zend_Db 等一系列類別，進行實作。圖 5-7、5-8、5-9 為人事管理模組使用 Zend Framework MVC 框架進行開發的部份程式碼畫面，View 的顯示畫面為圖 5-10。

```php
<?php
class ManagePeople_TeacherController extends Zend_Controller_Action {

    public function init() {
        $this->logic = new App_ManagePeople_Model_Logic();
    }

    public function viewAction() {

    }

    public function teacherViewAction() {
        $this->_helper->layout->disableLayout();
        $this->view->rowset=$this->logic->teacherViewData();
    }

    public function degreeAction(){
        $this->_helper->layout->disableLayout();
        if($this->_request->isPost()){
            $this->view->degree=$this->_request->getPost('degree');
            $this->view->rowset=$this->logic->degree();
        }
    }
```

圖 5-7：人事管理模組的 Controller 類別(只顯示部份程式碼)

```php
<?php
class App_ManagePeople_Model_Data{

    protected $_db;
    protected $_teacher;
    protected $_degree;
    protected $_level;
    protected $_officer;

    public function __construct(){
        $bootstrap=Zend_Controller_Front::getInstance()->getParam('bootstrap');
        $resource=$bootstrap->getPluginResource('db');
        $db=$resource->getDbAdapter();
        $this->_db=$db;
        $db->query('set names utf8');
        $this->_teacher = new App_Model_DbTable_Teacher();
        $this->_degree= new App_Model_DbTable_ListTeacherLevelLimit();
        $this->_level=new App_Model_DbTable_ListTeacherPayment();
        $this->_officer=new App_Model_DbTable_ListOfficer();
    }
```

圖 5-8：人事管理模組的 Model 類別(只顯示部份程式碼)

```php
<link rel="stylesheet"
href="<?php echo $this->baseUrl(); ?>/../js/jquery/jquery-ui/css/ui-lightness/ui.css"
type="text/css" />
<script type="text/javascript"
src="<?php echo $this->baseUrl(); ?>/../js/jquery/jquery-ui/js/ui.js"></script>
<script type="text/javascript">
    $(function(){

        $('form.teacher-modify').hide();
        $('form.teacher-del').hide();

        /**
         * 顯示教師資料
         */
        function showTeacher(){
            $.ajax({
                type:'POST',
                dataType:'html',
                url:'<?php echo $this->url(array('module' => 'manage-people',
                    'controller' => 'teacher', 'action' => 'teacher-view')); ?>',
                success:function(html){
                    $('div.teacher-view').html(html);
                }
            });
```

圖 5-9：人事管理模組的 View 類別(只顯示部份程式碼)

圖 5-10：人事管理模組 View 畫面(身份證字號屬性為 readonly，無法更改)

基本資料				基本薪津		追加薪津	追扣薪津	薪津	補助款			基本扣款				其他扣款	實領
身份證字號	姓名	年/月	俸額	薪津 研究費 主管加給	導師費 特教津貼	*晉級	*導師費扣款	薪津	退撫補助	公保補助	健保補助	退撫自付	公保自付	健保自付	公儲	*曾琳結婚	實領
G123456789	陳依依	100/1	525	40500 30560 0	2000 0	0	0	73060	6318	1882	4479	3402	1014	2258	10000	0	56386
B123456789	汪小小	100/1	450	35330 25570 4990	0 0	700	0	66590	5511	1642	4110	2968	884	1036	0	500	61202
F123456789	白測測	100/1	410	33390 25570 0	2000 0	800	-48	61712	5209	1552	3741	2805	835	943	0	3600	53529
E123456789	吳九九	100/1	350	30485 25570 4990	0 0	0	0	61045	4755	1417	3925	2561	763	3960	0	2400	51361
D123456789	林琳	100/1	330	29515 22520 0	0 1800	0	0	53835	4605	1371	3408	2479	739	859	0	1200	48558
A123456789	羅一一	100/1	310	28545 22520 0	2000 0	600	-36	53629	4453	1327	3261	2398	714	822	5000	600	44095

圖 5-11：薪水計算模組 View 畫面

圖 5-12：對照表編輯模組 View 畫面

圖 5-13：優惠存款設定模組的 View 畫面

5.4 RBAC 管理模組的實作

PRMRX 建構法將 Petri net 流程圖以角色觀點，依各流程中牽涉的角色不同，將流程轉換為不同的 MVC 模組，由不同的角色負責不同的 MVC 模組的工作，減化程式實作的規模，縮減實作中 MVC 模組當中相關的資料庫表格數量，降低實作的複雜度；並以使用者、角色及 MVC 模組的觀點實作獨立於薪水流程的 RBAC 權限管理模組[1][23]，控管流程中不同角色的執行權限。

例如薪水製作流程，以工作執行及資源配置的觀點進行 Petri net 流程圖的繪製，而 RBAC 權限控管模組則是以 MVC 模組、角色及使用者三者的關係為觀點，進行權限的控管，由於兩者規劃的觀點不同，因此，將 PRMRX 建構法的 RBAC 權限控管模組獨立於薪水製作流程的各 MVC 模組之外，另外繪製 Petri net 圖形，撰寫獨立於薪水製作流程的權限控管 MVC 模組。

MVC 權限控管模組的主要目的是以角色作為權限控管的樞紐，以管理使用者、角色及 MVC 模組三者之間的關係；根據 RBAC 的觀點設定使用者、角色及 MVC 模組及三者間的關係，適當地將 MVC 模組的執行權限分配給不同角色，再將使用者設定為不同的角色，讓使用者透過設定為適當的角色而能執行不同的 MVC 模組，以便簡化使用者權限設定上的複雜度。

如圖 5-14 所示，使用 Petri net 繪製 RBAC 權限管理模組中使用者、角色及 MVC 程式模組三者間的關係，進行權限設定時，即以角色為中心，分別設定角色與 MVC 模組的關係及角色與使用者的關係。

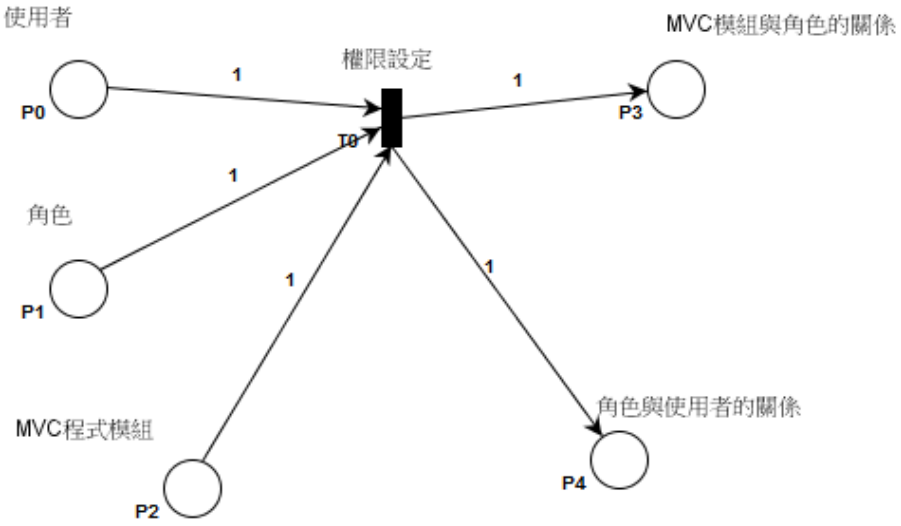


圖 5-14：RBAC 權限控管模組 Petri net 流程圖

圖 5-15 的資料庫關連圖中，rbac_module 資料表代表圖 5-14 的 MVC 程式模組，rbac_role 資料表代表圖 5-14 的角色，rbac_user 代表圖 5-14 的使用者，至於 rbac_module_role 代表 MVC 模組與角色的關係，rbac_role_user 則代表角色與使用者的關係。

底下表 5-2、5-3、5-4、5-5 及 5-6 各資料表分別代表 MVC 程式模組資

料表、角色資料表、使用者資料表、MVC 程式模組與角色關係資料表及角色與使用者關係資料表。

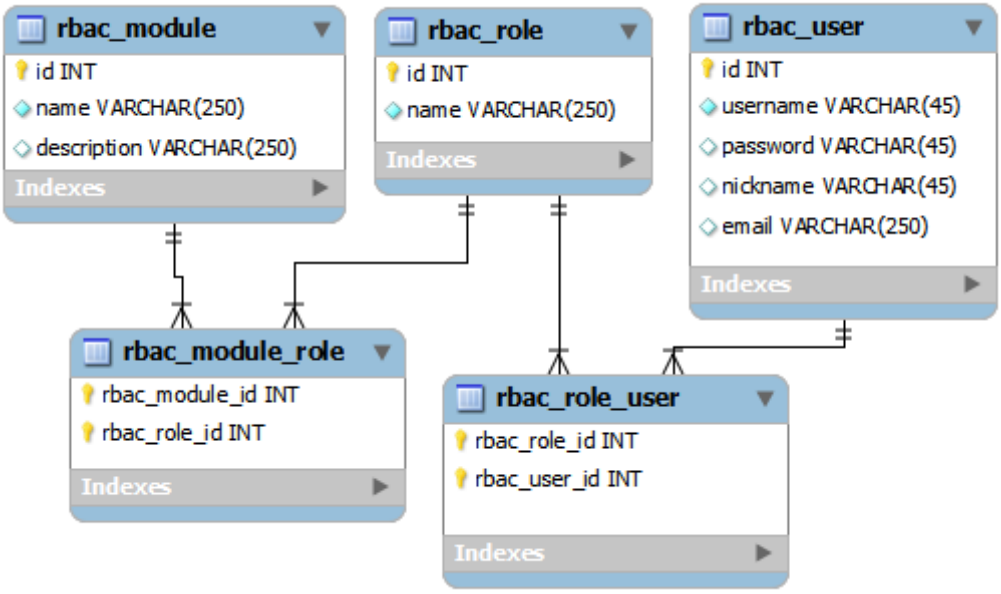


圖 5-15：RBAC 權限控管模組資料庫關連圖

表 5-2：rbac_module(MVC 程式模組資料表)

欄位名稱	欄位意義
id	流水號（primary key）
name	模組名稱
description	模組描述

表 5-3：rbac_role（角色資料表）

欄位名稱	欄位意義
id	流水號（primary key）
name	角色名稱

表 5-4：rbac_user(使用者資料表)

