國立交通大學

資訊科學與工程研究所

碩 士 論 文

『問題－解法』軟體知識庫之設計與實作

Design and Implementation of Problem-Approach-based Software Knowledge System

研 究 生：林志忠

指導教授：鍾乾癸　教授

中 華 民 國　九 十 六　年 六 月

『問題-解法』軟體知識庫之設計與實作

Design and Implementation of Problem-Approach-based Software Knowledge System

研 究 生：林志忠　　Student：Chi-Chung Lin

指導教授：鍾乾癸　　Advisor：Chyan-Goei Chung

國 立 交 通 大 學

資 訊 科 學 與 工 程 研 究 所

碩 士 論 文

A Thesis

Submitted to Institute of Computer Science and Engineering

College of Computer Science

National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of

Master

in

Computer Science

June 2007

Hsinchu, Taiwan, Republic of China

中華民國九十六年六月

『問題-解法』軟體知識庫之設計與實作

研究生：林志忠　　指導教授：鍾乾癸

國立交通大學資訊工程系碩士班

摘要

軟體發展過程中，軟體工程師遇到問題時常在技術期刊或技術報告尋找解答，但一般期刊資料庫、技術報告庫或公司技術資料庫的技術文件大都依年代、作者或文章名稱等作分類，軟體工程師需花甚多時間才能找到所需文件；針對此缺點，謝祖望學長[6]提出「問題-解法」知識分類法，並以一雛形系統證實可快速找出相同問題及解法之技術資料，唯其雛型系統過於簡單，應用於私人知識庫、技術報告庫及期刊知識庫時，仍有缺乏技術關鍵字逆向查詢功能、缺乏巨量資料處理功能、缺乏文件編碼機制等等缺點，本研究運用「問題-解法」知識分類法進行私人資料庫、技術報告庫及期刊資料庫之設計。

本研究首先依據「問題-解法」知識分類機制及技術文件數量提出一個十欄之 64-bit 知識編碼機制，可同時用於書本及技術論文之編碼，進而提出關鍵字逆向索引機制以快速找出相同解法或技術之論文或技術報告，最後提出一個適用於巨量文件資料之分散式儲存機制，運用這些機制分別提出技術報告庫、私人知識庫、期刊知識庫及全文資料庫之系統架構，並實作一技術報告庫雛形系統，藉以驗證上述機制之實用性，證實「問題-解法」知識分類法可供軟體知識師依問題、技術或解法快速找出所需技術文件，且可與現有知識搜尋機制並存。

Design and Implementation of Problem-Approach-based Software Knowledge System

Student：Chi-Chung Lin Advisor：Chyan-Goei Chung

Department of Computer Science and Information Engineering
College of Electrical Engineering and Computer Science
National Chiao-Tung University

Abstract

During software development, software developers often search solutions from books, technical reports, and papers when they encounter problem. Technical Documents in journal databases, technical report database, and private knowledge database are classified by publish date, author name, or title. It often spends much time to find the right documents. Because of this drawback, Tzu-Wang Hsieh[6] proposed the Problem-Approach-based Software Knowledge Classification. He also implemented a prototype system which shows that documents solving the same issue or using the same approach can be found very quickly.

Due to the prototype system is too simple, when it is applied to private knowledge database, technical report database, and journal database, it has some drawbacks. First, the system lacks supporting for inverted search using keyword. Second, it lacks the ability of processing huge amount of documentations. Finally, it lacks encoding mechanism for documentations and keywords. This research uses the Problem-Approach-based software knowledge classification to design system architectures for private knowledge database, technical report database, and journal database.

A 64-bits code with ten fields based on the Problem-Approach-based software knowledge Classification technique is proposed to represent the ID of a technical document which can be a technical paper, technical report, or book chapter. Base on the encoding mechanism, we propose an inverted index mechanism to store technical

documents with the same approach or technique. Finally, we propose a distributed knowledge management mechanism for huge amount of documents. With these mechanisms, System architectures for technical report database, private knowledge database, journal database, and full-text database are proposed. A prototype system for technical report database is also implemented. With the prototype system, software developer can quickly and precisely find needed technical documents by using issue, approach, or technique keywords. The conventional search mechanism, such as by author name, year, keyword, etc, is also implemented in the prototype system.

致謝

在就讀碩士班期間，非常感恩鍾乾癸老師的教誨，讓我學習到許多做學問的方法與態度，在往後的人生中也能受用不盡。這篇論文能夠完成，也因為老師不厭其煩及細心的教導，在此對老師獻上最高的謝意。

此外，感恩鄭靜紋學姊的細心指導與協助，幫助我度過許多不知如何處理的情況。感恩禪學社與領袖社夥伴的關心與祝福，同時也感恩父母親及親友的支持與鼓勵，伴我度過撰寫論文的日子。

最後感恩十方法界諸佛菩薩，在此以最虔敬的心獻上此成果。

目錄

第 1 章 緒論 12
1.1 節 研究背景與動機 12
1.2 節 各章節介紹 14
第 2 章 相關研究與背景知識 16
2.1 節 知識庫類型 16
2.1.1 私人知識庫 16
2.1.2 期刊資料庫 18
2.1.3 技術報告庫 23
2.1.4 全文資料庫 25
2.1.5 目前知識庫的查詢方法之缺點 25
2.2 節 『問題-解法』知識分類法 26
2.2.1 技術論文與技術報告本質 26
2.2.2 書本知識本質 27
2.2.3 『問題-解法』關鍵字架構 29
2.2.4 『問題-解法』知識庫雛型系統 31
2.3 節 『問題-解法』知識庫雛型系統之缺點 34
第 3 章 『問題-解法』知識庫設計構想 38
3.1 節 使用者角色分類 38
3.2 節 Survey 類型論文性質 41
3.3 節 大量資料儲存與查詢問題 46
3.3.1 技術報告庫 48
3.3.2 小型私人知識庫 54
3.3.3 大型私人知識庫 58
3.3.4 期刊資料庫 61
3.3.5 全文資料庫 62
第 4 章 技術報告庫之系統實作 63
4.1 節 技術報告庫之規格 63
4.2 節 伺服器端之資料結構 95
4.3 節 技術報告庫系統之設計 109
第 5 章 其他類型知識庫系統之設計 129
5.1 節 私人知識庫之設計 129
5.2 節 大型私人知識庫之設計 150
5.3 節 期刊資料庫之設計 154
5.4 節 全文資料庫之設計 158
第 6 章 結論 160

圖目錄

圖 2-1 私人知識庫新增知識流程 18
圖 2-2 私人知識庫的分類架構範例 18
圖 2-3 ACM 分類架構範例 22
圖 2-4 期刊資料庫新增知識流程圖 23
圖 2-5 期刊資料庫分類架構範例 23
圖 2-6 技術報告庫新增知識流程 24
圖 2-7 技術報告庫分類架構範例 24
圖 2-8 技術論文/報告關鍵字示意圖 27
圖 2-9 技術論文/技術報告範例 27
圖 2-10 書本知識關鍵字關係 28
圖 2-11 書本關鍵字關係範例 29
圖 2-12 『問題-解法』知識庫關鍵字架構 31
圖 2-13 知識庫雛型系統架構圖 33
圖 3-1 私人知識庫、技術報告庫新增知識流程 38
圖 3-2 期刊資料庫新增知識流程 39
圖 3-3 Survey 論文關鍵字架構 43
圖 3-4 Survey 論文範例 1 44
圖 3-5 Survey 論文範例 2 45
圖 3-6 發展成熟領域的 Survey 論文知識架構 46
圖 3-7 未發展成熟領域的 Survey 論文知識架構 46
圖 3-8 四種知識庫改善的關係 48
圖 3-9 解法列表查詢編碼示意圖 50
圖 3-10 領域類別知識列表儲存格式 51
圖 3-11 技術報告儲存資料 52
圖 3-12 知識編碼 53
圖 3-13 論文/報告逆向查詢檢索資料 54
圖 3-14 書本知識資料 56
圖 3-15 書本知識編碼 57
圖 3-16 書本知識逆向查詢資料 57
圖 3-17 期刊、Volume、Issue、論文之儲存資料 58
圖 3-18 大型私人知識庫架構 59
圖 3-19 知識分散範例 59
圖 3-20 期刊資料庫系統架構 62
圖 4-1 系統架構初步規劃圖 64
圖 4-2 使用案例圖 65
圖 4-3 新增報告流程 69

圖 4-4 新增報告流程所產生的檔案 69
圖 4-5 關鍵字檔案格式 71
圖 4-6 審核報告流程 80
圖 4-7 分割領域類別介面 1 91
圖 4-8 知識庫模組架構 94
圖 4-9 記憶體快取模組類別圖 95
圖 4-10 領域類別節點記憶體儲存資料示意圖 97
圖 4-11 領域類別節點儲存在硬碟的資料格式 98
圖 4-12 利用知識編碼建立領域類別節點之間的連結 99
圖 4-13 技術議題節點儲存資料示意圖 100
圖 4-14 技術議題儲存在硬碟中的格式 100
圖 4-15 解法節點儲存資料示意圖 100
圖 4-16 解法節點儲存在硬碟中的格式 101
圖 4-17 技術報告節點儲存資料示意圖 101
圖 4-18 技術報告儲存在硬碟中的格式 102
圖 4-19 Survey 議題節點儲存資料示意圖 103
圖 4-20 Survey 議題儲存在硬碟中的格式 103
圖 4-21 Survey 報告節點儲存資料示意圖 104
圖 4-22 Survey 報告儲存在硬碟中的格式 104
圖 4-23 子議題列表檔案格式 105
圖 4-24 關鍵字列表與逆向查詢資料 106
圖 4-25 關鍵字列表模組類別圖 106
圖 4-26 系統 Layer 架構圖 109
圖 4-27 使用者登入流程 118
圖 4-28 瀏覽知識流程 119
圖 4-29 檢視報告流程 120
圖 4-30 以關鍵字查詢報告流程 121
圖 4-31 提交技術報告流程 122
圖 4-32 建議新增領域流程 123
圖 4-33 審核技術報告流程 124
圖 4-34 加入技術報告流程 125
圖 4-35 加入領域類別流程 126
圖 4-36 搬移領域類別流程 127
圖 4-37 分割領域類別流程 128
圖 5-1 私人知識庫模組架構 143
圖 5-2 領域類別節點儲存在硬碟的資料格式 144
圖 5-3 領域類別節點記憶體儲存資料示意圖 145
圖 5-4 技術報告節點儲存資料示意圖 145

圖 5-5 技術論文儲存在硬碟中的格式 146
圖 5-6 Survey 論文節點儲存資料示意圖 147
圖 5-7 Survey 報告儲存在硬碟中的格式 147
圖 5-8 書本知識節點儲存資料示意圖 148
圖 5-9 書本知識儲存在硬碟中的格式 149
圖 5-10 小型知識庫之架構.....150
圖 5-11 大型私人知識庫的架構.....151
圖 5-12 大型私人知識庫模組架構.....153
圖 5-13 期刊資料庫系統架構.....155
圖 5-14 技術報告節點儲存資料示意圖 155
圖 5-15 技術論文儲存在硬碟中的格式 156
圖 5-16 Survey 論文節點儲存資料示意圖.....156
圖 5-17 Survey 報告儲存在硬碟中的格式.....156
圖 5-18 期刊資料庫模組架構.....158
圖 5-19 XML 轉換格式範例.....159

表目錄

表 3-1 使用者角色分類表 41
表 3-2 四種知識庫之差異 48
表 3-3 技術報告庫知識節點儲存之基本資料一覽表 49
表 3-4 ACM 子類別個數統計表 53
表 3-5 私人知識庫知識節點儲存資料 55
表 4-1 知識儲存資料量估計表 66
表 4-2 各 Servlet 處理訊息 112
表 5-1 新增或修改的使用案例 130
表 5-2 私人知識庫 Servlet 新增的命令 144
表 5-3 PaperAgent、BookAgent 的方法列表 149

第1章 緒論

1.1節 研究背景與動機

軟體產業是一項人力與知識密集的產業 (Birk, et. al., 1999)，軟體發展更是一個知識不斷創新的過程。由於資訊硬體產品的效能及功能大幅提升及價格大眾化，導致各類軟體需求愈來愈多，功能也漸趨複雜，故軟體知識快速且多樣化地成長，造成發展軟體時常遭遇開發過程費時、交貨延遲、產能不易提升、品質不易確保等問題，甚至產生了軟體危機。故知識管理對軟體產業是刻不容緩的課題。

體認軟體知識管理的重要性，對軟體知識種類與型態、軟體知識螺旋、軟體重用元件、軟體知識庫及軟體發展環境之研究與實作等議題進行一系列研究後，軟體的知識管理必須與軟體開發流程緊密結合，在不同的階段中，發展者需要不同類型的知識，在每一個階段中，遇到不同的問題，發展者亦需要相對應的知識作為輔助。然而，隨著知識庫中的知識愈來愈多，要如何有效地利用這些知識變成一個重要的課題。

知識庫的效率與知識分類方法有密不可分的關係，最常見的知識分類法大致可以分為四類，即 Taxonomy、Faceted Classification、Case-Based Reasoning、Ontology。本實驗室學長謝祖望針對上述四種分類法進行分析，並提出其缺點，進而提出『問題-解法』軟體知識分類法，從技術論文或技術報告的本質分析，發現這兩種知識主要內容都是為了解決某項議題，而提出其解決方法。議題必定存在於特定領域之下，而解決方法則可能運用了其他的技術，因此，描述一篇論文，只要描述這篇論文所在領域、解決的議題、採用的解法及運用的技術便能說

