赴英國渥維克大學進修電腦博士學位心得報告書

實務經驗模型於程式理解之應用

服務機關：聯勤後勤署整後組

出國人職稱：中校後參官

姓名：許興文

出國地區：英國

出國期間：八十六年六月二十四日－九十六月二十八日

報告日期：九十年九月九日

目錄

壹、前言

奉國防部八十六年六月四日(86)易旭字第一〇六一〇號令核准以八十六年度國軍官兵進修員額赴英國渥維克大學電腦博士班進修四年，自民國八十六年六月三十日至九十年六月二十九日止，茲將研究狀況作一摘要報告。

貳、過程

一、進修學校簡介

渥維克大學是英國前五大的其中一所大學，亦是全英唯一不需政府支助，能夠自己自足的學府，美國前任總統克林頓曾於西元二〇〇〇年應邀至該校演說，可見該校在英國之地位。目前全校之研究計有科學、社會學及藝術等三大領域，其中以科學類之數學、電腦科學及社會學之管理科學最為知名。

二、進修內容及論文概要

（一）**論文題目**：實務經驗模型於程式理解之應用（Empirical Modelling for Program Comprehension）。

（二）**論文摘要**：本論文主要就現行之程式理解模型為基礎，探討現行程式理解工具之限制因素，並結合對程式師之實務研究（Empirical Studies of Programmers）說明程式理解之挑戰與需求，其中對程式語言其本質在語義上的表達及人性因素之機制等問題，更用二個案例分析來描述以說明現行工具之不足，文中並導入實務經驗模型以建立一真正符合程式理解其本質需求的工具。

（三）**論文本體**：本論文共區分為七章，第一章介紹有關程式理解在軟體工程中所扮演之角色及其重要性；第二章在探討現行程式理解模型、工具

及其限制;第三章在介紹一新的實務經驗模型並說明其可突破現行工具之限制及符合程式理解模型之需求;第四章在根據文獻研究建立一有效的評估模式;第五章以二個實例說明實務模型所建立之工具於程式理解程序中的實際表現;第六章根據實例說明完成評估;第七章說明本論文之貢獻及未來發展方向。現僅將目錄及第一章部份置於附件。

參、心得

一、簡介

資訊技術是便捷的，企業體可透過資訊化的過程，使其整體作業的效率提昇、生產管理的時程及成本降低等，帶給企業的是產品本身之外的另一種競爭力，例如，縮短顧客等待服務的時間、提高產品的品質及產能、強化現

有的資源管理等皆是，尤其近年來資訊科技的快速發展，使資訊設備與企業體順利結合並逐漸成為企業體內重要資產，足以說明其重要性。

然而資訊化的過程是階段性的、冗長的、複雜的，尤其它是開放的，任何改變的產生（包含軟、硬體技術或企業管理模式的改變）都可使先前投資規劃之結果產生變動，軟體危機（Software Crisis）即是一九六0年代晚期人們意識到而逐漸予以重視的課題，這個問題截至今日尚無法完全解決，由程式語言發展的角度來看，傳統程式語言、工具及方法主要在於支援小型程式的發展，目的在解決局部或簡易的問題，然而，隨著日漸成長的軟體發展需求，大型且複雜的程式發展已無可避免，其關鍵因素—程式理解，是新一代軟體工程（Software Engineering）亟須滿足的需求。

二、軟體工程

軟體是為解決實務世界問題而設計的方法，並透過軟體發展的模式創造而成的；更仔細的說，軟體是由某待解問題所屬之專業領域知識（Domain Knowledge）、專業領域方法(Domain methodologies)及專業領域資料(Domain data）所構成，透過程式發展者對待解問題及其解決方法的解譯而據以撰寫完成的程式集，這種程式集可為電腦或其軟體解讀並執行。由邏輯的角度來看，軟體猶如一嚴謹的規格，說明理論上解決問題的程序，即任何程式組成元件的存在均奠基於理論上的需要，與實務上解決問題的程序並無絕對關係，依循此規格必可獲致解決問題的答案；但是就理解的角度而言，軟體規格本身僅為理論上獲致問題答案的程序，並不保證與實務世界問題的解決程序一致，因此，是否能由此軟體規格產生對問題解決方法的正確理解，則須俟理論上軟體程式規格與實務上問題解決程序是否接近而定。