欄位名稱	欄位意義
id	流水號（primary key）
username	使用者帳號
password	使用者密碼
nickname	使用者暱稱
email	使用者電子郵件

表 5-5：rbac_module_role(MVC 程式模組與角色關係資料表)

欄位名稱	欄位意義
rbac_module_id	rbac_module 資料表的流水號
rbac_role_id	rbac_role 資料表的流水號

表 5-6：rbac_role_user(角色與使用者關係資料表)

欄位名稱	欄位意義
rbac_role_id	rbac_role 資料表的流水號
rbac_user_id	rbac_user 資料表的流水號

RBAC 權限控管模組以角色、使用者及 MVC 程式模組的觀點繪製 Petri net 圖形，並依據 Petri net 圖形繪製資料庫關連圖後，接下來就是 RBAC 權限控管模組的實作，整個實作依照 Petri net 圖形及資料庫關連圖的規劃，撰寫程式邏輯並設計畫面，完成後的畫面如圖 5-16 及 5-17 所示。

圖 5-16：MVC 程式模組設定畫面

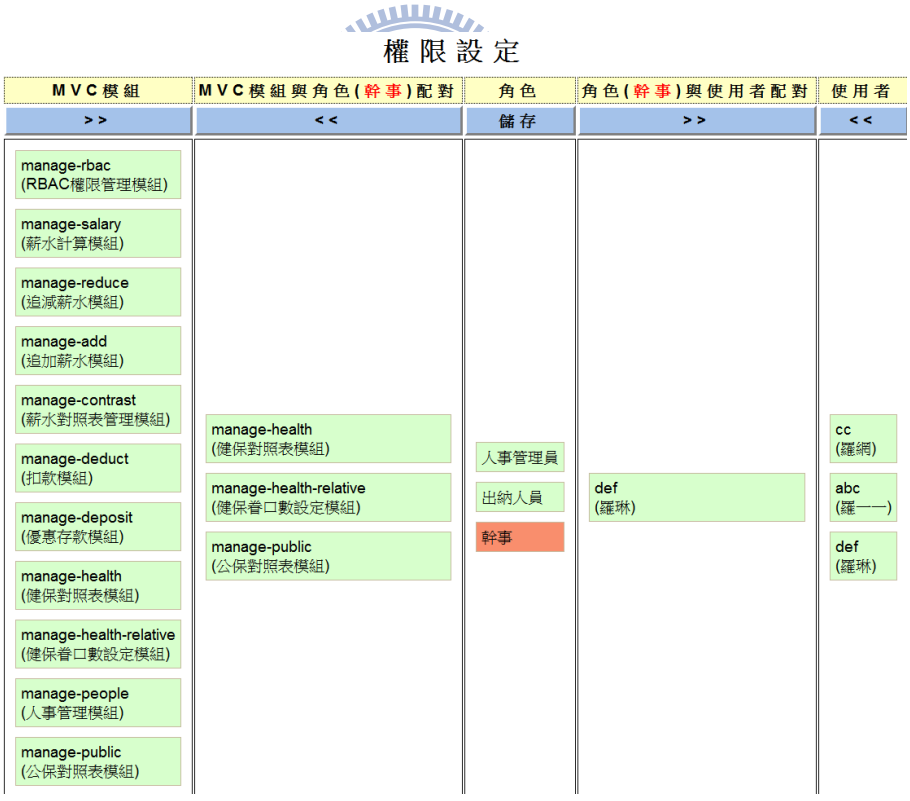


圖 5-17：權限設定畫面

5.5 XML 資料交換中心的實作

PRMRX 建構法中的 X，代表 XML 資料交換中心，其概念如圖 5-18，XML 資料交換中心利用 XML 具有與軟硬體無關的中立性，不論何種平台，均能讀取或產生 XML 資料，PRMRX 建構法即基於 XML 的此種特性，建立以 XML 作為資料交換格式的 XML 資料交換中心，不同作業系統環境、不同程式語言下開發的應用程式，透過 XML 資料交換中心，以網路更新作業流程中所需的資料，藉以整合各類異質系統。

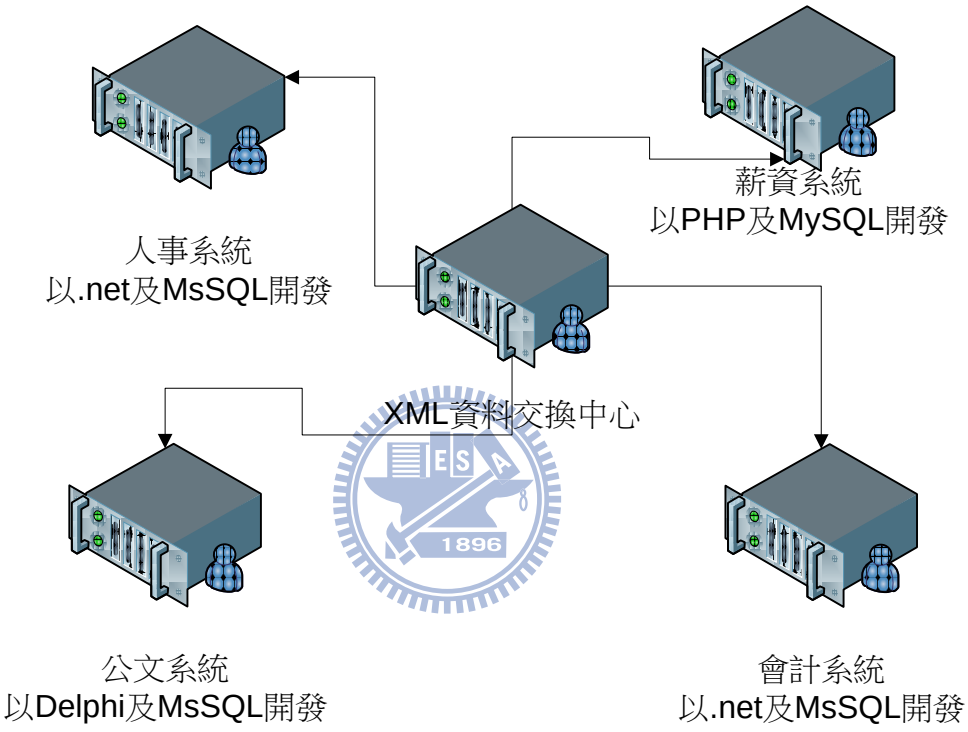


圖 5-18：XML 資料交換中心概念圖

本論文以 XML 資料交換中心作為人事資料的交換中心，薪資系統若要更新人事資料，直接連結 XML 資料交換中心即可更新人事資料，如果有多所學校的薪資系統需要更新人事資料，不需要一一更新這些學校的薪資系統的人事資料，只需更新 XML 資料交換中心的人事資料，學校端的薪資系統連結 XML 資料交換中心即可更新，如果學校使用不同於本論文使用的程式語言及資料庫軟體開發薪資系統，只要根據 XML 格式，新增資料交換的功能，也能從 XML 資料交換中心更新人事資料。

實作演算法如下：

(1)訂定交換資料種類及 XML 資料交換格式。

(2)確認程式邏輯。

(3)實作薪資系統的 XML 資料交換模組。

(4)實作 XML 資料交換中心的 XML 資料交換模組。

5.5.1 訂定交換資料種類及 XML 資料交換格式

本論文中,薪資系統與 XML 資料交換中心交換的資料為教師人事資料,因此,必須訂定教師人事資料的標準 XML 資料交換格式,以便讓 XML 資料交換中心,將資料庫中的人事資料轉換為標準 XML 格式,有此標準 XML 格式,才能讓薪資系統的 XML 資料交換模組,根據預先定義的標準格式讀取方式,讀取標準 XML 格式,更新薪資系統的人事資料。

薪資系統與 XML 資料交換中心交換人事資料的 XML 標準格式如下:

```
<?xml version="1.0" encoding="utf-8" ?>
  <people>
   <teacher>
      <id>K123456789</id>
      <degree>大學</degree>
       <level>625</level>
       <name>羅仁治</name>
       <address />
       <phone_home />
       <phone_cell />
       <mentor>0</mentor>
       <officer />
       <special>0</special>
       <point>0</point>
      </teacher>
    </people>
```

5.5.2 確認程式邏輯

以程式邏輯的角度切入，本論文的 XML 資料交換中心與學校端的薪資系統交換人事資料的流程如圖 5-19：

(1)學校端的薪資系統以 POST 方式，送出認證的帳號及密碼到 XML 資料交換中心

(2)XML 資料交換中心確認薪資系統的帳號、密碼及 ip 均正確無誤。

(3)抓取該校的人事資料。

(4)將該校的人事資料轉換為 XML 格式。

(5)學校端的薪資系統更新本機端的人事資料

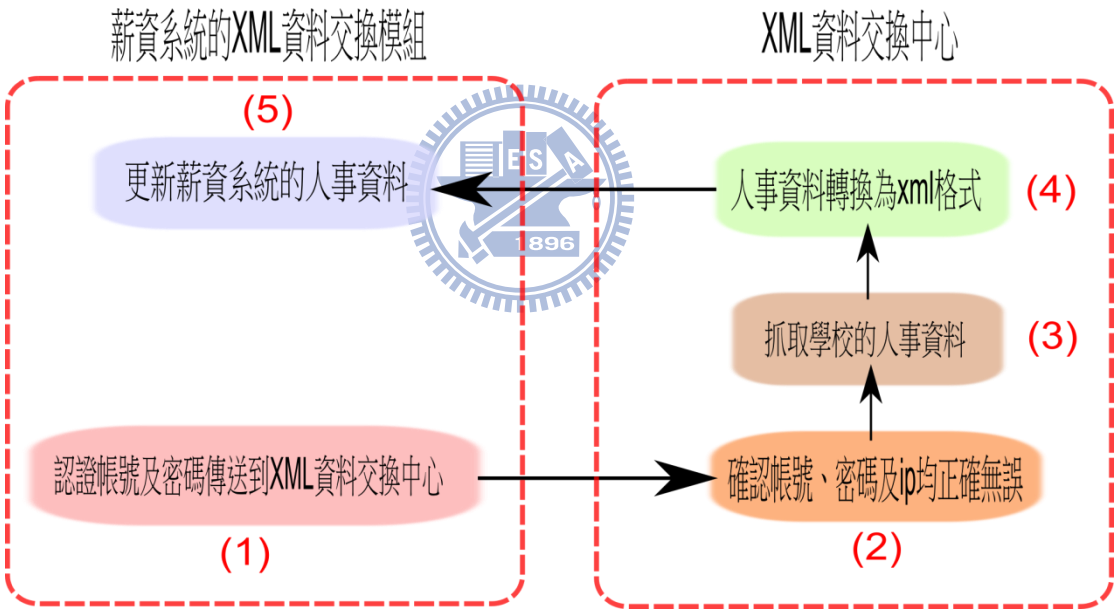


圖 5-19：XML 資料交換中心的程式邏輯

本論文以圖 5-20 的虛擬環境進行實作，XML 資料交換中心(140.113.2.226)儲存學校薪資計算所需的人事資料，璞易小學及琳玉小學的相關人事資料均置放於 XML 資料交換中心，璞易小學及琳玉小學則各有相同的薪資系統。以璞易小學為例，一旦薪資系統有更新人事資料的需求，璞易小學具有操作權限的人員，設定 XML 資料交換所需的帳號、密碼及網址，由程式將帳號及密碼，送到 XML 資料交換中心，XML 資料交換中心確認帳號、密碼無誤後，且再確認遠端的 ip 為 140.113.2.221，則將璞易小學