明其核心意義。故能將論文關鍵字分為「領域類別」、「議題」、「解法」、「技術」四個類型，一篇論文依此方法定義關鍵字便能表示其意義。採用上述方法表示技術知識後，可非常容易找出兩篇論文間之相關性，如解決相同議題的論文、採用相同解法或技術的論文、改善前人解法的論文，利用這些關係能提供知識間的關聯性，依議題、解法、技術找出同性質的論文，提昇搜尋的精度與準度。

謝祖望學長的研究實作一套雛型系統，以驗證『問題-解法』軟體知識分類法的可行性，唯此雛型系統有下列不足之處：

1. 缺少如何從解法、技術等關鍵字查詢論文的設計
2. 對於常見的知識庫類型，如私人知識庫、技術報告庫、期刊資料庫、全文資料庫等，沒有分析其使用者的行為模式，因此該雛型系統無法符合上述知識庫的流程
3. 分類法針對『技術論文』及『技術報告』進行分析，Survey 性質的論文(此類型論文針對特定問題蒐集他人所提出的解決策略，並分析其優缺點，這種論文對於軟體開發/研究等等亦有相當大的幫助)並未列入
4. 許多軟體知識庫之資料量龐大，龐大的軟體知識資料量需要分散式架構儲存，在其研究之雛型系統以單一主機實作，並無提出分散式架構及如何分散軟體知識
5. 雛型系統將許多索引資料儲存在記憶體中，然而在大量知識情況中，無法將所有索引儲存在記憶體之中
6. 知識庫需保留傳統常見的查詢方式，如查詢特定作者所發表的論文、依據期刊、Volume、Issue 來瀏覽特定 Issue 的論文等，此雛型系統並沒有整合這些功能。

本研究首先針對常見知識庫類型進行分析，發現使用者可分為四類：一般使用者、Submitter、Committer、知識管理者，一般使用者主要的行為是查詢知識，

Submitter 負責提交新的知識，Committer 負責將審核 Submitter 送過來的知識並加入知識庫中，知識管理者負責新增領域類別、議題等資料，並可執行搬移領域類別、分割領域類別等功能。

本研究接著針對 Suvery 類型的論文/報告進行分析，發現其關鍵字可分為『領域類別』、『Survey 議題』，Survey 議題為該論文所探討的主要議題，『領域類別』為該議題所屬領域，而 Survey 論文又可分為兩種：『針對已發展成熟領域的主題分析其各種解法之特色』，以及『針對尚未發展成熟的領域，提出其可能的發展方向』，『針對已發展成熟領域的主題分析其各種解法之特色』的 Survey 論文其 Survey 議題單一，並針對該議題列舉多種他人提出的解法，並進行比較，因此在 Survey 議題之下，有『解法』關鍵字。至於『針對尚未發展成熟的領域，提出其可能的發展方向』的 Survey 論文，由於尚未發展成熟，因此該議題可能包含的範圍較廣，因此議題下可能還會有子議題，而論文可能會針對子議題提出多個解法。

本研究接著針對各種類型知識庫，分析『使用單一伺服器』、『將索引資料全數載入至記憶體』、『缺少逆向查詢』、『缺少知識庫常用的方法』等缺點，並提出相關解決方法，進而提出這些類型的知識庫系統設計。

1.2節 各章節介紹

本論文章節安排如下，首先在第二章介紹各類型知識庫的性質及其常用的查詢方式，並針對該知識庫提出其缺點，接著介紹『問題-解法』軟體知識分類法及其雛型系統，並提出該分類法需改進之處及雛型系統之缺點。第三章介紹『問題-解法』知識庫系統的設計構想，從第二章所提出的缺點中，針對各類型知識庫提出解決方法。第四章則介紹技術報告庫的『問題-解法』知識庫系統之設計，包括使用案例(Use Case)、系統架構、模組架構、主要使用案例的流程等。第五

章則介紹私人知識庫、大型私人知識庫、期刊資料庫、全文資料庫之系統設計。第六章總結本論文之研究成果並提出未來可繼續發展的研究方向。

第2章 相關研究與背景知識

軟體產業是一高度知識密集的產業，軟體發展過程需不斷運用新知識來解決其問題，因此需要參考許多技術資料，這些技術資料除了來自手邊的書籍與論文外，還可透過電腦查詢知識庫，了解這些知識庫的查詢功能，有助軟體開發者快速和正確地找到想要的資料。

2.1節 知識庫類型

知識庫類型依發行者、經營者、使用者的不同約可分為：私人知識庫、期刊資料庫、技術報告庫、全文資料庫。私人知識庫為一機構為了儲存該機構所發展的技術文件以及其他相關知識供機構同仁分享而建立的知識庫。期刊資料庫則為期刊出版商/經營商為供認可的學術研究者與技術開發者查詢及使用期刊資料而建立的知識庫。技術報告庫為機構為了儲存其技術報告以供內部人員及特定公眾人員而建立的資料庫。全文資料庫則為期刊出版商/經銷商販售給一般單位並安裝在該單位以供其使用者查詢的技術資料庫。各類知識庫之性質分別說明如下：

2.1.1 私人知識庫

私人知識庫通常僅提供給機構內部使用，不對外開放。儲存文件型態包含論文、報告、書本，主要來源有二：機構內部自行產生的技術資料及由外部單位經合法蒐集而來的技術資料，這些技術資料均有助於該單位的營運，故稱為知識。機構內部自行產生的知識可透過其內部新增技術文件的程序新增至知識庫中，至於其他知識庫知識，則為由機構向期刊出版商/經營商購買全文資料庫安裝在其內部使用，或者向論文出版商或經營商購買期刊資料庫使用權，內部員工透過代理伺服器向期刊資料庫取得所需的技術論文，再經由內部新增技術文件的程序將

知識新增到私人知識庫中；另外，也可由員工從開放的技術報告庫取得技術報告後新增到知識庫中。私人知識庫的領域範圍與該機構的營業項目的多寡有密切關係，一般機構的知識庫領域範圍並不多，但跨國際的大型機構(例如微軟)其營運包含的知識領域多且使用者眾，故其私人知識庫所包含的領域較多。

私人知識庫的建構方法通常依照分類來建構，圖 2-1 為私人知識庫新增知識的流程，作者(Author，機構內部文件的作者)或提交人(Submitter，機構員工從外面知識庫取得知識文件後，新增至私人知識庫中)透過內部新增知識介面將文件上傳到伺服器中，並且提供了該文件的分類、摘要、標題、關鍵字等等資料，接著 Committer 檢查該文件的分類、關鍵字等等資料是否符合，如果合格的話，則將文件加入私人知識庫中。圖 2-2 為私人知識庫的分類架構範例，當 Committer 將文件提交到知識庫之後，知識庫便會依照文件所描述的分類，將該文件歸類到該分類下。而通常私人知識庫會提供關鍵字查詢、作者名稱查詢等等查詢方法，因此需要建立相關資料的索引，圖 2-2 中的 Keyword Index 就是用來查詢知識中的關鍵字有出現某個字的索引，而 Author Index 則為用作者名稱來查詢知識，若需要用論文標題、論文摘要中出現的字來查詢，則需要建立相關的索引。

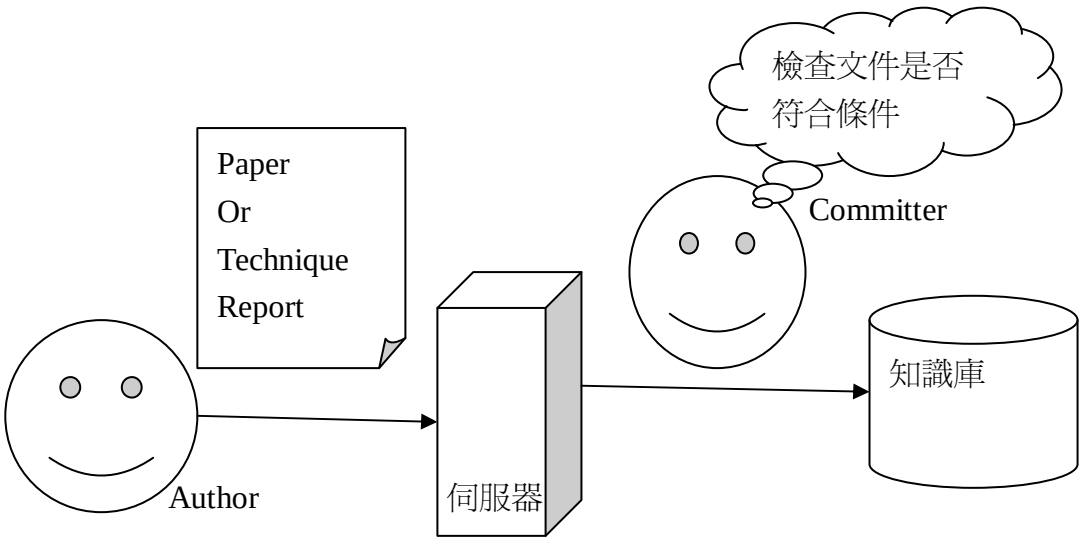


圖 2-1 私人知識庫新增知識流程

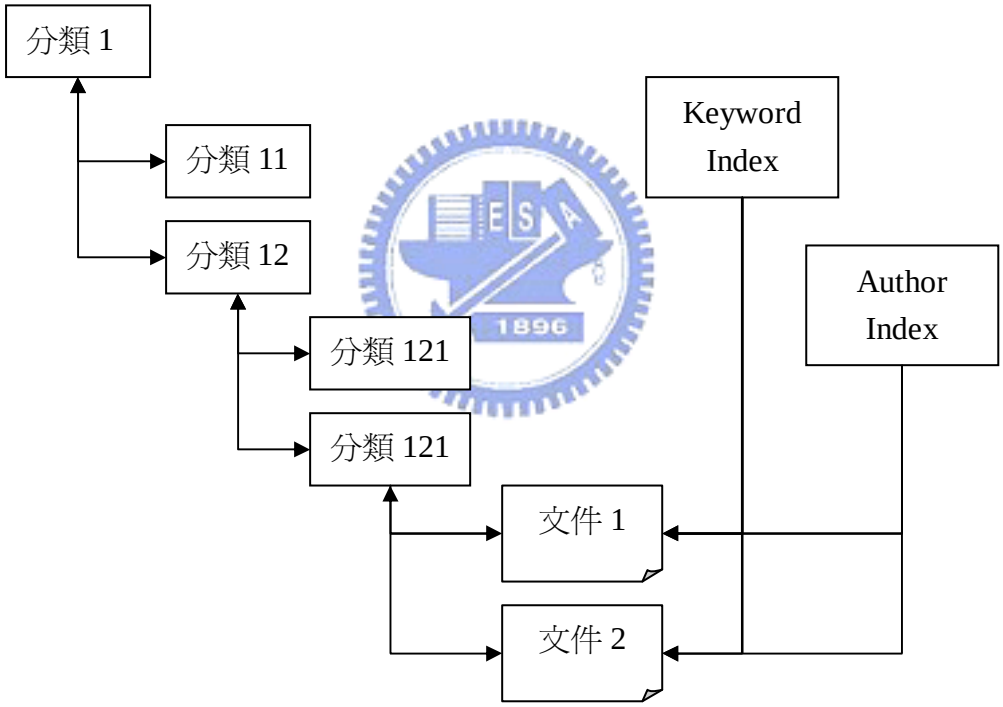


圖 2-2 私人知識庫的分類架構範例

2.1.2 期刊資料庫

期刊資料庫的建構者有兩種：出版商、經營商。如 ACM[1]、IEEE[2]等學術組織自行為獨立發行的技術期刊建立資料庫並供會員使用，故這些組織為出版商，也為經營商；另有一些出版商將其出版的論文期刊之電子檔，透過其他資料庫經營商提供給經認可的會員使用，如 JSTOR[3]、MUSE[4]即為經營商匯集不同

出版商的期刊資料庫。期刊資料庫所包含的領域很廣，知識資料量龐大，不限制其使用對象，一般使用者皆可免費查詢，但要閱讀論文全文時，使用者需向其出版商或經銷商購買使用權方可閱讀全文。

期刊資料庫提供的查詢方式有『從期刊名稱、Volume、Issue、論文列表檢索論文』、『從使用者輸入關鍵字與論文的論文標題、關鍵字及/或摘要比對檢索論文』、『以分類(Classification)查詢』等三種方式，分別介紹如下：

1. 從期刊名稱、Volume、Issue、論文列表檢索論文

讀者先選擇期刊名稱，然後選擇 Volume，期刊庫列出該 Volume 所有 Issue，使用者點選 Issue 後，期刊庫列出該 Issue 的所有論文，供讀者選擇所要閱讀的論文。舉 ACM Digital Library 為例，ACM Digital Library 會先列出有哪些期刊：

Browse the ACM Journals:

- ACM Computing Surveys (CSUR)
- ACM Journal of Computer Documentation (JCD)
- ACM Journal on Emerging Technologies in Computing Systems (JETC)
- Journal of Experimental Algorithmics (JEA)
- Journal of the ACM (JACM)
- Journal on Educational Resources in Computing (JERIC)