傳統的程式撰寫較注重個人主觀的發展模式，例如，僅重視輸出/入的正確性而不講究理論（程式內涵）與實務經驗（問題解決程序）的砌合程度，因此，當程式複雜度高時，這種認知差距就會造成程式理解上的困難。軟體工程即在於改進因這種認知差距帶來的問題，其程序是由軟體發展過程中萃取重要關鍵因素，加以識別所產生，這種程序建立之模型具有分析、設計、程式撰寫、測試及維護等不同階段，並具有文件化（Documentation）、抽像化（Abstraction）、視覺化（Visualisation）及模組化（Modularizations）等技術之運用，使軟體發展於不同階段都能保持理論與實務經驗上認知的一致性，並據此衍生出控制流或資料流等不同的模型，透過比較分析可找出情境（Situation）下適用者；由此可知，程式撰寫不僅須考量其是否能解決問題，還須包含其理論上解決方法是否符合實務經驗而定。

軟體工程置重點於建立軟體模型所須的理論、方法與工具，又由於軟體系統的發展已走向大型化、複雜化，尤其軟體並無具體形式，因此，軟體工程有別於其它工程，其目的是將大型、抽像及複雜的軟體在其具體的管理下，產生維護成本低的軟體，其中須考量軟體本質上的二項重要因素：

- 軟體主要在解決多變性實務世界的問題，因此當實務世界發生變化時，軟體也須隨之變動。
- 軟體很難找到語義上相同的二個組成元件，不像其它如建築、汽車等工程，可供實體模型之建立。

由以上的說明不難看出軟體本質所具有的複雜度，這種複雜度不能只靠著程式語言中結構化的功能解決，即便結構化功能可過濾繁瑣的資訊進而表現與實務上相關的資訊，但其概念產生之程序如分析、設計及確認仍取決於人為

因素，因此在軟體發展與理解的過程中，程式設計師的思考模式，不再只是程式語法的安排而是實務概念的落實，著重於理論與實務上概念的辨識與組成。

軟體發展除有上述的複雜度之外，尚有：

- 依從性（Conformity）：軟體須依從外在現存之介面以遂行其工作，例如電子郵軟體件須依從複雜的網路協定。
- 可變性（Changeability）：軟體的經常性改變是理所當然的，例如使用者提出或改變其需求。
- 不可見性（Invisibility）：許多圖表雖可依軟體產生，例如流程圖、資料流圖及結構圖等，但沒有一種被認可與軟體有完整的對應關係。

綜合來說軟體工程是一種程序，主要包含分析、設計、及維護三個階段，而

以降低各階段成本、時間及效率為主，其範疇可概略定義如下：

- 分析現行軟體系統及使用者需求。
- 瞭解現行系統以適切發展自動化。
- 改善現行系統邏輯設計以降低軟體複雜度。
- 降低因系統功能更動而發生的負面效應。

三、逆向工程/再造工程（Reverse Engineering/Re-engineering）

由上節可知軟體工程涵蓋範圍相當廣泛，一般可區分為順向工程（Forward Engineering）、逆向工程（Reverse Engineering）及再造工程（Re-engineering），略述如後：

- 順向工程：為傳統的軟體發展程序，包含分析、設計至產品（含文件及使用手冊）的產生。

- 再造工程：透過為對現行軟體系統的再造，產生更具維護性的軟體。
- 逆向工程：從現行軟體系統的資源，產生軟體系統另一種設計或規格。

逆向工程不同於再造工程是前者由程式中找出原程式之另一種設計表達方式，而後者須拋棄原程式的束縛，企圖運用全新的技術重新發展軟體；在維護的角度上它們都有卓著的貢獻，但是要成功的執行這二種工作，還必須對軟體有充分的理解。

再造工程須具有軟體轉譯（Software Translation）及軟體再結構（Software Restructuring）二種功能：

- 軟體轉譯：由軟體中邏輯的對應關係，透過轉換的定義達成轉譯的功能，通常可以自動化方式完成。

● 軟體再結構：由軟體中識別其組成功能以及各功能間之關係，並藉重組的規則達成再結構的功能，通常須由人為方式達成。

這二種功能的選擇是因情境而定，主要以知識獲取為核心，而交談的方式又是知識獲取的手段，因此，再造工程可運用此種交談的特性調解自動與人為的介入，即當共識形成，自動化可以實現，若否，則人為因素就會介入，又由於這二種模式在再造的過程中為交互發生，因此，再造工程在本質上應以交談為核心，為遞迴方式逐次修正模型的程序；程式理解其本質亦具有再造之特性，當程式本身或文件說明不具完整性時，它須要透過人為方式對程式作認知、憶測來決定理解程式的策略，並以實作的方式確認其正確性；而實務證明，解決程序與程式元件具明顯對應關係者，較易獲致理解，因此，程式理解的基本前提是由程式中切割並識別其於實務世界所對應之概念，如