的相關資料以 XML 的格式傳送到璞易小學薪資系統，璞易小學的薪資系統即會自動更新人事資料。

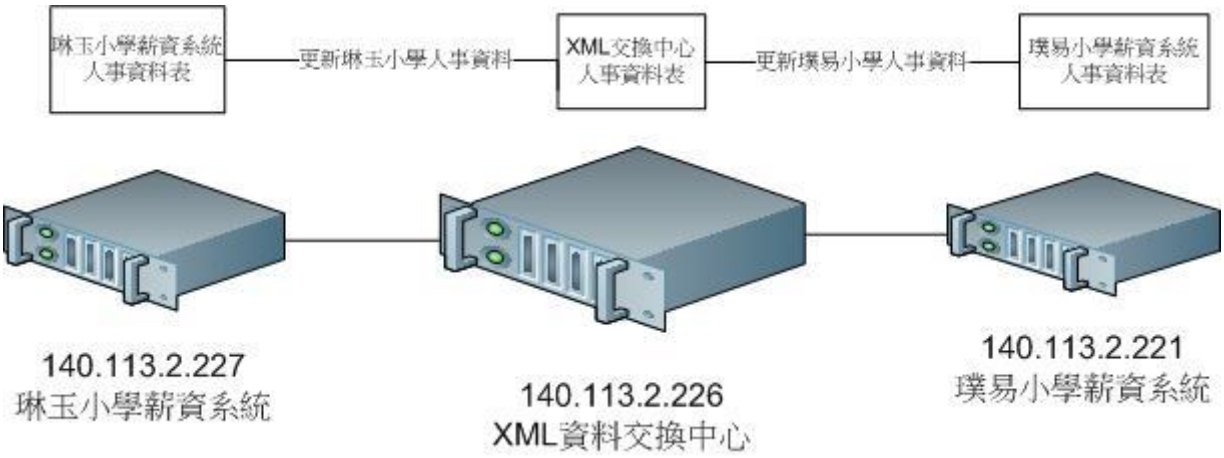


圖 5-20：虛擬的 XML 資料交換中心實作的實體配置圖

5.5.3 實作薪資系統的 XML 資料交換模組

圖 5-21 為薪資系統的 XML 資料交換模組的實作，XML 資料交換模組為一獨立的 MVC 模組。薪資系統的 XML 資料交換模組，功能為設定連結 XML 資料交換中心的帳號、密碼及網址，設定後，提供自動化更新本機端人事資料的功能。

圖 5-21：薪資系統的 XML 資料交換模組初始設定圖

5.5.4 實作 XML 資料交換中心的 XML 資料交換模組

圖 5-22 為 XML 資料交換中心，因為安全性理由，需設定連線學校的帳號、密碼及連線 ip，限制存取的 ip，並以密號及密碼認證，避免不當的存取，外洩人事資料。進行初始設定後，即可允許薪資系統以自動化的方式更新人事資料。

圖 5-22：XML 資料交換中心初始設定圖

第六章　問卷調查與使用經驗分析

本論文設計的薪資系統，目前擬提供的主要使用對象是以行政人員(一般職員如出納、幹事等)或兼具行政職的教師為主。為了實際了解本論文實作中小學薪資系統的可用性等各個面向，我們特別邀請將近十位中小學老師及職員登入後操作測試，並蒐集相關意見，以作為後續改進的參考依據。

本問卷調查的實施方式如下:

(a) 目前實作完成的系統，SSL 採用自行建立的臨時憑證系統，其連線網址為 https://140.113.2.226。(login 畫面如圖 4-4 所示)

(b) 運用 Google 線上問卷調查系統(如圖 6-1 所示)，蒐集相關意見，其網址如下：
https://spreadsheets.google.com/spreadsheet/viewform?formkey=dEdQTGU5NzBEWTdWTUhEbzdpN0k1V0E6MQ。

(c) 我們提供測試帳號(不同權限)及密碼，邀請近十位中小學老師及職員登入後操作測試，並請試用後的人員，以不記名的方式，填寫網路問卷系統，作為後續系統設計及改進的參考依據。

備註: 為了預防 robot 程式攻擊，本系統的 login 畫面，有特別增加數字驗證機制

圖 6-1: Google 問卷調查

本項問卷，共分成七個項目，其中的第一至第六項目的給分為一分到五分(最高)，項目七則由使用者自由填寫測試後的意見。目前所蒐集的使用者的回饋如表 6-1。

從問卷的初步彙整結果可以看出,使用者基本上都持正面肯定的態度。另一方面，許多使用者期待此套系統能夠持續改進操作介面，強化使用功能。最後，由於目前測試的對象，屬於邀請性質為主，而且參與測試的人數還不是很多，後續將擴大測試對象與範圍（本研究將來持續努力的目標與方向)。

表 6-1：網路問卷填寫結果

項目	給分 (匿名 1)	給分 (匿名 2)	給分 (匿名 3)	給分 (匿名 4)	給分 (匿名 5)	給分 (匿名 6)	給分 (匿名 7)	給分 (匿名 8)
1.可靠程度 (是否常當機)	5	5	未填	5	5	5	5	5
2.資訊即時更新程度 (修改內容後是否立刻顯示更新結果)	5	5	5	5	4	5	5	5
3.使用的容易度	5	5	4	5	5	5	4	4
4.操作介面的一致性 (每個頁面的操作方式是否類似)	5	5	5	5	5	4	5	5
5.執行速度	5	5	5	5	5	5	5	5
6.正確性	5	5	5	5	5	5	5	5
7.其他意見	(1)身份證號可加入規則判斷，避免輸入錯誤 。 (2)能點選的地方是否能仍顯示底線, 或改為按鈕, 較易知道該處可進行操作。 (3)可以考慮增加扣稅界面。							

第七章　結論與未來研究方向

本論文主要的貢獻為提出 PRMRX 建構法並應用此法設計暨實作中小學薪資系統。PRMRX 建構法結合了 Petri net 流程規劃、RDBMS 規劃、MVC 設計模式、RBAC 權限控管及 XML 資料交換機制。本論文的實作成果，也驗證了 PRMRX 建構法是一套能付諸實現的有效方法。

尤其，更重要的是「PRMRX 建構法」不但適用於薪資系統的開發，只要經過適當的調整及修正，這套方法也可適用於其他常見的應用平台(例如人事系統、採購系統等)，採用不同種類程式語言的系統開發專案(不論是架構在 PHP 及 MySQL 上的系統，或是採用.net 及 MsSQL 的平台，或是以 JAVA 及 PostgreSQL 進行建構的系統專案)，均可一體適用。

7.1 結論

本論文提出一套 PRMRX 建構法以利實作網路系統，透過緊密結合 Petri net 流程圖(P)、關連式資料庫(R)、MVC 程式模組(M)、RBAC 權限控管(R)及 XML(X)資料交換機制的概念，提供明確的步驟，轉換各種概念及觀點，使規劃及實作中，原本彼此關係鬆散的各個階段彼此環環相扣，互有關連，避免各自為政的狀況發生。

PRMRX 建構法整合各自獨立作業的觀點，提供一套各類觀點間轉換的框架，協助程式的開發，其轉換步驟摘要如下。：

(1)P(Petri net)：以工作及資源觀點繪製 Petri net 作業流程圖。

(2)R(RDBMS)：將 Petri net 流程圖中的 place 轉換為 RDBMS。

(3)M(MVC)：以角色觀點切割 Petri net 流程圖為一塊塊 MVC 程式模組，並加以實作。

(4)R(RBAC)：MVC 模組名稱轉換為權限名稱，實作 RBAC 權限控管模組。

(5)X(XML)：實作 XML 資料交換模組及 XML 資料交換中心。

我們以「中小學薪資系統」的規劃及建置為例，按照本論文提出的 PRMRX 建構法，製作出 Petri net 流程圖、概念上的 MVC 程式模組等，驗證此法在規劃設計階段確實可行。其次，本論文並進一步實作「中小學薪資系統」的各個 MVC 程式模組，證明 PRMRX 建構法可以克服規劃及實作上的落差，將作業流程轉換為上線的自動化程式。

本論文所提出的 PRMRX 建構法可以於規劃面、實作面及系統整合幾個層面分別加以討論：

(1)就規劃層面而言具有下列優點：

將複雜的作業流程，轉換為精確定義的 Petri net 流程圖，再接著轉換為資料庫關連圖、程式設計藍圖，大幅降低系統規劃的困難度。

(2)就實作面而言具有下列優點：

MVC 程式模組可以快速反映程式邏輯及畫面的變動，減輕維護的負擔。其次，採用 RBAC 權限控管機制，增加模組開發的彈性及效率。而 XML 資料交換模組可與異質系統交換資料，提升資料交換的方便性。