點選 JACM 之後，出現最新的 Issue，也有一個連結供讀者選擇其他 Issue，點選『所有 Issue』連結之後出現下列 Issue 列表：

點選 Volume52 的 Issue6 之後，系統列出該 Issue 的所有論文主題與作者列表供讀者選擇閱讀，如下圖所示：

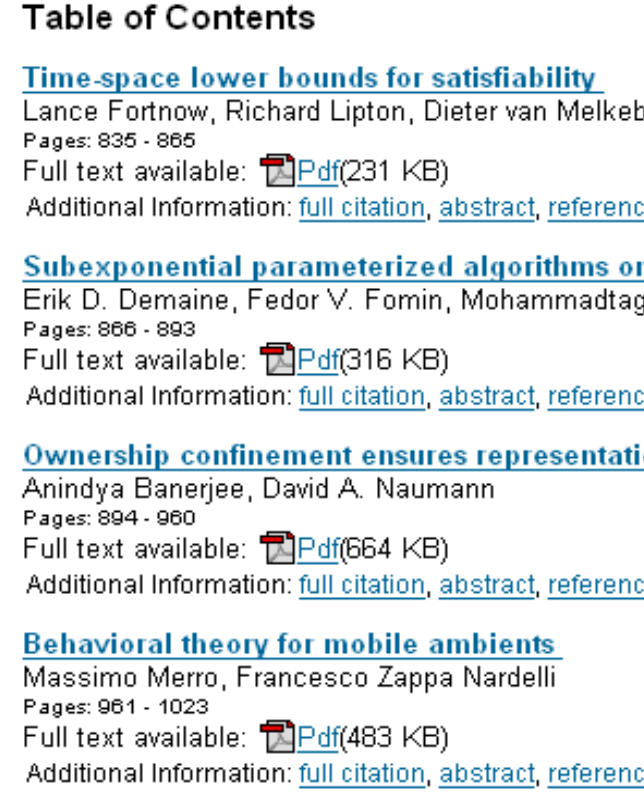


Table of Contents

Time-space lower bounds for satisfiability
Lance Fortnow, Richard Lipton, Dieter van Melkebeek, Anastasios Viglas
Pages: 835 - 865
Full text available: Pdf(231 KB)
Additional Information: full citation, abstract, references, index terms

Subexponential parameterized algorithms on bounded-genus graphs
Erik D. Demaine, Fedor V. Fomin, Mohammadtaghi Hajiaghayi, Dimitrios M.
Pages: 866 - 893
Full text available: Pdf(316 KB)
Additional Information: full citation, abstract, references, index terms

Ownership confinement ensures representation independence for obj
Anindya Banerjee, David A. Naumann
Pages: 894 - 960
Full text available: Pdf(664 KB)
Additional Information: full citation, abstract, references, index terms

Behavioral theory for mobile ambients
Massimo Merro, Francesco Zappa Nardelli
Pages: 961 - 1023
Full text available: Pdf(483 KB)
Additional Information: full citation, abstract, references, index terms

每個期刊所刊登的文章都有其特定領域，如 IEEE Transactions on

Software Engineering 所發表的論文均屬 Software Engineering 領域，因此有些期刊庫(如 Springer Link[5])會將其所有期刊先依其領域作分類，選擇領域類別，期刊庫列出此領域內之期刊供讀者選擇。

2. 從使用者輸入關鍵字與論文的論文標題、關鍵字及/或摘要比對檢索論文

許多論文標題只能反應其要解決的問題及採用的解法，使用者需閱讀論文內容後才能判斷是否符合需求，為反應文章的性質，期刊庫要求作者對其發表的論文加上『關鍵字』，以加強論文意義的描述。使用者輸入欲查詢關鍵字，資料庫系統檢查論文是否有這些關鍵字，若有，即將其列出。此外，也可以作者名稱(Author)、編輯者名稱(Editor)、檢閱者名稱(Reviewer)當關鍵字來找尋符合該條件的論文。

3. 以分類(Classification)查詢

ACM 期刊庫將其論文依領域作分類，以四層的樹狀架構表示，第一層分類代碼由 A-Z，第二、三層分類代碼使用數字，第四層則沒有分類代碼，直接使用名稱，圖 2-3 為 ACM 分類的範例，此種方式對搜尋特定領域文章較為方便。

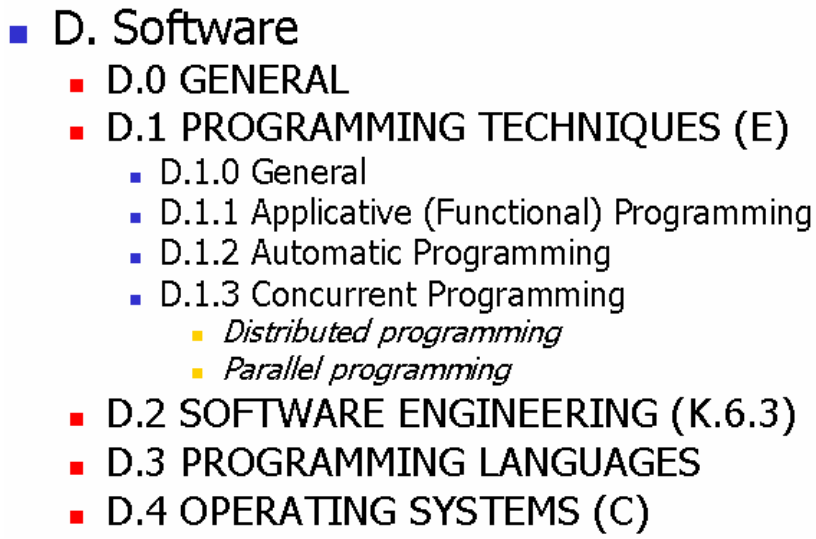


圖 2-3 ACM 分類架構範例

論文作者要在 ACM 期刊發表論文時，需先設定該論文的主要分類(Primary Classification)，然後依照論文性質，設定多個額外分類(Additional Classification)。當使用者要找尋該分類下的論文時，輸入該代碼(如 D.1.3)，系統就會依照代碼列出該分類下的論文。使用者也可以輸入多個分類代碼，以縮小查詢範圍。

目前期刊庫的建構方法為利用期刊、Volume、Issue 來建構。圖 2-4 為期刊資料庫的新增知識流程，論文作者將論文上傳到伺服器，並說明該論文的標題、摘要、關鍵字等等，然後檢閱者(Reviewer)開始看該篇論文是否夠新穎、提出的方法是否有效等等，當檢閱者接受作者的論文，伺服器便通知編輯者(Editor)，編輯者將所接受的技術論文編輯成冊，指定論文所屬的 Volume、Issue、以及頁數，把論文的電子檔放入資料庫中。如圖 2-5 所示，當編輯者將論文放進資料庫之後，資料庫便將該論文放在某期刊的 Volume X, Issue Y 中，並建立其相對應的查詢索引(如作者、關鍵字、ACM 分類架構等等)。

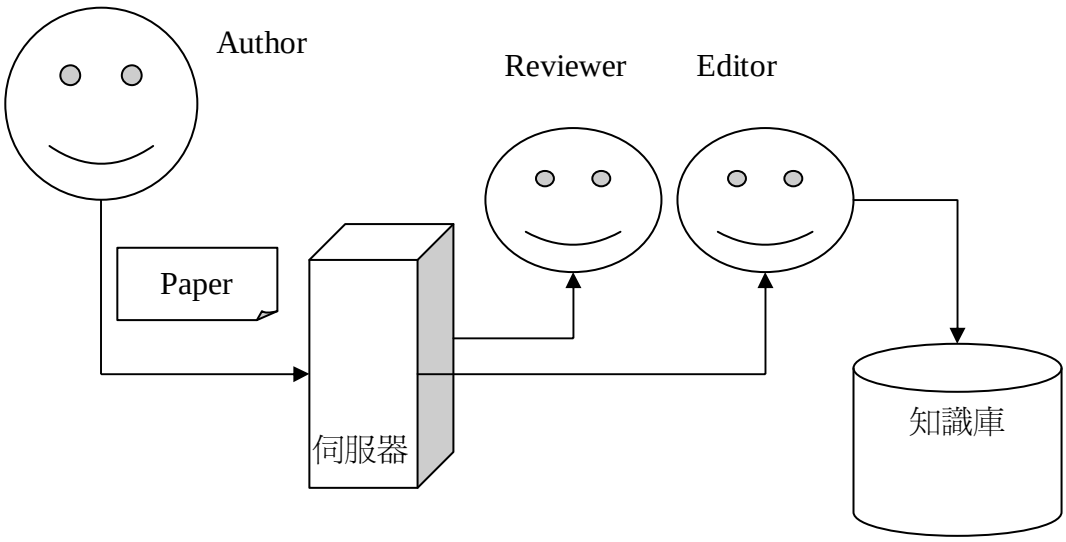


圖 2-4 期刊資料庫新增知識流程圖

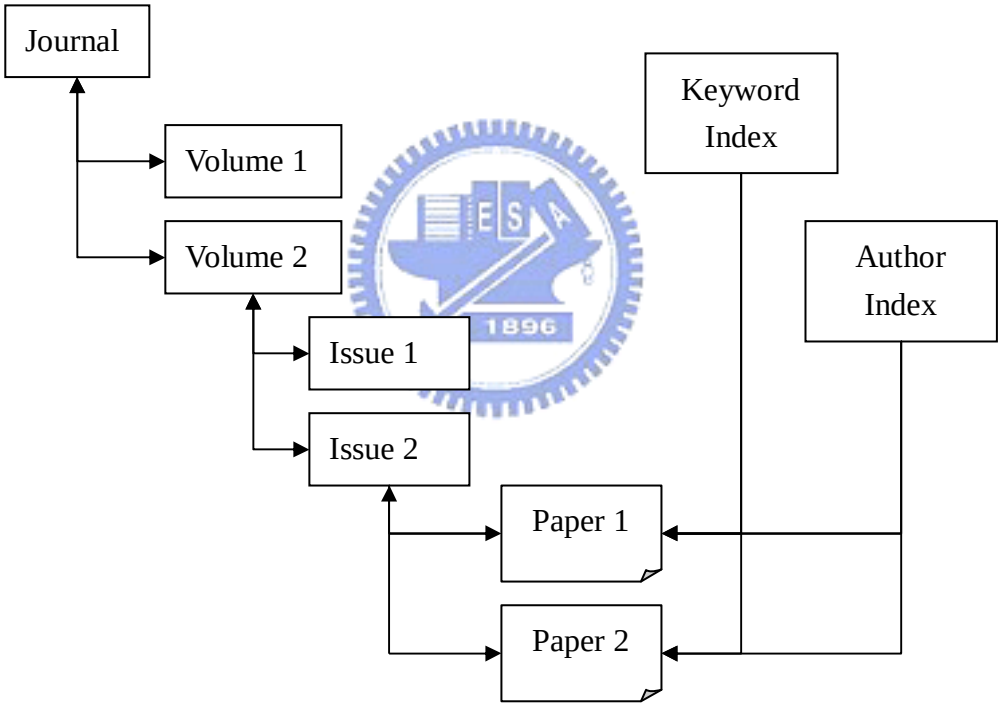


圖 2-5 期刊資料庫分類架構範例

2.1.3 技術報告庫

技術報告庫為技術單位建立自己的技術文件資料庫，技術單位包含很廣，例如 NASA[7]、交大資工系等等都可建立自己的技術報告庫，技術報告庫的使用對象，除了自己單位內部使用者之外，也開放讓外面使用者查詢，如 NASA 開放其技術報告庫供外面使用者查詢。

技術報告庫提供的查詢功能，除了輸入關鍵字查詢，從技術報告的標題、作者名稱、摘要、關鍵字等找尋符合條件的技術報告外，亦提供從年代開始查詢，使用者點選年代後，報告庫列出該年代的技術報告，供使用者選取而閱讀該技術報告。

圖 2-6 為技術報告庫新增知識流程，作者將技術報告上傳至伺服器中，並說明該技術報告的標題、摘要、關鍵字等等，接著 Committer 檢查該技術報告的關鍵字是否符合，經認可後將該技術報告提交(Commit)至報告庫中。當報告庫收到報告時，如圖 2-7 所示，會將此報告編排在該年度底下，並且會為其建立相關查詢用的索引(如作者、關鍵字等等)。