此，程式元件（Program Components）不再是僅與語法相關，如函式（Subroutines）、模組（modules）或物件類別（Classes）元件，而是與語義相結合的概念元件（Conceptual chunks）。

雖然這些概念元件因情境因素而異，因此，沒有固定的形式，須隨時予以修正，以合時宜，但重要的是，當這些概念藉由交談模式（Interactive Mode）一一被植入系統發揮作用之後，系統應能就各不同概念作出反應，亦就是將程式發展過程透明化。又由於這種概念均由實務層發展而來，因此，易於確認各概念於程式中所扮演的角色。

四、軟體維護（Software Maintenance）

就軟體工業而言，維護現有系統之成本約佔程式撰寫的百分之五十至百分之七十，而程式理解約佔軟體維護的百分之三十至百分之六十，可見程式

理解之重要性。在軟體的生命週期中，軟體維護是自始至終與軟體脫不了關係的，由於維護行動的執行不像實體物件之修護，它可按使用者功能需求更改軟體系統的任一部份，甚至與原系統需求衝突或不一致的狀況都可能被接受，致使原有軟體維護難易程度隨之改變。軟體維護概分為四類：

- 改進軟體系統功能符合使用者需求。
- 更正軟體系統發展時發生的錯誤。
- 修正軟體系統符合外在硬體環境的改變。
- 改變軟體系統提昇維護能力。

而這四類之維護工作皆以程式理解為核心。

五、程式理解（Program Comprehension）

由上述幾節的描述可知，軟體維護是軟體發展的核心，而軟體維護又以

程式理解為重點，由此可見程式理解的重要性。然而程式雖可輔以文件增加理解之效果，但文件常因人們不經意的疏忽而導致不適用的情況，最後程式本身被公認為可供程式理解的惟一資源，這也是系統維護上窒礙難行的因素。程式理解在軟體工程中扮演著重要的角色，同時它也是逆向工程及再造工程的重要一環，若於軟體維護時對軟體已有相當之理解，則維護行動將有較大的成功率。目前，程式理解的模型及工具均是以實務研究的結果為依據，例如，程式維護師依狀況會採取不同的策略以完成軟體理解進而達成軟體維護之目的。

廣義的說，系統維護者之所以能夠完成瞭解系統的工作，是因為他們可將程式所表達的概念用人類世界的語言表述。反之，人類世界的語言具有精細且經驗導向的特性，有時反而不易於用程式語言表達，因此有可形式化

（Formal）與不可形式化（Informal）二種，前者是所指概念可為程式語言表示，後者則不可為程式語言表示；因為程式理解的主要工作是從程式碼中辨識出原程式師所使用的設計理念並找出這些理念與待解問題的關係，最後獲得的是維護者對軟體的認知模型，在建立模型的過程中，維護者的經常性介入、提出假設、憶測、實驗、確認及更改等需求，充分顯示概念獲取及交互確認之重要性。概念獲取主要是由程式的語法中辨識出與概念有關的部分並進而以抽象化的方式將這些部分整合為一完整的概念，例如，當插入（Insert）的動作與串列（Link-list）的資料型態相結合時，移除（Remove）的行動自然就會被納入概念獲取的考量；此外，交互確認可供不同層次觀點的整合與確認，例如，概念的可以不同的抽象程度表示，其間程度之決定由需求而定。

當維護者處理的是概念元件而非程式元件的時候，其所處的系統狀態可能是不完整的，所謂不完整是指系統尚不具備所有解決問題的能力，而且維護者通常一次僅能處理5~7個程式元件，因此，理解模型若能納入不完整狀態並顯示目前的理解狀況，相信維護者較易突破瓶頸，達到程式理解之效果。