(3)就整合面而言具有下列優點：

PRMRX 建構法串接不同的概念及觀點，提供明確的轉換步驟，將流程規劃、資料庫實作、權限控管、異質系統的資料交換等系統開發上常見的問題，順利轉換，完成系統實作。

本論文提供一套通用於各式平台及開發語言的標準化方法──PRMRX 建構法，包含開發規劃到實作階段中流程設計、程式設計藍圖、資料庫規劃、系統權限控管機制及異質系統資料交換的需求，將作業流程一步步轉化為實際可上線運作的程式。再者，有相同需求的專案，套用此法，可以避免設計及實作上各自規劃，造成多頭馬車的狀況發生。

7.2 未來研究方向

本論文進行期間所實作的中小學薪資系統，目前僅完成與正職教師相關的基本功能，仍有諸多不盡理想之處，未來將朝下列幾點持續改進。

(1)本論文以 PRMRX 建構法設計中小學薪資系統，目前此套薪資系統主要完成的部份包含了正職教師的薪資計算功能，其餘包括代課費計算、職員薪資計算、工友薪資計算、年終獎金及考核獎金的計算功能仍有待後續努力。

(2)本研究目前僅以薪資系統的建置，驗證 PRMRX 建構法，未來如果能以此法建置更多不同種類的專案，發現可能遭遇的潛在問題並加以持續改進，將更加充份確立 PRMRX 建構法的完整性及適用性。

(3)PRMRX 建構法中的功能需求，封裝於 MVC 模組的實作中，本論文目前以變更 MVC 程式模組的程式邏輯快速反映需求的變更，未來可以進一步以 OOD 的方式設計 MVC 當中的 Model 部份，整合功能面的設計需求，使 PRMRX 建構法更趨完整。

(4)本論文建置的薪資系統權限控管模組，較為單純，對於中小學的規模而言，已經足夠，如果單位的規模更加複雜龐大，可套用 RBAC 中的繼承機制等更複雜的方法，以適用於狀況較為複雜的專案。

(5)目前本論文僅實作人事資料的 XML 資料交換模組，未來如果有其他資料的交換需求，能夠再增加其他類別的 XML 資料交換模組，強化系統的功能。

參考文獻

[1] 林裕峰，「跨網域之校務單一登入系統」，國立交通大學，碩士論文，2007。

[2] 李孟寰，「以情節為基礎之需求擷取與需求文件客製化」，東海大學，論士論文，2006。

[3] 邱郁惠，學會 UML/OOAD 這樣開始就對了，二版，台北，碁峯資訊股份有限公司，2010 年。

[4] 詹雪梅，「薪資管理系統的再生與繼往開來」，台灣，TANET 2006 台灣網際網路研討會，2006。

[5] 標雯琪，「以統一塑模語言 2.0 活動圖為基礎的分析模型」，東海大學，論士論文，2006。

[6] 簡培修，「利用延伸 Petri nets 建構校園文書工作流程模型之初探」，台灣，2007 台灣網際網路研討會，2007。

[7] 蘇冠緯，「以 MVC 架構為基礎之開發者入口網站設計與建置」，中原大學，碩士論文，2002。

[8] 羅仁治，陳昌盛，「以 Petri net 及 MVC 建構中小學薪資作業系統雛型」，台南，2010 台灣網際網路研討會，2010。

[9] Apache. Available from: http://www.apache.org/.

[10] CITRIX. Available from: http://www.citrix.com./lang/English/home.asp.

[11] IptablesHowTo. Available from: https://help.ubuntu.com/community/IptablesHowTo.

[12] jQuery. Available from: http://jquery.com.

[13] MVC Available from: http://www.techrepublic.com/article/mvc-design-pattern-brings-about-better-organization-and-code-reuse/1049862

[14] MySQL. Available from: http://dev.mysql.com/.

[15] Ni, Q., et al., “Privacy-Aware Role-Based Access Control.”, Acm Transactions on Information and System Security, **13**(3), 2010.

[16] OWASP(Open Web Application Security Project). Available from: https://www.owasp.org/index.php/Main_Page.

[17] OOAD(Object-oriented analysis and design). Available from: http://en.wikipedia.org/wiki/Object-oriented_analysis_and_design.

[18] PHP. Available from: http://www.php.net.

[19] PAROS(for web application security assessment). Available from: http://www.parosproxy.org/.
[20] PIPE(Platform Independent Petri net Editor). Available from: http://pipe2.sourceforge.net/.
[21] Petri net. Available from: http://en.wikipedia.org/wiki/Petri_net.
[22] RUP(Rational Unified Process). Available from: http://en.wikipedia.org/wiki/IBM_Rational_Unified_Process.
[23] Sandhu, R.S., et al., "Role based access control models.", Computer, **29**(2): p. 38-&, 1996.
[24] Ubuntu. Available from: http://www.ubuntu.com.
[25] van der Aalst, W.M.P. and A. Kumar, "XML-based schema definition for support of interorganizational workflow.", Information Systems Research, **14**(1): p. 23-46, 2003.
[26] XML. Available from: http://www.w3schools.com/.
[27] Zend Framework. Available from: http:///framework.zend.com.

Appendix A1

Petri net 圖形中，所有指向該 transition 的 place，稱為該 transition 的 input place，由該 transition 往外指的 place，則稱為該 transition 的 output place。例如圖 A1-1 中的 P0、P1 及 P2 就是 transition T0 的 input place， P3 及 P4 則為 transition T0 的 output place。

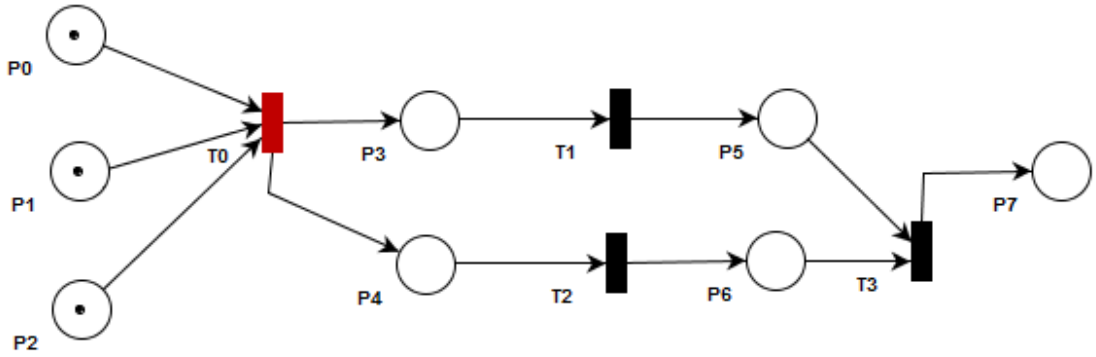


圖 A1-1：Petri net 的狀態說明圖(1)

Petri net 以 token 的分佈狀況表示各種不同的狀態，圖 A1-1 到圖 A1-5 解釋 token 的移轉方式，紅色的方塊，代表 enabled 的 transition，可以被 firing。

圖 A1-1 中當 T0 所有的 input place 都至少有一個 token 時，T0 就可以被 firing(這時 T0 就被稱為 enabled)，T0 被 fired 時，所有 T0 的 input place，包括 P0、P1 及 P2 會消耗掉一個 token，所有 T0 的 output place，包含 P3 及 P4 會增加一個 token，而轉變成圖 A1-2 的狀態。

圖 A1-2 中的 T1 為 enabled 的狀態，被 fired 後，P3 的 token 消失，P5 增加一個 token，轉換為圖 A1-3 的狀態。圖 A1-3 中的 T2 為 enabled 的狀態，被 fired 後，P4 的 token 消失，P6 增加一個 token，轉變為圖 A1-4 的狀態。

圖 A1-4 中的 T3 為 enabled 的狀態，被 fired 之後，T3 的 input places P5 及 P6 各消失一個 token，T3 的 output place P7 增加一個 token，轉變為圖 A1-5 的狀態。

圖 A1-1 到圖 A1-5 中，token 分佈的五種不同狀態，就代表流程的五種狀態，每一種狀態的轉換，都是藉由 transition 的 firing，移轉 token，而轉移到不同的狀態，因而能夠由 token 的分佈，了解流程的執行狀態。

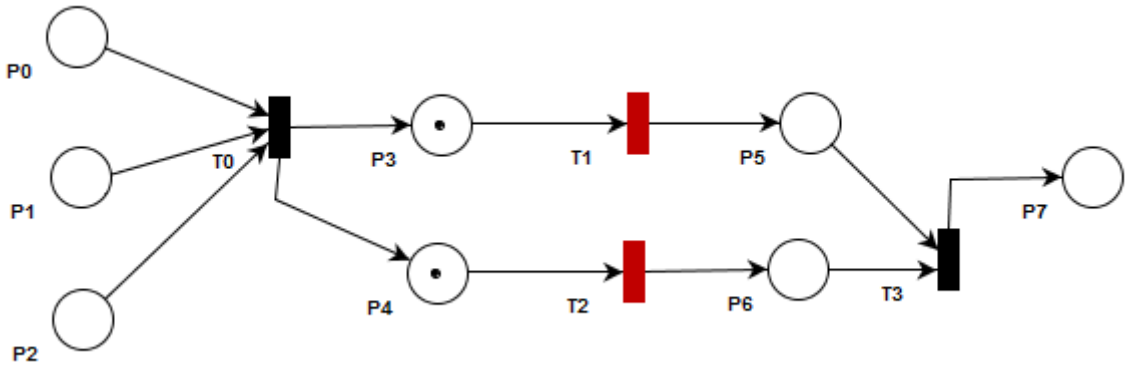


圖 A1-2：Petri net 的狀態說明圖(2)

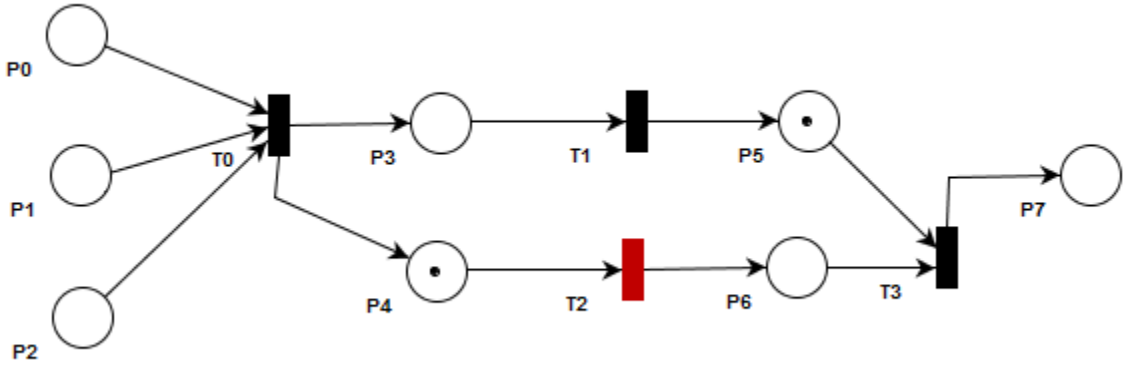


圖 A1-3：Petri net 的狀態說明圖(3)