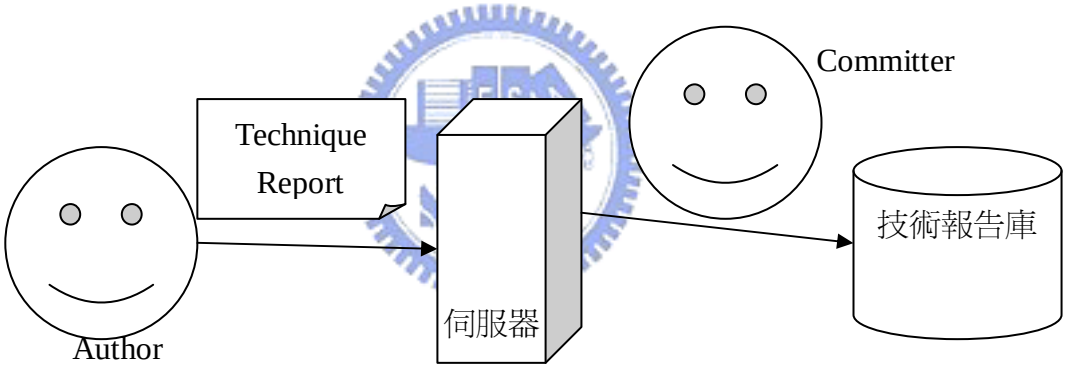


圖 2-6 技術報告庫新增知識流程

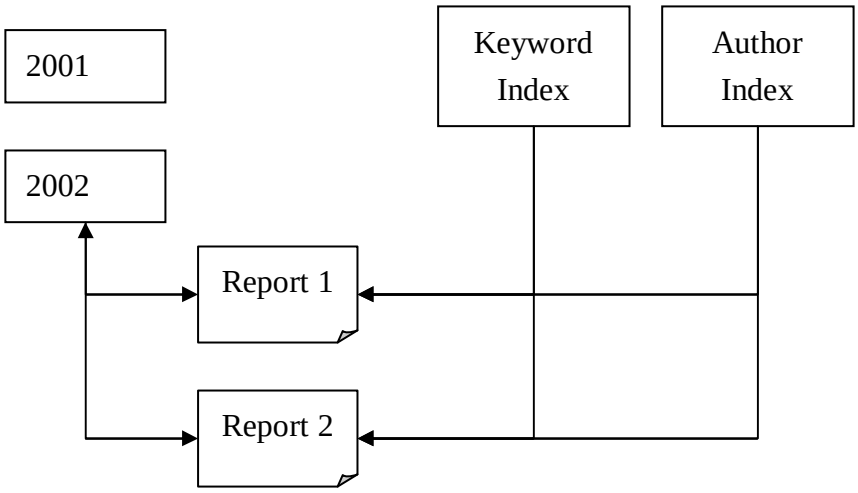


圖 2-7 技術報告庫分類架構範例

2.1.4 全文資料庫

全文資料庫如同期刊資料庫一般，可提供期刊資料庫的查詢功能，不同的地方在於更新知識庫的方式與使用者對象。全文資料庫的更新是由販售全文資料庫的出版商或經銷商負責，出版商或經營商會每隔一陣子將全文資料庫的資料更新。全文資料庫的取得為單位(如交大圖書館)向某個出版商或經銷商購買全文資料庫，然後由廠商為該單位安裝全文資料庫，並每隔一段時間更新全文資料庫，直到購買合約到期為止。全文資料庫的使用對象通常為該單位的使用者(如交大圖書館的話，使用對象為交大師生以及其他有合作關係的學校師生)。

2.1.5 目前知識庫的查詢方法之缺點

以上分析了現有知識庫的類型，然而目前知識庫所提供的查詢方法，有以下兩個缺點：

1. 現有知識庫的關鍵字沒有分類(如著名的 IEEE 期刊也只規定關鍵字定義的基本原則，只提供關鍵字列表供作者從中選擇，並建議作者採用五到八個關鍵字作為索引，若有新的關鍵字則提供給 IEEE 參考)，一個關鍵字有可能代表此論文所屬的領域、要解決的問題、採取的解決策略、使用的技術、相關研究等等，導致當使用者想要查詢『特定主題有哪些解法』時，使用者輸入該議題的關鍵字，找出來的論文，卻不一定是解決該議題的論文，有可能是採取該解決策略的論文、或者只是相關研究而已。
2. ACM 的分類層數不足，當使用者輸入關鍵字時，所得的論文有許多並不符合需求，需一一閱讀這些論文之後，才了解哪些解法對其有助益；且查詢所得論文數量可能很多，閱讀費時，對研究者相當不便

為了解決以上問題，本實驗室謝祖望學長提出『問題-解法』軟體技術知識分類法[6]，2.2 節將介紹此知識分類法。

2.2節『問題-解法』知識分類法

謝祖望學長的研究中指出，技術論文、技術報告、書本知識都可視為『問題導向』的知識，大多數技術論文是針對某個領域的某特定主題提出解決策略或方法，技術報告是針對特定主題作探討的研究成果。書本知識探討的內容側重介紹及教學的功能，這對想要了解或學習特定主題的軟體工程師而言特別有幫助，一般人對一個不熟悉的領域通常都是透過書本知識來學習的。以下將介紹技術論文、技術報告、書本知識的本質，並藉由這些本質，歸納其關鍵字架構，最後描述謝祖望學長在其研究中所實作的雛型系統。

2.2.1 技術論文與技術報告本質

技術論文與技術報告的關鍵字可分為四類，即『領域』、『問題』、『解法』、『技術』。一篇論文可用欲解決的『問題』(Problem)、此問題所屬『領域』(Domain)、解決策略(即解法)及採用的『技術』來定義它。每一領域下有許多問題待解決，描述一個解法的重點在於『解決策略』(Approach)以及運用的『技術』(Technique)。圖 2-8 為技術論文/報告關鍵字示意圖，領域內有許多問題(Problem)，每一個問題有一至多個不同的解法(Approach)，而每一個解法(Approach)將使用(Use)一些技術(Technique)，舉例來說，Steffen Heinz 的”Burst tries: a fast, efficient data structure for string keys”論文定義了 Binary trees、splay trees、string data structures、text databases、tries、vocabulary accumulation 等關鍵字，其內容是處理 text database 領域中的 string data structure 問題，他提出了稱為 burst tries 的解法，此解法運用了 splay trees、tries、binary trees、hash table 等技術，圖 2-9 為此論文的關鍵字關係示意圖。

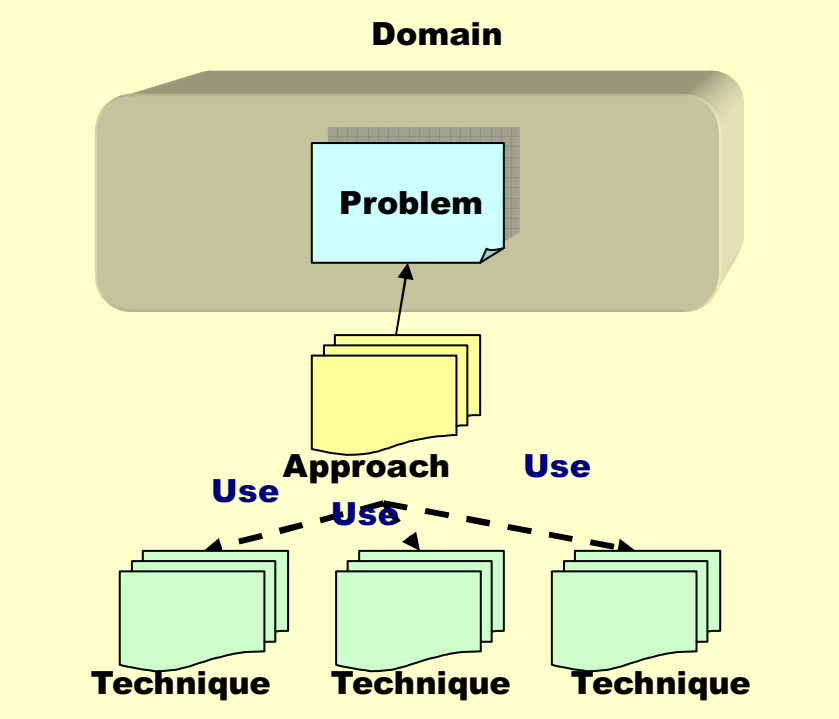


圖 2-8 技術論文/報告關鍵字示意圖

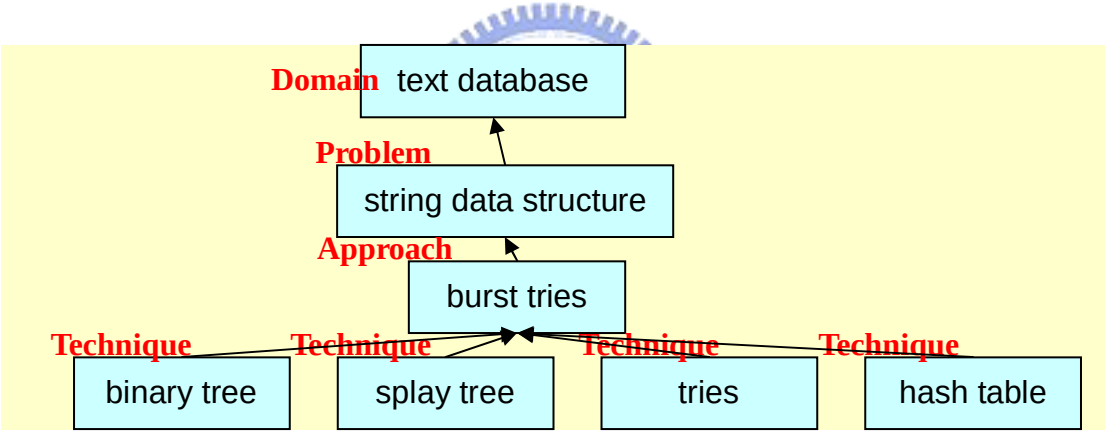


圖 2-9 技術論文/技術報告範例

2.2.2 書本知識本質

書本知識的關鍵字分為三類，即「主題」、「議題」、「解法」：

1. 主題（Subject）：主題即是一本書所探討的主要內容，通常會是書名，可藉由一良好的分類架構將軟體技術知識的主題分類，使其定義標準化。

2. 議題（Topic）：一本書的主題下會包含幾個主要的議題。這些議題下可能又可細分為子議題，由於這些子議題的內容組成主要議題，故亦可看

成主要議題包含數個子議題。書本知識的議題間可能會有閱讀上的順序關係，例如閱讀 Object-oriented analysis 跟 Object-oriented design 之前應先閱讀 Object-oriented concept，因此議題之間會有依存的關係。

3. 解法（Approach）：某些書本在介紹議題時，並不是介紹其子議題，而是介紹該議題的有哪些不同的解法或方法，這類型便以「解法」定義。

書本知識關鍵字可以圖 2-10 表示，以 Roger Pressman 的 Software Engineering 一書中針對 Project Management 這個議題的內容以上述的方式分類，便可得到如圖 2-11 的結果