六、程式理解之挑戰性（Challenges in the Research）

程式理解是公認為具高度挑戰性的題目，它主要須克服下列認知差距的問題：

- 實體世界的問題與其求解程式間之差距。
- 實體世界的硬體環境與軟體程式間之差距。
- 實體世界系統結構與軟體系統結構間之差距。
- 人類認知架構與軟體系統架構間之差距。

● 應用面由上而下與程式面由下而上之本質差距。

（一）**應用及程式領域**（Application and Program Domain）

從應用的角度來看，程式是一種問題的解決方法的描述，其本身不包含對問題的描述，因此，僅變數名稱及註解可供作長期的提示，此外，不可形式化的提示常因時過境遷而不適用或不存在，面對這種困難，目前程式理解工具的自動化僅限於程式本身為主，超出這個範圍，如涉及程式語言之外不可形式化的應用領域，其自動化將會產生極大的困難。

（二）**實體機器與抽象描述**（Physical Machines and Abstract Descriptions）

程式是鉅細靡遺的，其本質為電腦內部龐大記憶單元的中樞。在逆

向工程中的首要工作就是從這些細項資料中，理出重要的概念，通常這程序稱為抽象化，抽象化後之概念具有不同的層次及架構，若一概念（含分佈在程式不同區域程式碼之組合）被辨識出來後，其概念及架構可被抽象化為更接近人為思考模式的架構。

（三）砌合的模型與不砌合的造具（Coherent Models and Incoherent Artifacts）

程式最初被建立的時候，其結構之設計與程式為一致的，但在設計與展示方面，由於大量新技術的投入，使程式語言即便作了抽象化及結構化的加強，依然無法解決技術改變導致設計概念與程式語言的不砌合，而這又為程式理解所必須，而目前軟體維護所引入的改變，更擴大了這種不砌合的程度。

（四）層級的程式及認知（Hierarchical Programs and Associative Cognition）

其它致使軟體難以理解的原因尚有：

- 軟體須遵循其外在所給定的介面。
- 軟體須應使用者需求不定期更動。
- 軟體難以用實體方式呈現。

因此，軟體工程應致力於發展整合專業領域（Integrated Domain）用以結合實務經驗與程式理論二不同專業領域，而此整合專業領域之發展則須奠基於實務與程式二不同領域的共同意識（Consensus）之上。

正因為此種特性，使得程式邏輯的複雜度因程式師具備之專業程度而有不同，就目前實務模型的內涵，這些經驗並不是憑空獲得，而是具

有文化背景的，這種文化背景可能與個人所接受的教育有關，可能與個人的行為個性有關，也可能與個人的策略運用有關。

肆、建議

一、綜觀國際各先進國家，無不擁有強大之國防，其所展現之實力均以科技作後盾。反觀台海情勢，面對敵之覬覦及因應縱深短淺之作戰需求，建立反應速度快且精實的國防力量更是目前國軍防禦體系下之首要任務，而其也應以科技作後盾。為掌握及結合科技的脈動，在國軍現有人才培育豐碩成果上，建議能持續進行此深具意義百年樹人的工作。

二、軍人亦是社會的成員，一旦退伍後其身份自會由軍人回復為一般百姓，以軍中目前有計畫之培養，再考量未來民間之需要，便可對社會有所貢獻，再則可透過民間之經驗交流，對軍中需求作出有效的考量，如此良

性互動，勢必對軍民一體的全民國防有所助益。

三、目前職服務於後勤署整後組，本組之任務特性為造就一高度效率之後勤支援體系，其根本在引入科技以達成組織再造的需求，其中，對資訊科技的運用如軍備壽期資訊管理（CALS）等項目，可看出組織精實與資訊科技結合之迫切性，建議就此需求強化人才培育工作。

四、職於英國求學期間，深感英語文與科技發展之關聯性極高，例如電腦網頁服務與期刊論文發表等；建議能在國軍現有英語文人才培育計畫下，再普遍提昇各階層之英語文能力，勢必對軍事科技發展有實質的幫助。

五、資訊的應用可說無遠弗界，其重要性可由當今網路使用的程度輕易看出，目前國軍網路業已建置且本部基層各單位均能透過電腦連線與本部主電腦聯合作業，資訊人員素質亦有提昇，建議強化人員經管制度，避

免人才流失，以因應未來因組織整合而產生之資訊作業需求。

附件一

論文之目錄

TABLE OF CONTENTS