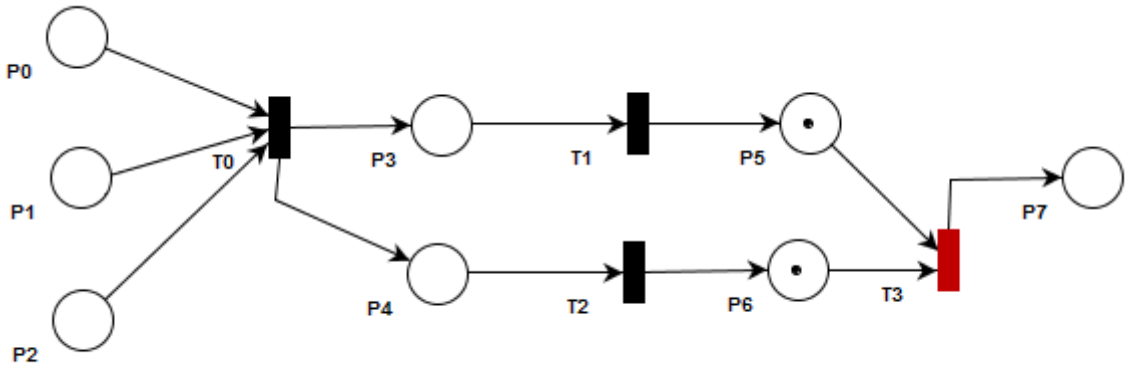


圖 A1-4：Petri net 的狀態說明圖(4)

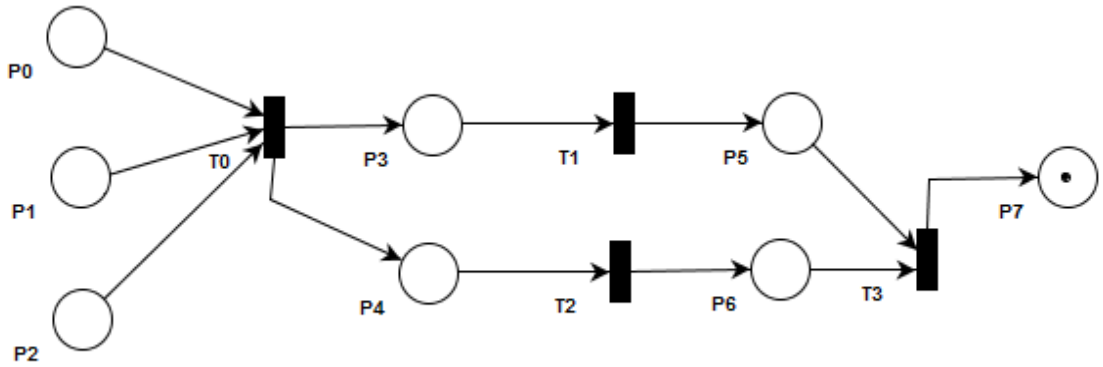


圖 A1-5：Petri net 的狀態說明圖(5)

Appendix A2

Zend Framework 的 Controller 類別依據功能不同，再細分成 Zend_Controller_Front 、 Zend_Controller_Request 、 Zend_Controller_Response、Zend_Controller_Dispatcher 等不同的類別，整個分工方式，如圖 A2-1，不同的流程由不同的類別負責處理，再由 Zend_Controller_Front 類別統合所有的類別，處理類別間的互動，完成使用者的要求。

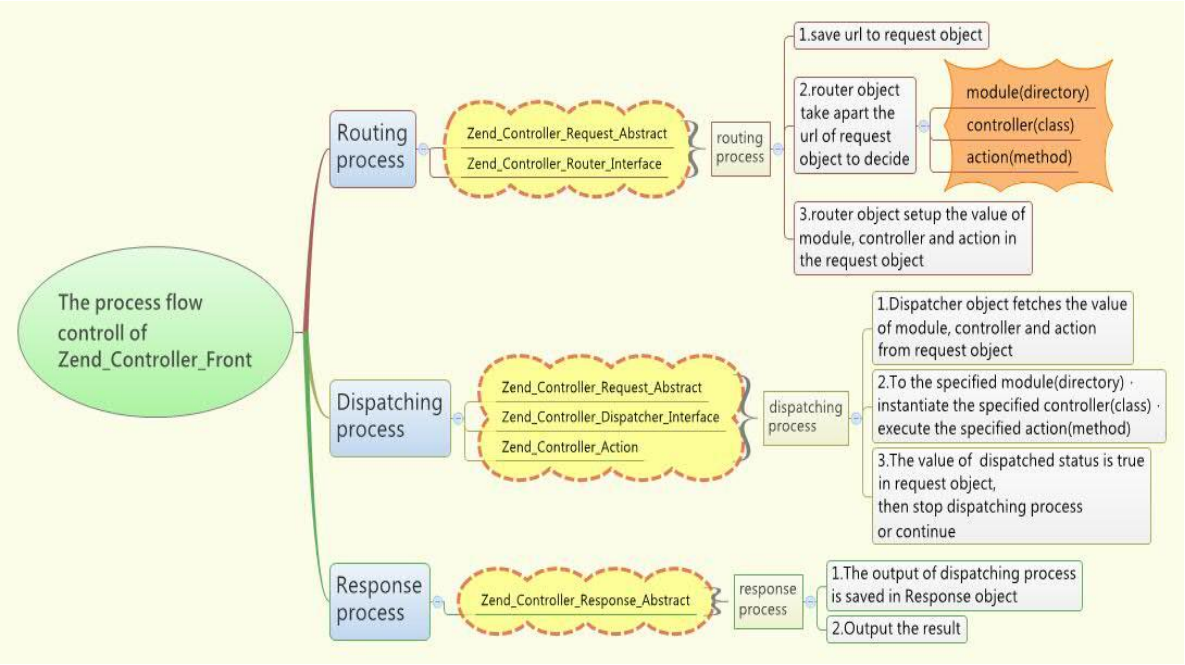


圖 A2-1：Zend Framework Controller 類別分工圖

使用者的瀏覽歷程被區分為 routing process、dispatching process 及 response process 三個歷程，routing process 歷程由 Zend_Controller_Router 類別主控，將使用者的網址轉化成 module、controller、action 三個參數及使用者設定的其他參數資料，再將這些參數資料及使用者的網址存入 Zend_Controller_Request 類別裡頭。

Routing process 結束後，緊接著進入 dispatching process，不同於 routing process 只能經歷一次，dispatching process 可以不斷循環，執行多次。dispatching process 由 Zend_Controller_Dispatcher 類別負責控制 dispatching 的歷程，Zend_Controller_Dispatcher 類別從 Zend_Controller_Request 類別中取出 module、controller 及 action 的參數資料，由 module 參數決定 Zend_Controller_Action 類別所在的目錄位置，由 controller 參數判斷 Zend_Controller_Action 類別的名稱，由

action 參數決定執行 Zend_Controller_Action 類別中的哪個方法，再整合 Model 類別及 View 類別，最後，將結果存入 Zend_Controller_Response 類別當中。

Dispatching process 歷程可以執行多次，Zend_Controller_Dispatch 類別，依據 Zend_Controller_Request 類別中的參數判斷是否結束 dispatching 的歷程，使用者瀏覽的流程控制可以更有彈性，不會只限定只能執行某些工作，而能彈性調整，執行多項不同的工作。

Response process 非常單純，負責將 Zend_Controller_Response 物件中的資料輸出，讓使用者看到執行結果。

Routing process、dispatching process 及 response process 中的所有類別調度並不是各自為政，而是由 Zend_Controller_Front 類別統一調度配置，控制流程進行的節奏及順序，完成使用者的要求。

Appendix A3

圖 A3-1 至圖 A3-10 為圖 5-1 的作業流程驗證圖，紅色的方框，代表 enable 狀態的 transition，可以被 firing，每一張圖，代表 token 的分佈狀態的轉移，借以驗證作業流程中工作邏輯的合理性，如果作業無法轉移至下一個狀態，即表示作業流程規劃錯誤，此處採用圖形化的驗證，容易以直覺式的方式察覺規劃的謬誤，實務上，採用圖形化的驗證，也容易被程式設計師接受，降低使用門檻。

圖 A3-1 中，更新本機端人事資料的工作狀態為 enable，可被 firing，代表更新本機端人事資料工作準備執行，一旦被 firing，就會執行，更新本機端人事資料這項資源，以供後續作業流程運用。

圖 A3-2 中，計算薪水工作的 input place 為本機端人事資料，token 已移轉到本機端人事資料，計算薪水工作的狀態為 enable，可被 firing，計算薪水的工作被 firing，token 會移轉到人事資料這項資源，代表人事資料可被後續的流程運用，開始計算薪水。

圖 A3-3 中，計算特教津貼的 input place 為人事資料及特教津貼列表，此二項資源均有 token，因此計算特教津貼這項工作的狀態為 enable，可被 firing，被 firing 後，token 移轉到特教津貼，計算出特教津貼，轉換為圖 A3-4 的狀態。

圖 A3-4 中，計算本俸的 input place 為人事資料及薪額本薪對照表，此二項資源均有 token，因此計算本薪這項工作的狀態為 enable，可被 firing，被 firing 後，token 移轉到本俸資源，計算出本俸，轉換為圖 A3-5 的狀態。

圖 A3-5 中，計算導師費的 input place 為人事資料及導師費列表，此二項資源均有 token，因此計算導師費這項工作的狀態為 enable，可被 firing，被 firing 後，token 移轉到導師費，計算出導師費，轉換為圖 A3-6 的狀態。