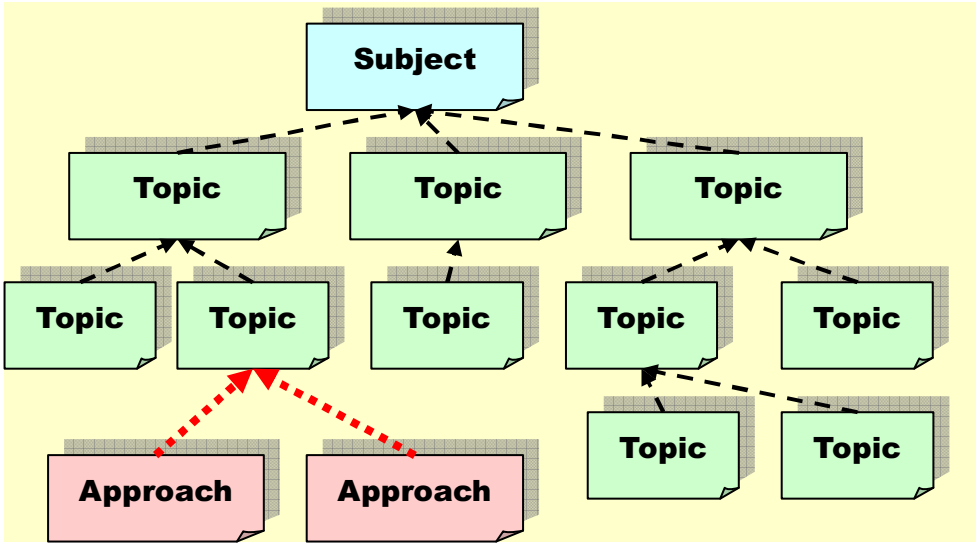


圖 2-10 書本知識關鍵字關係

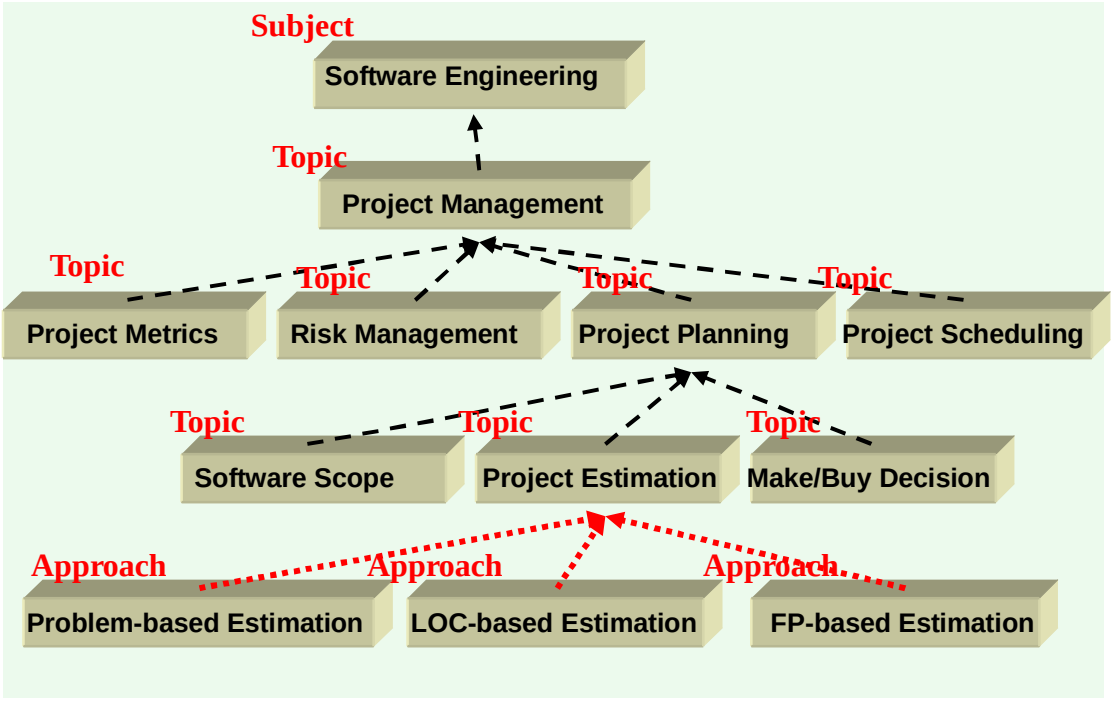


圖 2-11 書本關鍵字關係範例

雖然書本知識與技術論文、技術報告的「問題導向」特性有些不同，但仍可將書本知識視為具有「問題」與「解答」特性的知識。對書本知識而言，使用者的問題通常就是：特定主題應看哪本書？特定議題的介紹？特定議題有哪些方法？而針對這些問題的解答就是書本的內容，透過「主題」、「議題」、「解法」三類關鍵字並配合階層架構，使用者能夠知道特定主題有哪些書，這些書提到了哪些議題，特定議題下又介紹了哪些子議題，或介紹了哪些方法，因此，「主題」、「議題」可視為問題，而「子議題」、「解法」便是解答。透過上述的表示方法，可為書本知識定義完整的關鍵字做為描述及索引，也能夠讓使用者快速掌握書本知識的內容，並可對特定主題、特定議題的書本知識進行概略的比較，增進書本知識的效率。

2.2.3 『問題-解法』關鍵字架構

技術論文及技術報告的關鍵字分為四類，即「領域」、「問題」、「解法」、「技術」。這四個分類正好符合「問題－解法」架構，因為要描述一個問題，應當先描述其所在領域，軟體知識的領域是有階層性的，一個大領域可分為數個小領

域，因此描述知識的領域最恰當的方式就是利用階層架構（Hierarchy-structure），故我們可以稱領域為知識類別（Category），有了領域之後，我們便可以在階層架構上加入對問題的描述。這些問題便可稱為該領域下的「議題」（Issue）；而描述一個解法，重點就在於「解決方法」（Approach）以及運用的「技術」（Technique），因此，我們只需利用有良好定義的關鍵字，即可對技術論文及技術報告進行有效率的「問題－解法」分類與描述。

書本知識的關鍵字分為主題（Subject）、議題（Topic）、解法（Approach）三類，這個分類能夠與上述的分類整合。因為書本知識所探討的主題其實與上述的知識類別（Category）是很接近的。我們利用階層架構將知識領域做分類，這些分類其實都可做為書本知識的主題，例如分類上有 Software Engineering 的領域類別，也有書本知識的主題為 Software Engineering，如上所述，我們可以藉由一良好的分類架構將軟體技術知識的主題分類，使書本知識的主題定義標準化，因此，只要這個分類架構具有足夠的深度與內容，便可做為書本知識的主題。而書本知識下面的議題（Topic），雖然與技術論文的議題（Issue）意義不同，但亦可將 Topic 視為 Issue，只是書本知識的 Issue 是有主要的 Issue，底下可再細分為 sub-Issue 或 sub-sub-Issue，而技術論文的 Issue 則是知識類別下的問題，其性質不同，但都可以用議題（Issue）來代表。由於書本知識與技術論文本質上就不同，因此也不會造成定義模糊的問題。至於書本知識的解法（Approach），與技術論文的解法也有些許差異，書本知識通常是針對一個議題介紹幾種解法，而技術論文則是針對一個問題提出其解法，但在本質上仍是問題與解法的關係，故可與技術論文的分類整合。

由上述可知，領域類別是『問題-解法』知識分類法中的基本架構，由於領域類別太多，因此可利用階層特性，採大領域細分為小領域方式，採用樹狀架構

來定義每一個領域類別。議題與解法建構在領域類別之下，因此分類機制需有足夠的廣度與深度，方能讓使用者有效率地找到合適的知識。目前常用的分類法都不超過七層架構，且心理學家 George Miller[10]於 1956 年提出人的暫時記憶數量是七加減二，因此謝祖望學長提出如圖 2-12 的七層領域架構圖，而在此七層領域架構下，包含了技術論文/報告的『議題』(Issue)、『解法』、『技術』等關鍵字，也包含書本知識的『議題』(Book Issue)、『子議題』(Sub Issue)、『解法』等關鍵字。

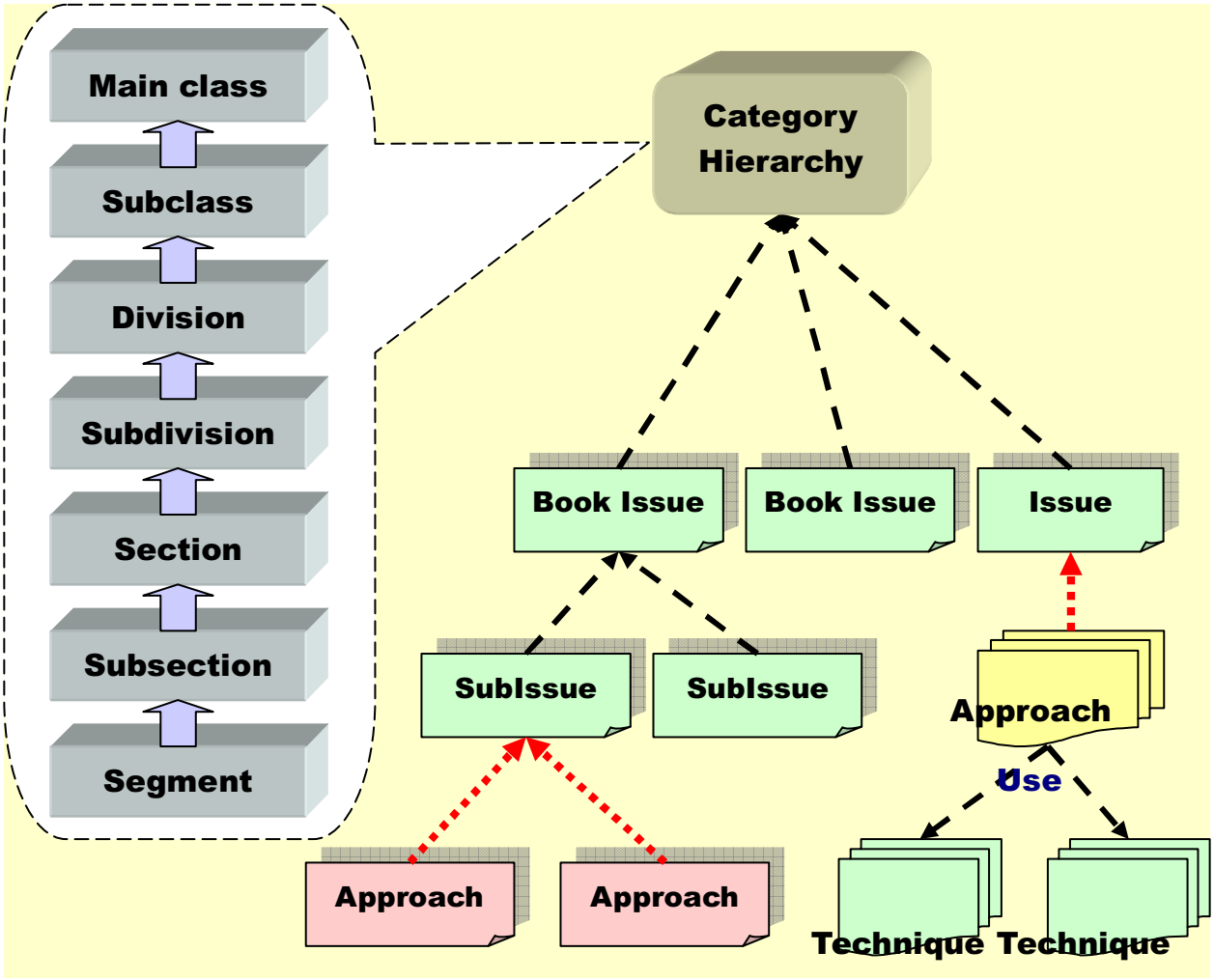


圖 2-12 『問題-解法』知識庫關鍵字架構

2.2.4『問題-解法』知識庫雛型系統

為了證明以上『問題-解法』軟體知識分類法的可能性，謝祖望學長實作了

此知識庫雛型系統，其主要提供使用者進行知識的搜尋、檢索以及提供知識工程師對知識內容及分類架構的維護。依一般使用者與知識工程師的不同，可整理其需求如下：

1. 一般使用者需求：一般使用者主要利用分類架構由上往下找尋知識，其中包括找領域類別介紹、議題介紹、議題解法等目的，使用者也可以找尋特定技術被運用在哪些論文或領域。此外可利用符號表示法新增知識。其需求可整理如下表：

一般使用者需求

編號	需求	說明
R1	新增知識	利用符號表示法新增知識
R2	類別瀏覽	從分類架構中瀏覽知識類別，找到類別後可閱讀其子議題、書本知識、最新狀態等
R3	讀取議題	特定議題介紹、書本知識、最新狀態等
R4	讀取解法	讀取解法介紹及相關論文等
R5	閱讀技術論文或技術報告	閱讀其摘要、引用、運用技術等
R6	閱讀書本知識	讀取書本知識議題架構及依存關係
R7	技術運用狀況	了解技術運用論文、解法、議題、領域
R8	最新知識庫文件	取得知識庫最新狀態

2. 知識工程師需求：知識工程師主要工作是維護知識分類架構，其需求可整理如下表：

知識工程師需求

編號	需求	說明
R9	調整知識類別	知識類別的搬移、分割、整合，以及設定知識類別的關聯性
R10	管理議題	議題的新增、搬移、修改等
R11	管理解法	解法的說明以及繼承關係等
R12	管理技術	技術的新增及其相關或相等關鍵字