圖 A3-6 中，計算實際本薪的 input place 為人事資料及最高本薪<->學歷對照表，此二項資源均有 token，因此計算實際本薪的工作狀態為 enable，可被 firing，被 firing 後，token 移轉到實際本薪，計算出實際本薪，轉換為圖 A3-7 的狀態。

圖 A3-7 中，計算研究費的 input place 為人事資料、實際本薪、本薪

<->研究費對照表，此三項資源均有 token，因此計算研究費的工作狀態為 enable，可被 firing，被 firing 後，token 移轉到研究費及實際本薪，計算出研究費，轉換為圖 A3-8 的狀態。

圖 A3-8 中，計算主管加給的 input place 人事資料，實際本薪、職務<->主管加給對照表，此三項資源均有 token，因此計算主管加給的工作狀態為 enable，可被 firing，被 firing 後，token 移轉到主管加給，計算出主管加給，轉換為圖 A3-9 的狀態。

圖 A3-9 中，薪水計算的 input place 為特教津貼、本俸、導師費、研究費、主管加給、人事資料，此五項資源均有 token，因此薪水計算的工作狀態為 enable，可被 firing，被 firing 後，token 移轉到薪水表，產生薪水表資源，以供後續的作業流程運用。

圖 A3-1 至圖 A3-10，表示圖 5-1 所規劃的工作流程所有的工作執行狀態，從這些驗證圖中，沒有發現不可達到的狀態或死結，驗證這是一個確實可行的流程規劃。

圖 5-2 及流程驗證皆以相同方式進行，運用的原理及方式均相同，因此不再進行贅述。

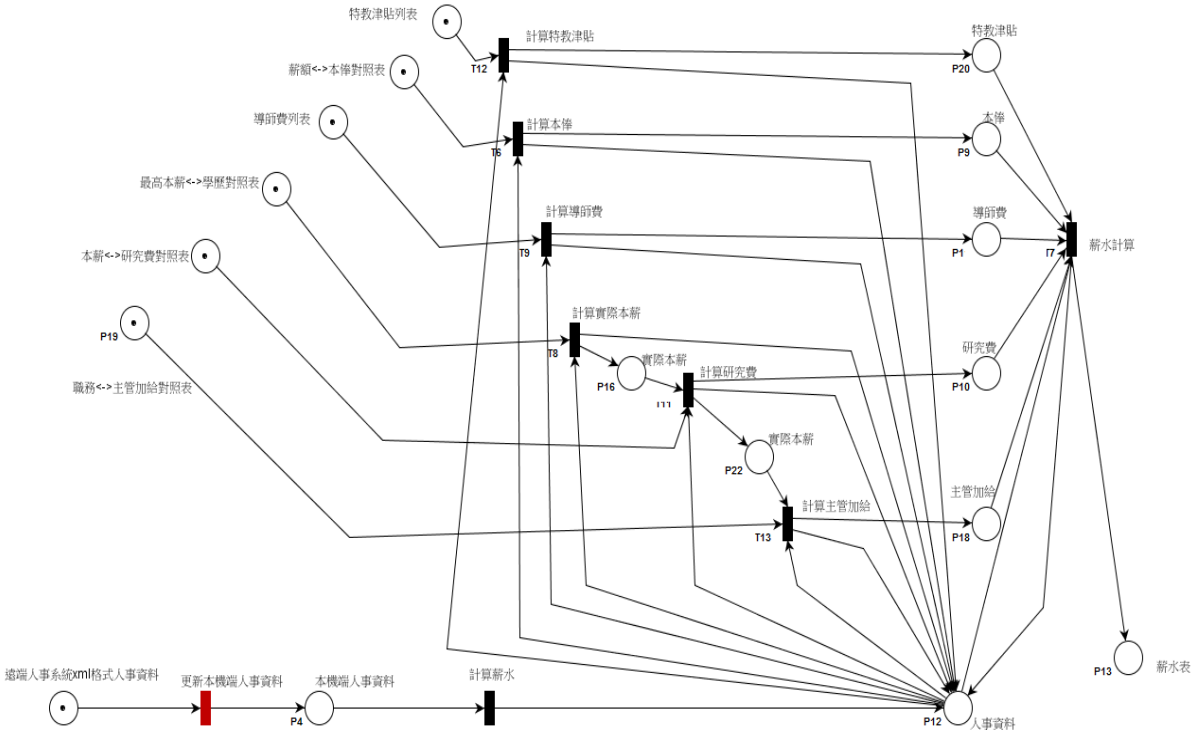


圖 A3-1：正職教師的薪水計算作業流程驗證圖(1)

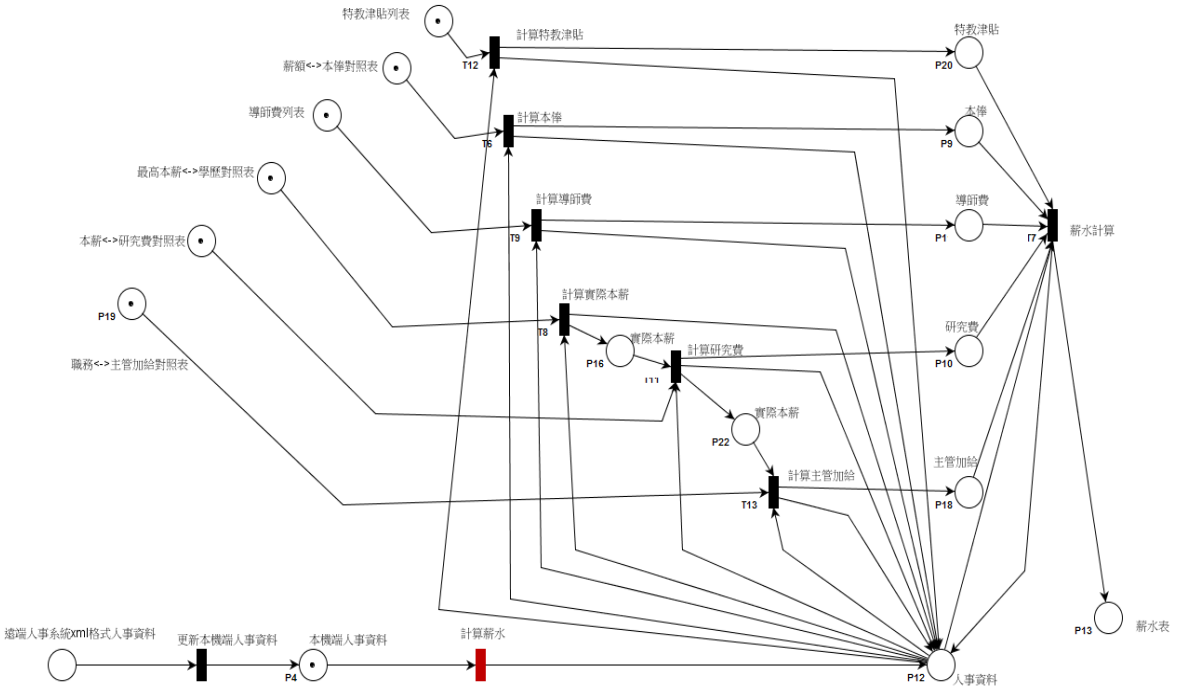


圖 A3-2：正職教師的薪水計算作業流程驗證圖(2)

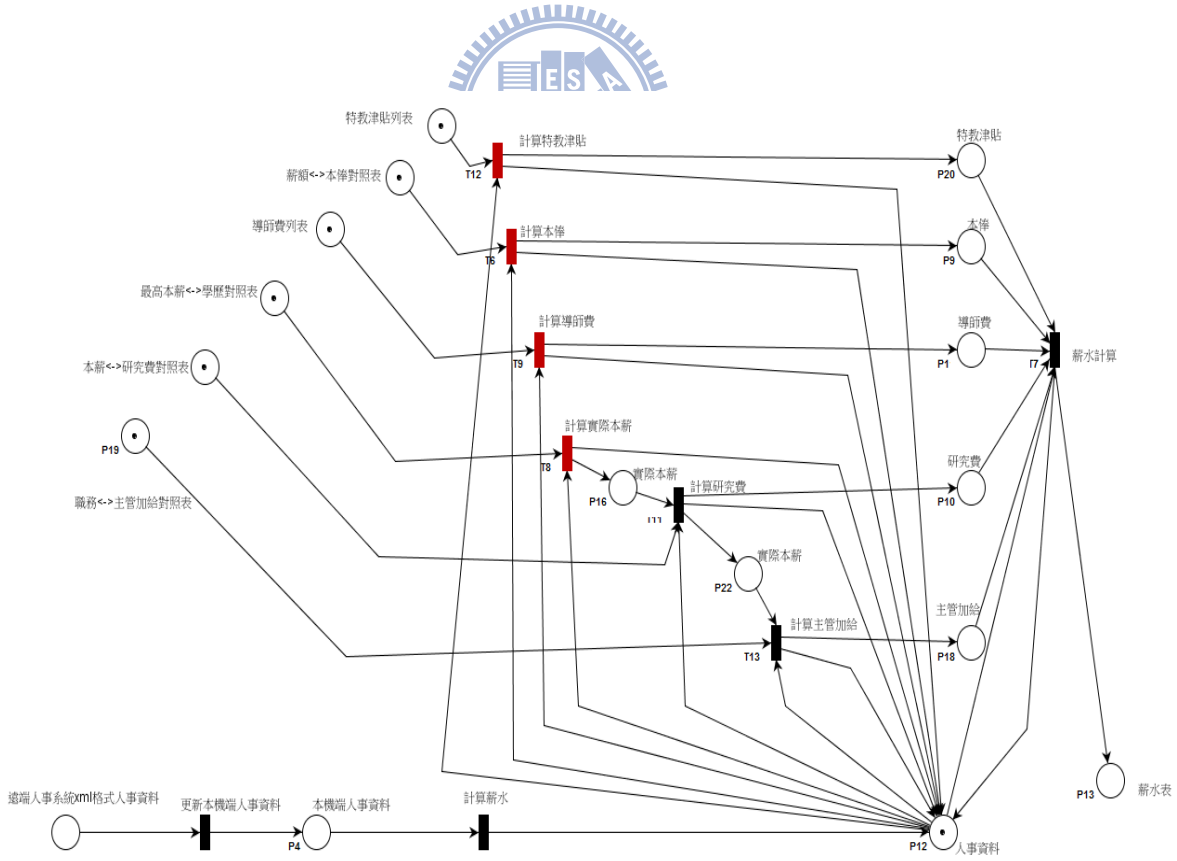


圖 A3-3：正職教師的薪水計算作業流程驗證圖(3)

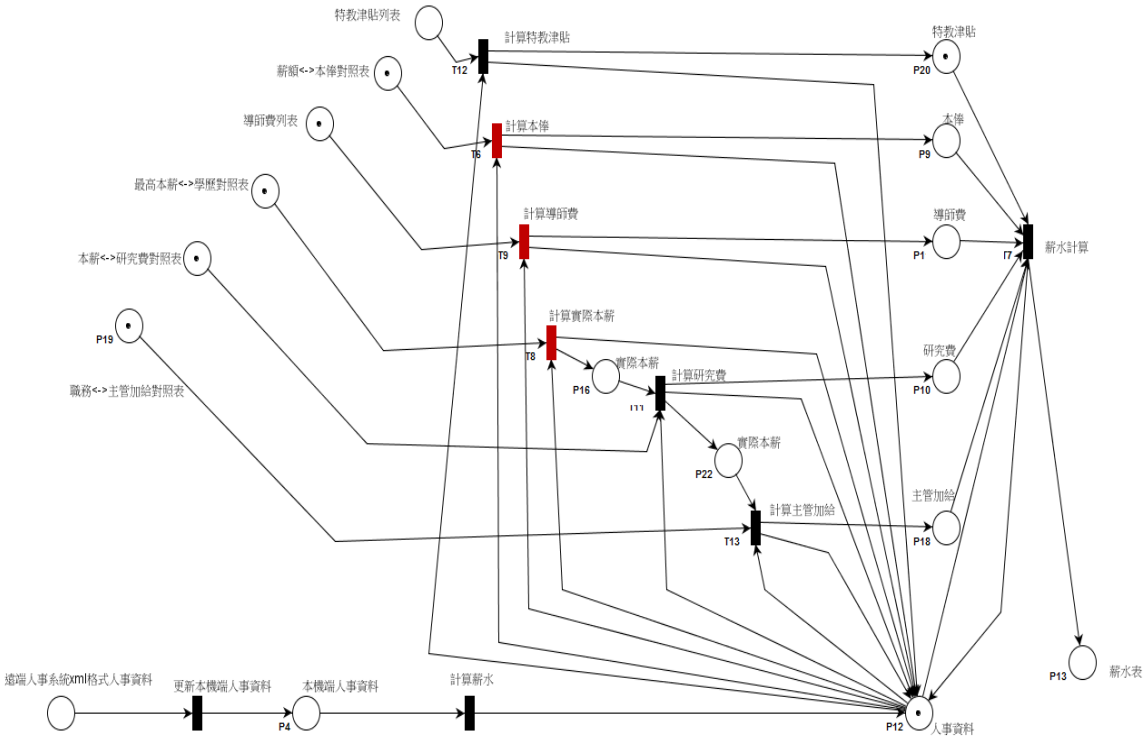


圖 A3-4：正職教師的薪水計算作業流程驗證圖(4)

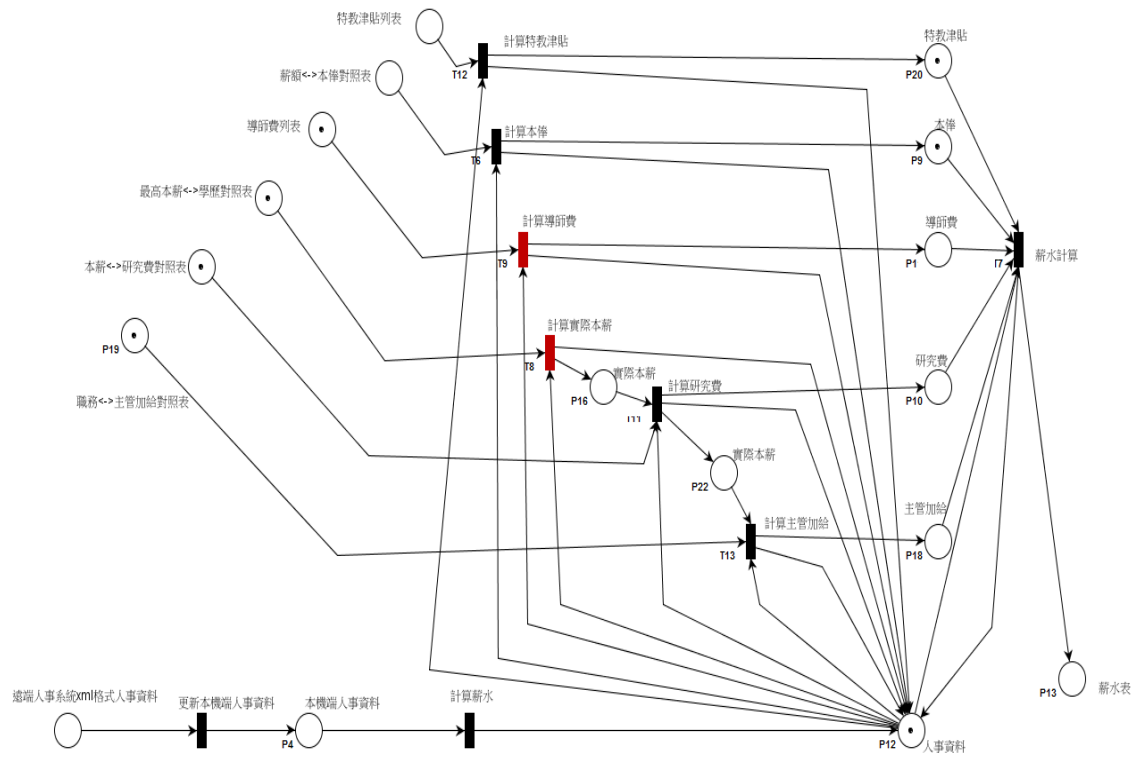


圖 A3-5：正職教師的薪水計算作業流程驗證圖(5)

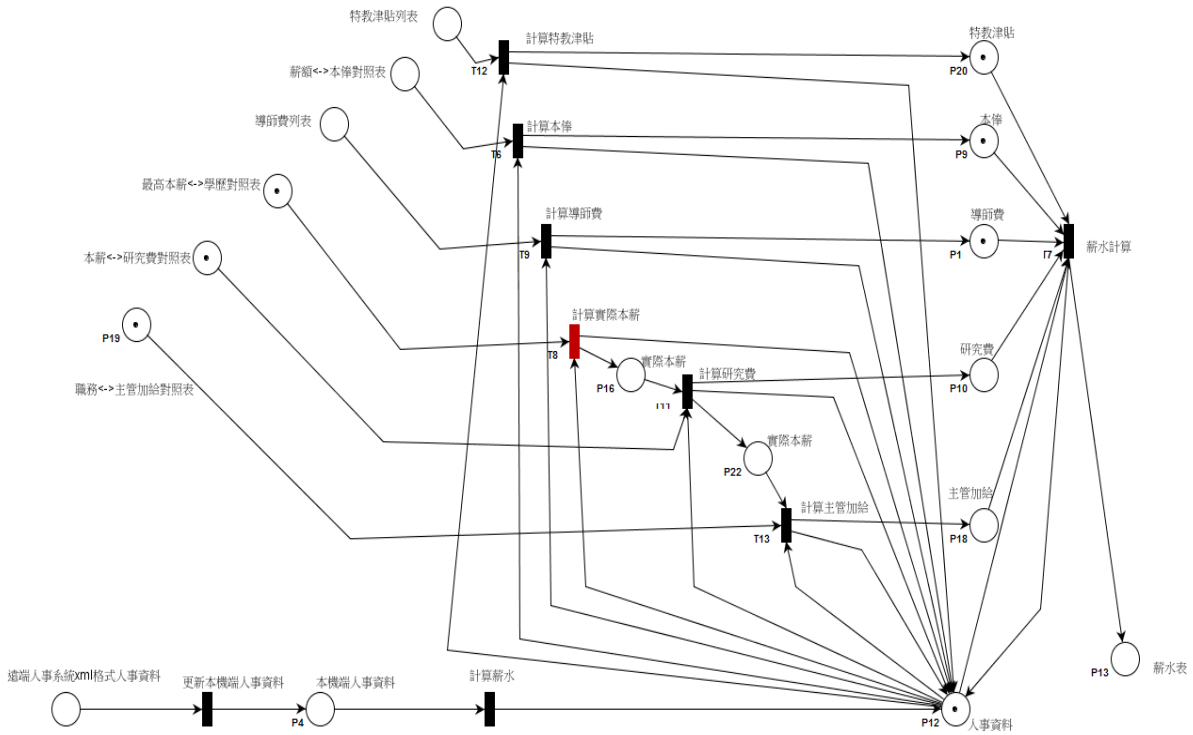


圖 A3-6：正職教師的薪水計算作業流程驗證圖(6)