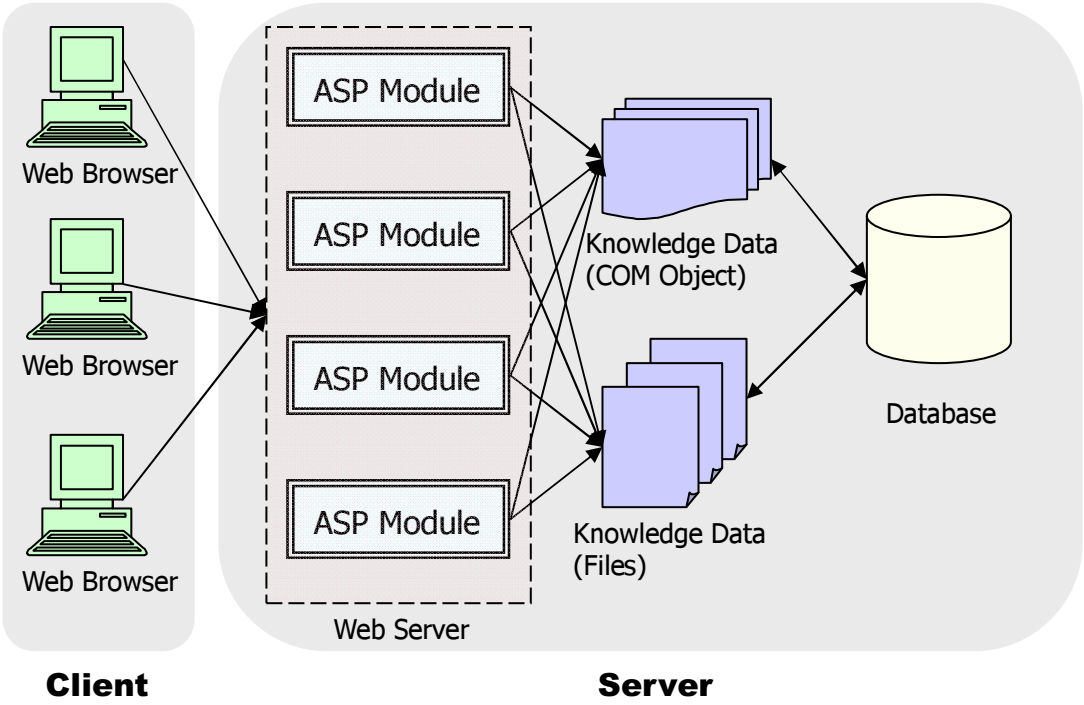
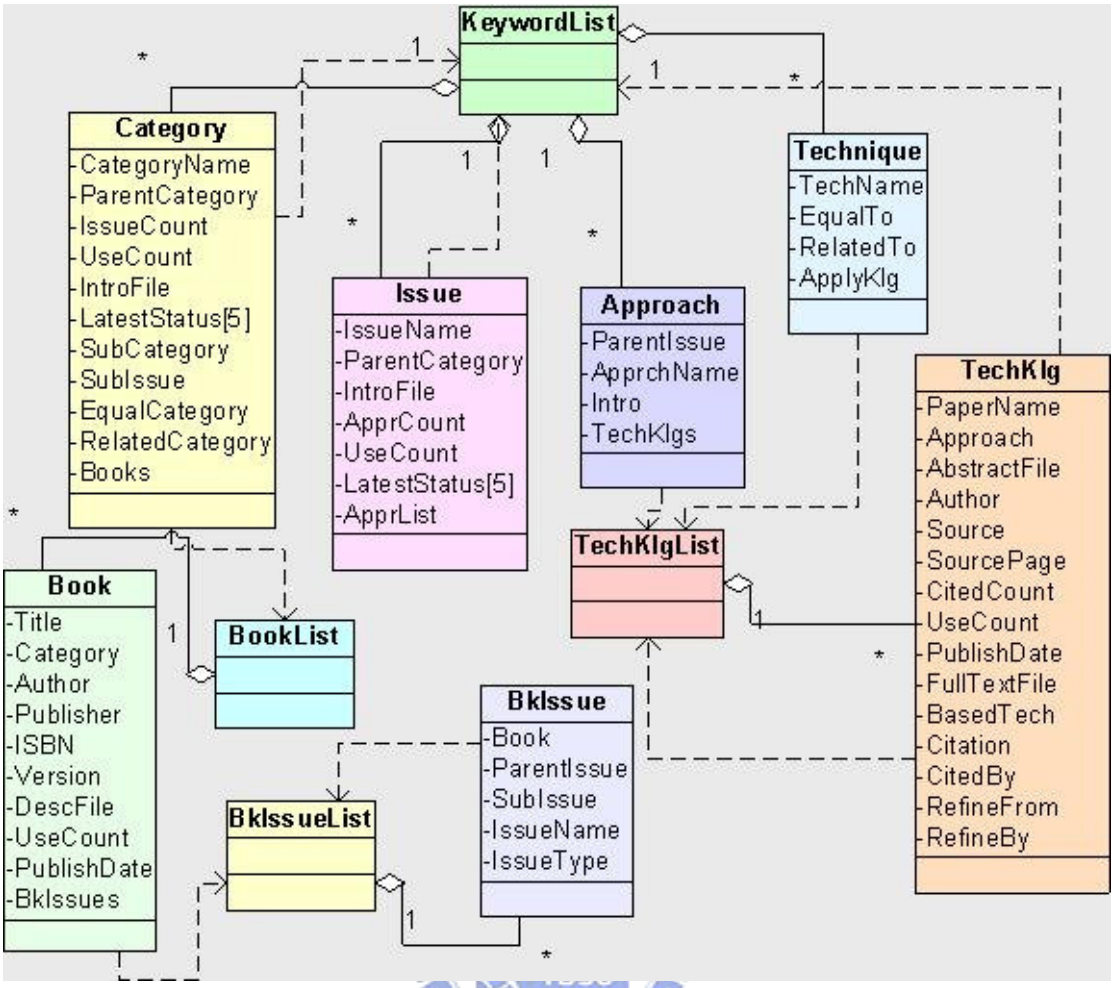


圖 2-13 知識庫雛型系統架構圖

圖 2-13 為『問題-解法』軟體技術知識庫系統，此系統可供使用者瀏覽、檢索、新增知識，此系統以 IIS（Internet Information Server）搭配 ASP（Active Server Page）做為網站架設及系統開發平台，後端採用 SQL Server 資料庫存放資料。由於直接存取資料庫較費時，此系統將關鍵字索引資料(如領域類別的名稱、子領域類別列表、議題名稱、議題的解法列表等等)在系統初始化時從資料庫讀出至系統記憶體中，這部份採用 COM（Common Object Model）物件來完成，另外較不常存取的資料則先自資料庫取出存至檔案系統中，以加速運作效率。快取資料的類別關係圖如下：



快取資料類別關係圖

2.3節『問題-解法』知識庫雛型系統之缺點

謝祖望學長之『問題-解法』技術知識庫為雛型系統，可應付一般公司的私人知識庫情況，但將此雛型系統應用在較大型的知識庫時，將會面臨以下六個問題：

1. 該系統提供的查詢功能，都是由上到下的查詢(亦即由大範圍到小範圍的查詢，如領域類別->子領域類別、領域類別->議題、領域類別->書本、領域類別->書本議題)，沒有提供如『以書本議題找尋書本知識』以及『以論文/報告採用的解法技術找尋論文/報告』的逆向查詢功能。少了這些逆向查詢的功能，會減低使用者使用『問題-解法』知識庫的方便

性。

2. 此雛型系統並未考量使用者分類，例如一般使用者即可新增知識，但四種類型知識庫都需要經過認可後，才可將知識新增到資料庫中，若不依循此四種類型知識庫的新增知識流程，『問題-解法』知識庫將無法適用於此四種類型知識庫中。

3. 原『問題-解法』分類法中說明技術論文的關鍵字包含『領域類別』、『議題』、『解法』、『技術』，而除了技術論文以外，尚有 Survey 性質的論文，此類型的論文特性為有問題領域、問題以及很多針對此問題所提出的解決策略，但是這些解決策略都是經由他人提出，該論文中並無提出任何新的解法，如 Fabio Kon 的”Distributed File Systems Past, Present, and Future – A Distributed File System for 2006”[11]僅提出現有分散式檔案系統的優缺點及未來的可能發展方向，並無提出新的解法、技術。Survey 性質的論文，僅有領域、議題、相關解法等關鍵字，此與『問題-解法』知識分類法中的技術論文/報告不同，因此無法適用於該關鍵字架構，然而 Survey 性質的論文卻是初學者學習該領域的良好來源，因此應該將此種論文納入知識庫中。

4. 軟體技術知識隨時間增加，知識量越來越大，單一伺服器無法儲存，而原雛型系統採用單一伺服器的架構，沒有提出分散式系統的架構，也沒有提出如何分配資料，因此無法負荷這些知識資料的儲存以及大量使用者的查詢要求，導致系統無法發揮其功能。

5. 誠如 2.2.4 小節談到，此雛型系統將知識庫關鍵字索引資料(如領域類別的名稱、子領域類別列表、議題名稱、議題的解法列表等等)在系統初始化時從資料庫讀出至系統記憶體中，但知識數量龐大時，這些索引資料將耗費許多記憶體，甚至無法全部存於記憶體，而導致查詢效能低落。

6. 在期刊類型知識庫中，有許多使用者仍然習慣原有期刊名稱、Volume、

Issue 以及輸入作者名稱等查詢方式，但此雛型系統沒有提供這些查詢方法，少了這些查詢方法，將導致使用者的不便，例如許多資深的專家，通常會想要了解最新的期刊發表了哪些新文章，如果不提供依照期刊名稱、Volume、Issue 的查詢方式，將無法滿足資深專家的需求，如此一來，『問題-解法』知識庫的功能性便大大降低。

以上六個問題，除了『沒有提供逆向查詢功能』、『使用者分類不清』這兩個問題為雛型系統應用在四種知識庫類型都會面臨的問題外，其他四個問題都會依照知識庫類型以及規模而有所差別。以下將針對『一般私人知識庫』、『大型私人知識庫』、『技術報告庫』、『期刊資料庫』（全文資料庫僅新增知識流程與期刊資料庫不同，其餘皆相同，因此省略)說明：

1. 一般私人知識庫：一般公司的私人知識庫的知識量不多，因此其知識索引資料可能可以完全載入記憶體，而且其知識資料也可以完全放在同一台伺服器中。而缺少常用的查詢如『期刊名稱、Volume、Issue』這些查詢方式對私人知識庫而言並不影響，因為私人知識庫本身就沒有處理期刊類型的查詢，然而缺少 Survey 性質的論文，可能會導致員工無法取得一般性的介紹資料，增加其學習曲線。
2. 大型私人知識庫：大型私人知識庫由於資料量龐大，因此無法將其知識的索引資料完全載入記憶體中，而且也無法將所有的知識資料放在同一台伺服器中。除了資料量龐大所引起的缺點外，其餘缺點與一般私人知識庫相同。
3. 技術報告庫：技術報告庫的資料量通常不會很龐大，因此除了少數大型技術報告庫以外，其知識索引資料可以完全載入記憶體中，知識資料也可以放在同一台伺服器中。缺少『期刊名稱、Volume、Issue』的查詢方法對技術報告庫不影響，然而缺少以年代找尋技術報告可能會影響到使用者，因此對於技術報告庫，需要增加以年代查詢技術報告的功能。

4. 期刊資料庫：期刊資料庫的知識量通常比較龐大，因此與大型私人知識庫一樣，無法將知識索引資料完全載入記憶體中，也無法將所有知識資料放在同一台伺服器中。而缺少『期刊名稱、Volume、Issue』這些查詢方式，將會造成期刊資料庫的使用者使用上的不便，降低期刊資料庫的功能性;缺少 Survey 性質的論文分類,無法讓使用者快速找到 Survey 性質的論文，亦會降低期刊資料庫的功能性。

綜合上述，可知雛型系統無法運用在目前常見的四種知識庫類型，而且各類型知識庫都會面臨不同的問題，因此需要針對此四種知識庫個別設計其『問題-解法』知識庫系統。本論文實作之『問題-解法』軟體技術知識庫希望能與以往的知識庫查詢方法互相輔助，用以往的查詢方法所得到的知識，希望能夠在該知識中顯示『問題-解法』的關鍵字，當使用者想要進一步閱讀相關知識時，可透過這些關鍵字來瀏覽『問題-解法』系統的知識，以發揮軟體知識的最佳效用。

第3章『問題-解法』知識庫設計構想

如第二章所述，謝祖望學長所提之『問題-解法』知識庫應用在不同性質之知識庫，有下六個不足之處：『缺少逆向查詢功能』、『使用者分類不清』、『缺少 Survey 性質的論文』、『使用單一伺服器，無法處理大量知識資料以及大量查詢』、『將索引資料全部載入至記憶體，無法適用大型知識庫』、『缺少知識庫常用的查詢方法』，其中以『使用者角色分類不清』、『缺少逆向查詢』及『無法應用在大量知識資料的情況』之問題影響最大。使用者角色分類不清，無法與新增知識片段結合，有必要先對四種知識庫的使用者角色作定義。缺少逆向查詢功能造成使用者找尋知識範圍縮小，無法發揮『問題-解法』知識庫的效能。缺乏處理大量知識資料之能力造成雛型系統無法應用在大型私人知識庫及期刊資料庫，降低其實用性，本章將對這些問題提出改善之道。

本章首先在 3.1 節介紹不同知識庫的使用者及其應有之功能需求，3.2 節討論 Survey 類型論文之分類及儲存機制，3.3 節探討其餘四個問題對各類型知識庫的影響，進而說明解決之道。