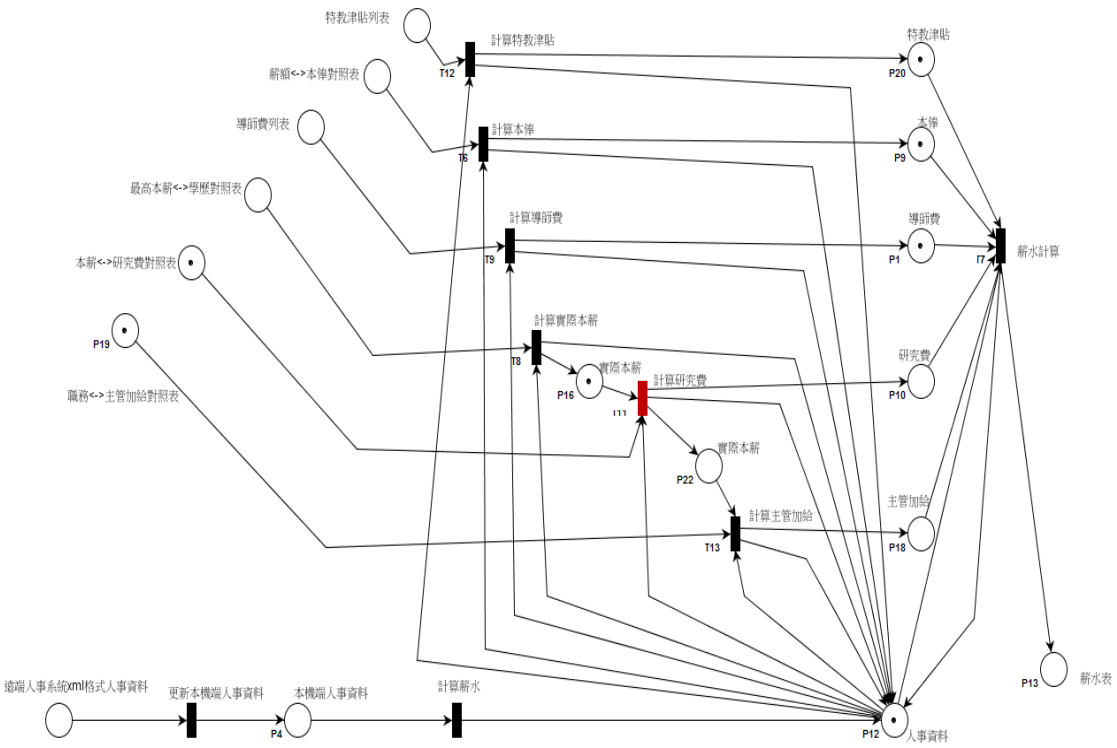


圖 A3-7：正職教師的薪水計算作業流程驗證圖(7)

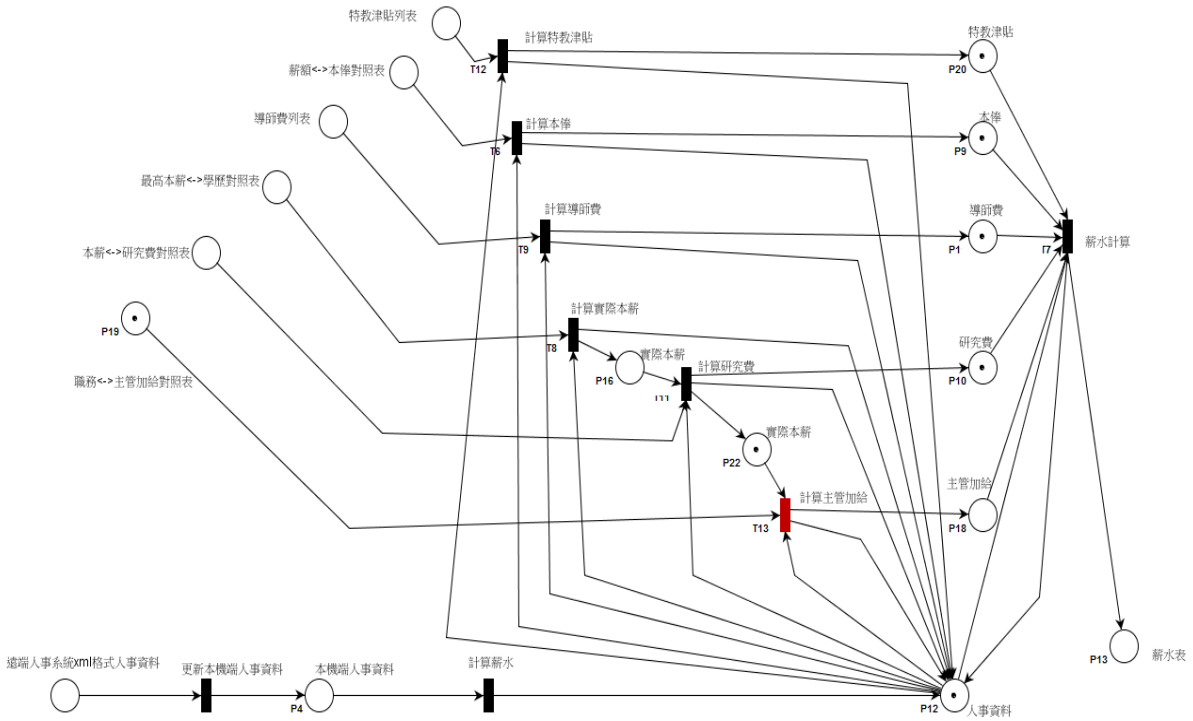


圖 A3-8：正職教師的薪水計算作業流程驗證圖(8)

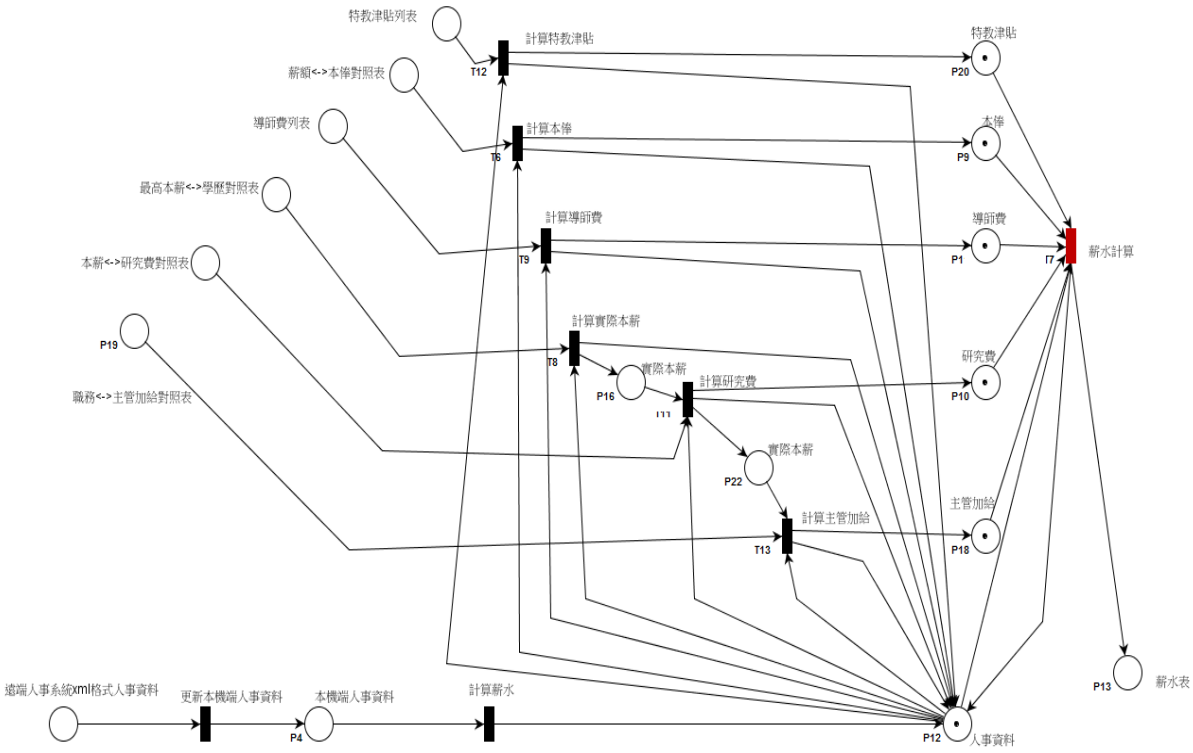


圖 A3-9：正職教師的薪水計算作業流程驗證圖(9)

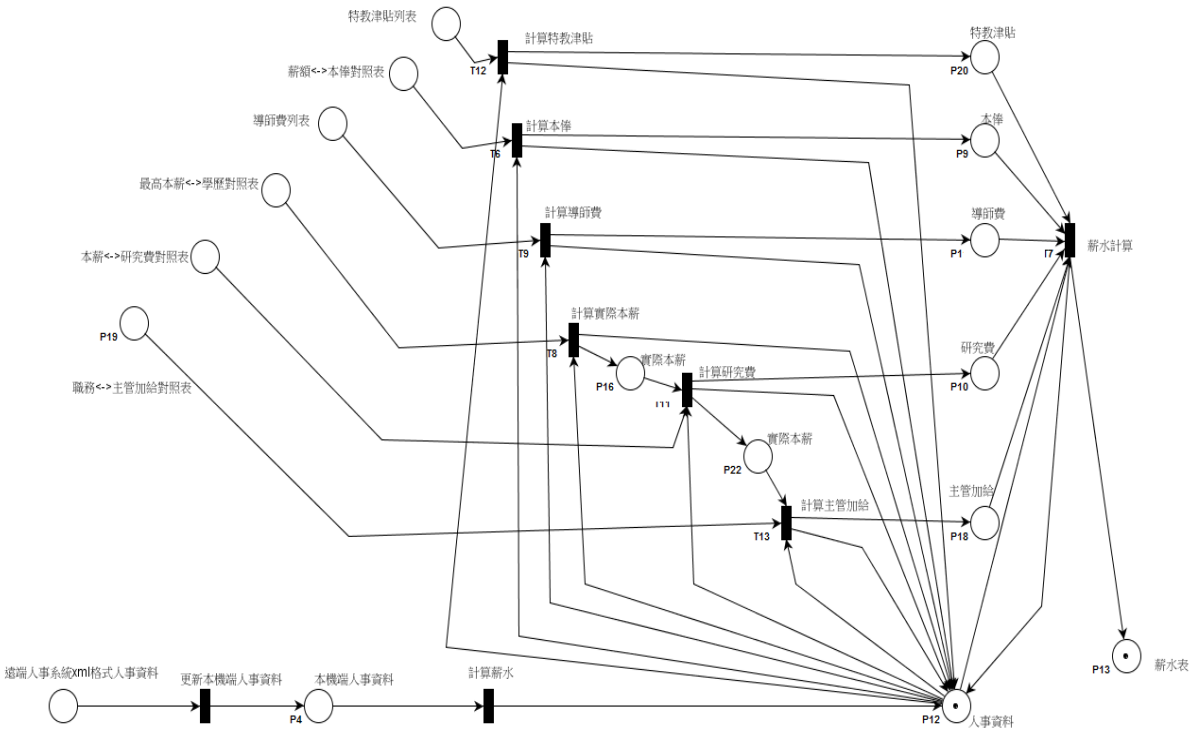


圖 A3-10：正職教師的薪水計算作業流程驗證圖(10)