3.1節 使用者角色分類

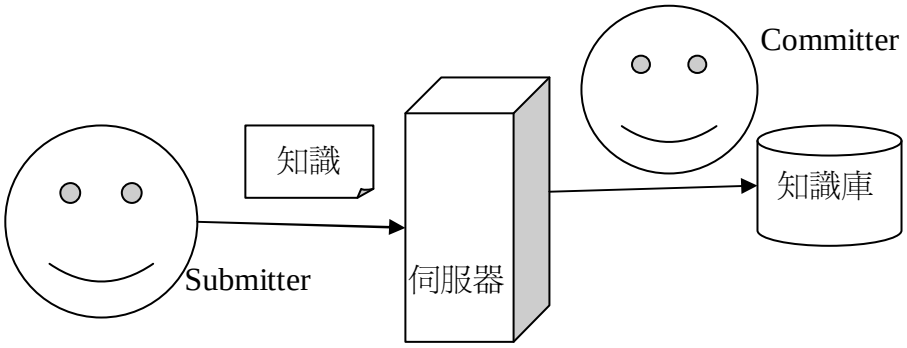


圖 3-1 私人知識庫、技術報告庫新增知識流程

圖 3-1 為私人知識庫、技術報告庫的新增流程示意圖，Submitter 將新增知

識文件及其關鍵字(領域類別、議題、解法、技術、書本議題等)、作者、標題等資料上傳到伺服器中，Committer 檢查關鍵字是否符合格式，若正確則存入知識庫中。

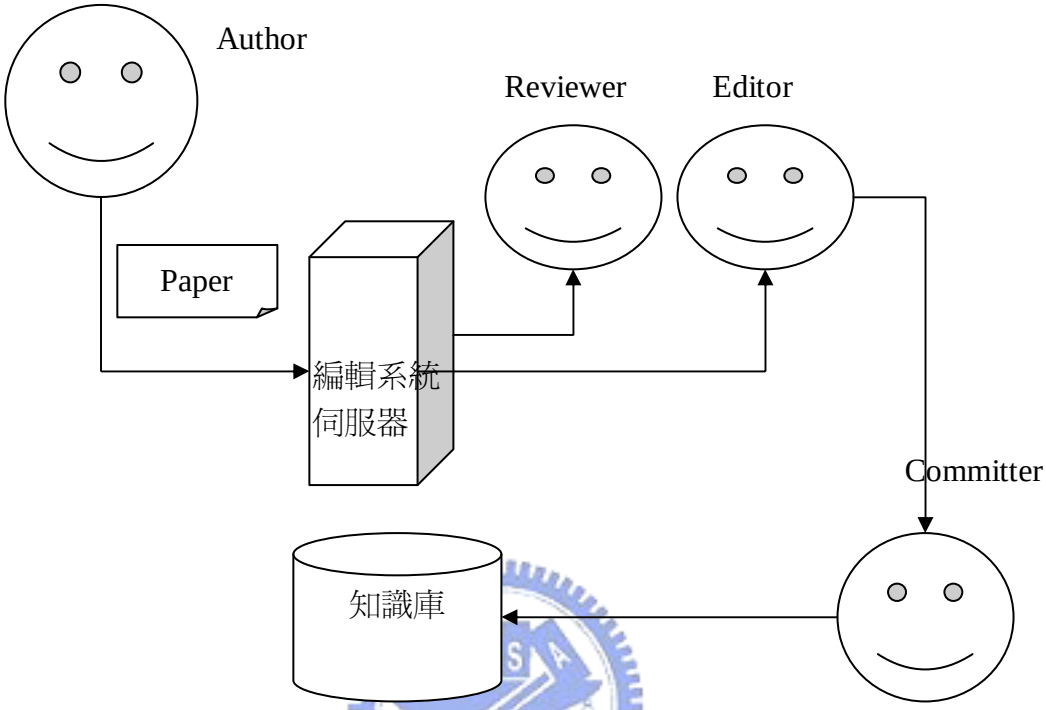


圖 3-2 期刊資料庫新增知識流程

圖 3-2 為期刊資料庫新增知識的流程，作者(Author)將論文標題、關鍵字、論文內文上傳至編輯系統的伺服器，經多位檢閱者(Reviewer)審核，並給予接受/拒絕的意見；編輯者(Editor)將檢閱者審核通過的論文編輯成冊，建立期刊目錄頁次後(Volume、Issue、頁次、年代)，將電子檔(論文內文、關鍵字等等)交給出版商之 Committer 加入知識庫中。

全文資料庫之資料新增與更新是由出版商/經營商負責，由於出版商/經營商所提供之領域類別架構及關鍵字分類機制可能與現有知識庫不同，知識庫管理者需要將其領域類別架構及關鍵字的對應關係加入知識庫中；若出版商/經營商之檢索資料不支援『問題-解法』知識分類法的關鍵字架構時，則知識庫管理者需將每一個更新的知識一一以『問題-解法』分類機制價入其領域、議題、解法及

運用技術等關鍵字。

依上述的流程，使用者角色共有：一般使用者(User)、Submitter(作者算 Submitter 的一種)、Committer、知識庫管理者(Knowledge Administrator)等四種類型[1]，各類使用者之功能需求如下：

1. 一般使用者主要作查詢知識以及閱讀知識。查詢知識可分為一般正向查詢以及逆向查詢，正向查詢為由大知識至小知識，由上而下查詢，以現有資料庫來說，從期刊、Issue、Volume、論文的順序來查詢論文；而逆向查詢則為輸入論文關鍵字，列出符合的論文，再挑選適合的論文來閱讀。若以『問題-解法』知識分類法來說，從領域類別、議題、解法、技術、論文的順序來閱讀，為正向查詢，輸入議題、解法或技術等關鍵字來查詢論文，則為逆向查詢。
2. Submitter：將知識的標題、作者、關鍵字、知識內文(如果有的話)等資料上傳至伺服器中，知識的來源可能是 Submitter 自己撰寫，也可能是從外面的資源取得。如果是期刊資料庫的話，Submitter 即是編輯者(Editor)，將已取得論文的標題、作者、關鍵字、內文等資料轉交給 Committer。
3. Committer：負責審核 Submitter 送過來的知識，檢查其中的關鍵字是否符合該論文的描述、格式是否正確，審核通過之後加入知識庫中。
4. 知識庫管理者：知識庫隨著時間的增加，領域類別會越來越多，知識管理者需調整領域類別架構(包含新增、搬移、分割領域類別)；此外還需增加技術論文議題，以供 Submitter 提交文件時當作關鍵字(領域類別、議題)的參考依據；及新增、調整解法、技術等關鍵字的列表，提供一般使用者查詢之用或 Submitter 提交文件時當作關鍵字的參考依據。在既有期刊資料庫、技術報告庫應用『問題-解法』分類法時，若要讓舊

[1]由於本論文僅實作知識庫的部分，編輯系統的部分，並不處理，因此沒有檢閱者的角色。

有的技術論文/報告能夠利用『問題-解法』關鍵字查詢，則需由知識庫管理者負責加入。在全文資料庫的狀況，知識管理者需要建立關鍵字的對應，以及將不支援『問題-解法』分類法的全文資料庫知識加入『問題-解法』關鍵字。在大型知識庫中，其資料量可能很大，需要建立分散式系統以及鏡射站台(Mirror Site)，此部分的工作亦由知識庫管理者負責。

表 3-1 列舉各類使用者會在哪些資料庫類型中出現，及該使用者角色會用到的功能需求。

角色	私人知識庫	期刊資料庫	技術報告庫	全文資料庫	使用功能/工作
一般使用者	O	O	O	O	1. 正向查詢 2. 逆向查詢 3. 閱讀知識
Submitter	O	O	O	X	從網路下載或其他管道取得知識，並針對該知識定義關鍵字，提交到資料庫
Committer	O	O	O	X	審核 Submitter 提交的知識，並將知識加入知識庫
知識庫管理者	O	O	O	O	1. 調整知識領域類別架構 2. 增加技術論文/報告之議題 3. 加入舊有知識 4. 調整出版商/編輯商所提供的全文檢索資料中的領域類別架構 5. 加入不支援『問題-解法』分類法的全文資料庫論文

表 3-1 使用者角色分類表

3.2節 Survey 類型論文性質

謝祖望學長所提出的『問題-解法』知識分類法並未考慮 Survey 型論文。一般論文只針對某一個議題提出特定解法，而 Survey 論文則是針對一個議題，提

出目前該議題下一至多個解法，並比較其優缺點，因此有必要針對 Survey 論文的性質作探討。

Survey 論文在四種知識庫都會出現，Survey 論文有兩種不同類型：『針對已發展成熟領域的主題分析其各種解法之特色』，以及『針對尚未發展成熟的領域，提出其可能的發展方向』。Simonetta Balsamo 等人的 “Model-based Performance Prediction in Software Development: A Survey”[12] 即是針對成熟領域所作的探討，此 Survey 論文主要在比較『Performance modeling and prediction』議題各種解法之優劣點；此類 Survey 論文的關鍵字可分為兩種：(1)所屬領域(Domain)與探討的問題(Problem)、(2)各種解法，例如此篇論文所訂定的關鍵字為『Software Verification』，『Software Verification』即為此篇論文的領域，而『Performance modeling and prediction』即為其探討的議題，此論文關鍵字也包括多種不同解法，如『Queuing Network-Based Methodologies』、『Process-Algebra-Based Approaches』、『Petri-Net-Based Approaches』、『Methodologies Based on Simulation Methods』、『A Methodology Based on Stochastic Process』；一般期刊資料庫或技術報告庫規定的關鍵字數目有限，有可能無法將所有的解法均納入關鍵字，而導致使用者無法利用解法關鍵字查詢到此 Survey 論文。有些 Survey 論文的解法，又可再分子解法，例如此論文的『Queuing Network-Based Methodologies』又可再分『Methodologies Based On SPE Approach』、『Architectural Pattern-Based Methodologies』、『Methodologies Based on Trace-Analysis』及『UML for Performance』等四種子解法。

Minoru Etoh 等人所著的”Wireless Video Applications In 3G and Beyond”[13]則為探討尚未發展成熟領域的 Survey 型論文，其探討的領域為 3G Mobile Network，論文中介紹目前之多媒體資料傳輸標準有『Circuit-Switched Multimedia Telephony』、『End-to-end Packed-Switched Multimedia

Telephony 』、『 Multimedia Messaging Service 』、『 Multimedia Broadcast/Multicast Service』，並介紹 Wireless Video 的 Error Control 該如何處理，如利用 Application Video Coding(MPEG4、H.264/AVC)、或 Transport Layer 的技術(Error Concealment、Feedback-based error control)，因此未成熟領域類的 Survey 論文探討之議題可能有好幾個，此篇論文之議題就有”Content Delivery”、”Error Control”兩個，每個議題又有一至多個解法；此外，作者可能會對某些議題提出可能解決方向，但未作深入說明，因此不應將此解法納入關鍵字中。

綜合上述，提出如圖 3-3 所示的 Survey 論文關鍵字架構，所屬領域包含 Survey 論文所探討的主題(主題可能會有多個)，主題之下有解法，而解法也會有子解法；圖 3-4 為“Model-based Performance Prediction in Software Development: A Survey”的關鍵字關係圖，圖 3-5 為”Wireless Video Applications In 3G and Beyond”的關鍵字關係圖。

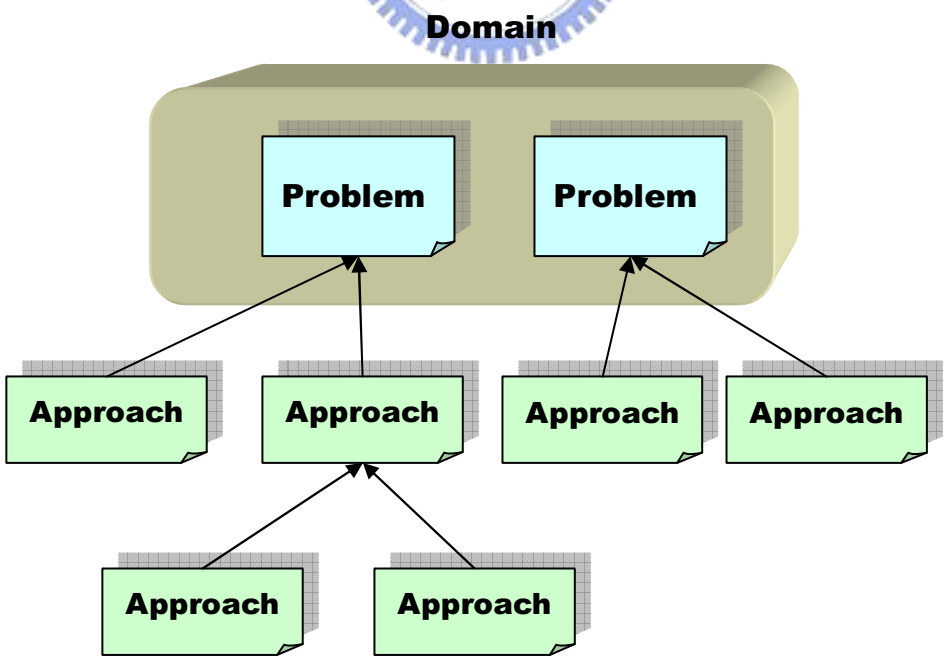


圖 3-3 Survey 論文關鍵字架構

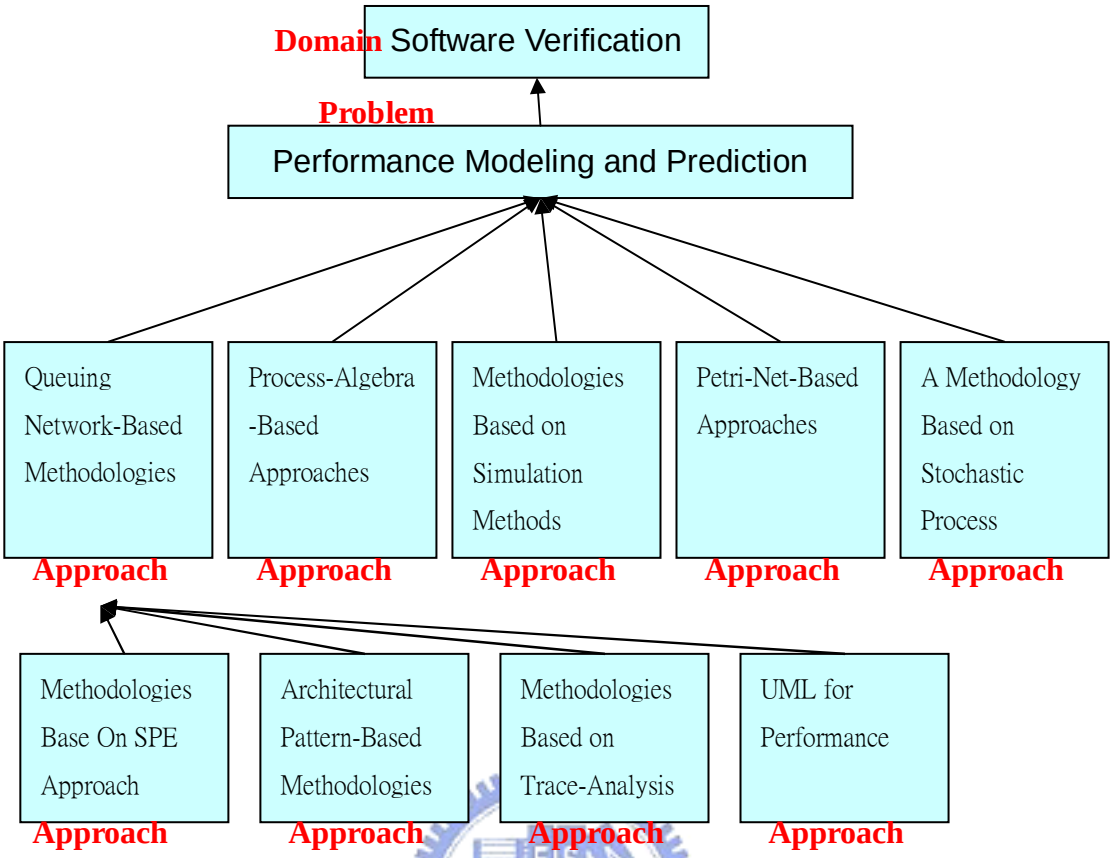


圖 3-4 Survey 論文範例 1

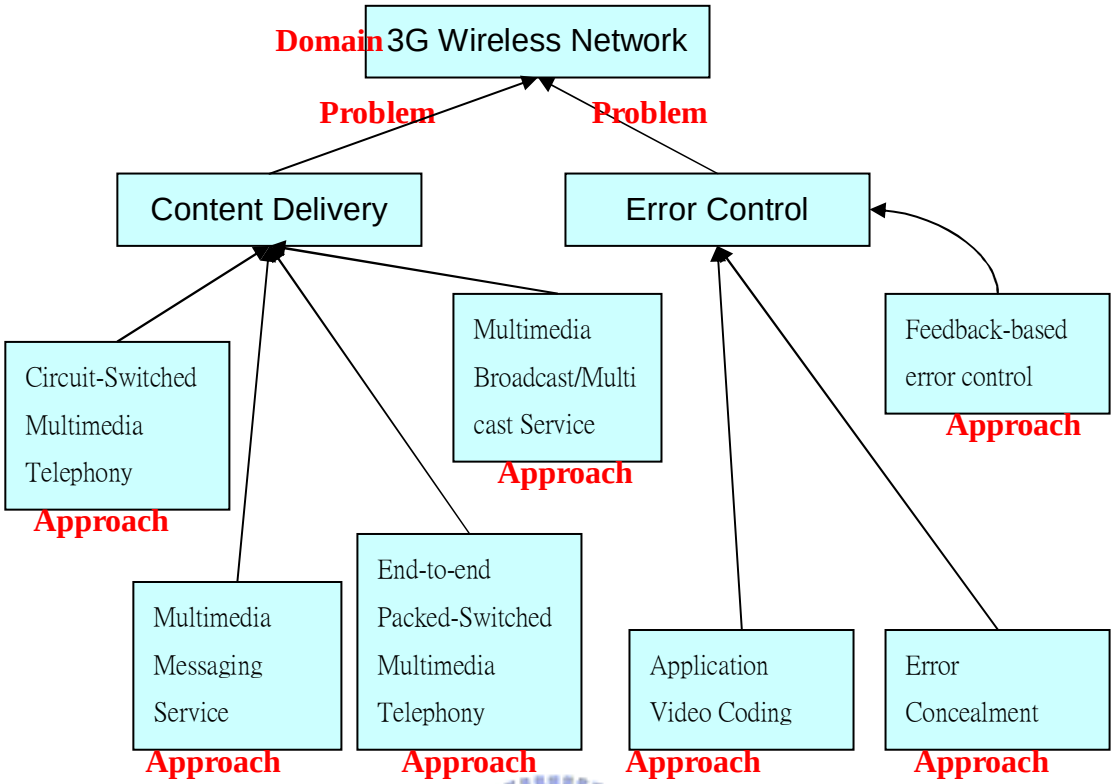


圖 3-5 Survey 論文範例 2

針對發展成熟領域探討的 Survey 論文，其知識架構如圖 3-6 所示，領域類別之下有一至多個 Survey 議題，每 Survey 議題之下有一至多篇 Survey 論文，每一 Survey 論文有多個解法，一解法又可能有一些小解法。未成熟領域探討的 Survey 論文之知識架構如圖 3-7 所示，領域類別之下有一至多個 Survey 議題，一 Survey 議題之下有一至多篇的 Survey 論文，一 Survey 論文有一至多個子議題，每一子議題有一至多個解法。

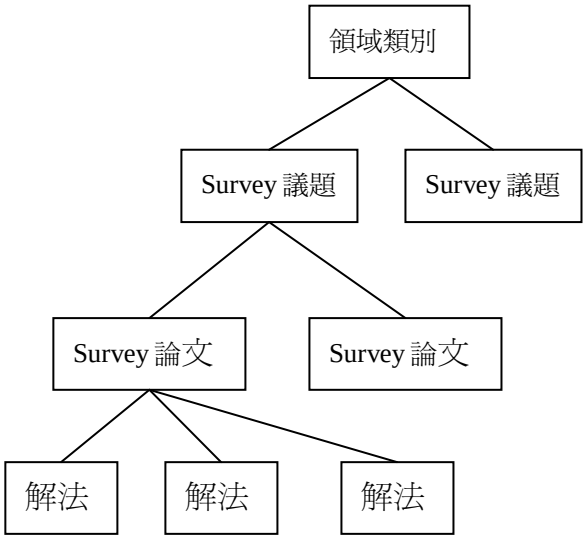


圖 3-6 發展成熟領域的 Survey 論文知識架構

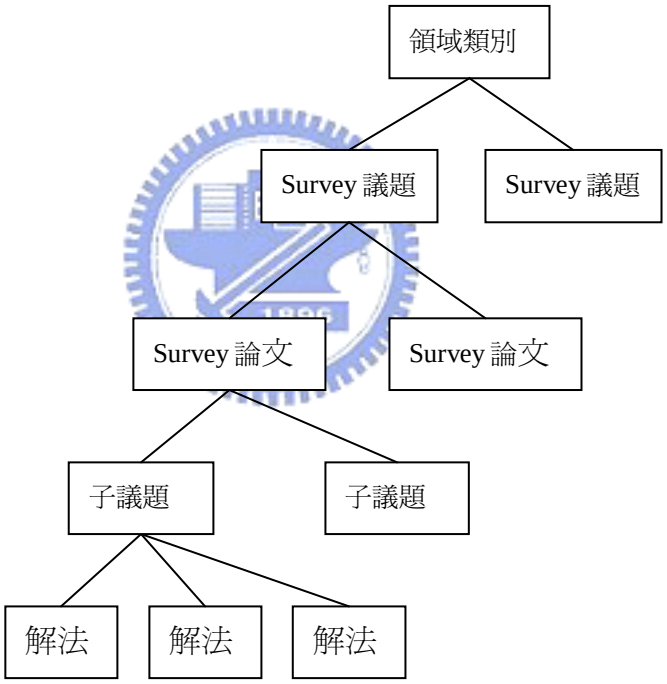


圖 3-7 未發展成熟領域的 Survey 論文知識架構

3.3節 大量資料儲存與查詢問題

3.1 節與 3.2 節說明了『使用者分類不清』、『缺少 Survey 性質的論文』的部分，剩餘下列四個問題尚待解決：

1. 使用單一伺服器處理查詢，無法處理大量知識資料以及大量查詢
2. 將索引資料全部載入至記憶體，無法應用在大量知識資料

3. 缺少逆向查詢

4. 缺少知識庫常用的查詢方法

這四個問題與知識庫種類不同有密切關係，解決策略亦不同，第一及二個問題僅發生於大型私人知識庫及期刊資料庫，對小型私人知識庫及技術報告庫無任何影響。逆向查詢的儲存資料方式會因為知識庫之逆向查詢表之大小而有所不同，『缺少知識庫常用的查詢方法』與知識庫性質有密切關係，本節將討論這些問題的解決方案。

知識庫可分為私人知識庫、期刊資料庫、技術報告庫、全文資料庫，如表 3-1 所示，四種知識庫儲存的知識種類、資料量以及領域範圍不同，設計一套知識庫系統套用到此四種知識庫，可能導致效率及資源利用不佳的問題，有必要依照知識庫特性而設計。小型私人知識庫的資料量少，領域範圍窄，可在一部電腦上建置；大型私人知識庫，則須利用分散式系統來儲存。由於期刊資料庫的領域範圍比大型私人知識庫大，且沒有書本知識，因此其知識種類形式與大型私人知識庫也有所不同，因此無法將大型私人知識庫系統套用到期刊資料庫。一般機構的技術報告庫資料量不多，其知識種類類似小型私人知識庫，但無書本知識和技術論文，此知識庫系統最為簡單。全文資料庫與期刊資料庫的差異僅在更新知識的方式，因此只要修改期刊資料庫的更新資料方式，即可套用。圖 3-8 為四種知識庫的關係，由此圖可知，技術報告庫為最簡單的知識庫，私人知識庫與技術報告庫之差異是增加書本知識，而大型私人知識庫、期刊資料庫與小型私人知識庫、技術報告庫相較，增加資料分散式之儲存，全文資料庫則是從期刊資料庫針對其資料更新方式作修改而成，因此可先討論技術報告庫之設計，再依差異而討論小型私人知識庫、大型私人知識庫、期刊資料庫、全文資料庫的設計。

	私人知識庫	期刊資料庫	技術報告庫	全文資料庫
知識種類	技術論文/報告、	技術論文、Survey	技術報告、Survey	技術論文、Survey

	Survey 論文、書本知識	論文	論文	論文
知識資料量	依據公司規模而定	資料量大	資料量小	資料量大
知識範圍	窄	廣	窄	廣
是否公開	不公開	公開	公開	公開

表 3-2 四種知識庫之差異

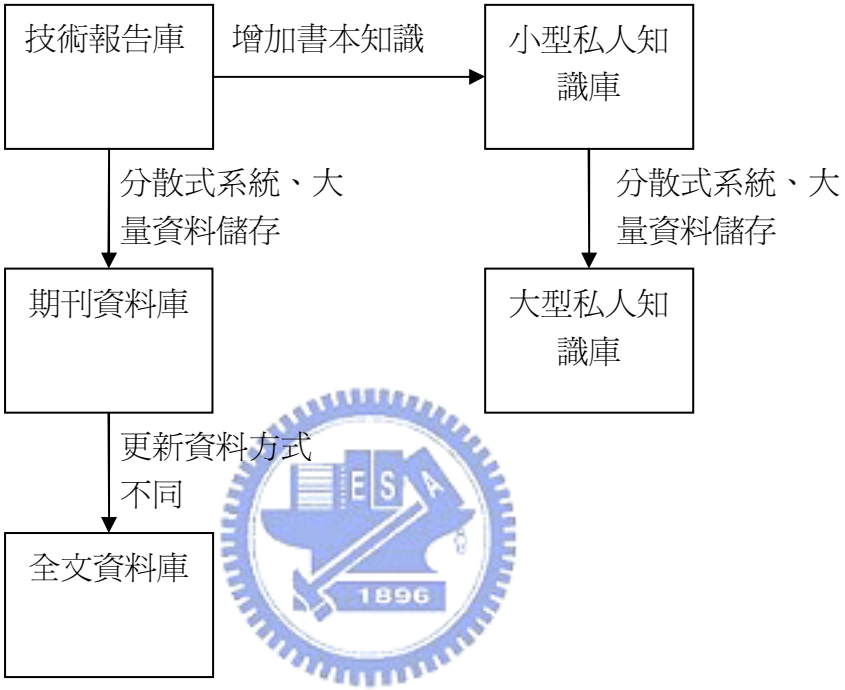


圖 3-8 四種知識庫改善的關係

3.3.1 技術報告庫

要討論技術報告庫之知識編碼、知識儲存、逆向查詢檢索資料儲存機制，首先應分析各知識需儲存哪些資料。表 3-3 為技術報告庫應有之知識節點儲存資料，領域類別節點應有基本資料包括名稱、簡介，領域類別下有子領域類別或技術議題，如果有技術議題的話，就不會有子領域類別；此外，由 3.2 節可知領域類別之下可有 Survey 議題，故領域類別節點需儲存『子領域列表/技術議題列表』及『Survey 議題列表』。Survey 議題節點紀錄的基本資料包含議題名稱及簡介，為了紀錄 Survey 議題節點之下有哪些 Survey 報告節點，因此 Survey 議題節點需紀錄其 Survey 報告列表；Survey 報告節點需要紀錄的資料包含名稱、作者、

年代、報告編號、摘要、及內文，如果 Survey 報告為成熟領域型的 Survey 報告，則 Survey 報告需要紀錄其列舉的解法；如果 Survey 報告為未成熟領域型的 Survey 報告，則需要紀錄其列舉的子議題以及子議題之下的解法。

知識節點	基本資料	其他資料
領域類別	名稱、簡介	子領域列表/技術議題列表、Survey 議題列表
Survey 議題	名稱、簡介	Survey 報告列表
Survey 報告	名稱、作者、報告編號、年代、摘要、全文	子議題列表或解法
技術議題	名稱、簡介	解法列表
解法	名稱、簡介、繼承解法	報告列表
技術報告	名稱、作者、報告編號、年代、摘要、全文	技術列表

表 3-3 技術報告庫知識節點儲存之基本資料一覽表

技術議題節點所需紀錄之基本資料有議題名稱及簡介，需儲存解法列表以指向所有解法節點；解法節點資料包含名稱、簡介及繼承解法，此外，解法節點下需紀錄報告列表以指向該解法的所有報告；技術報告節點需要紀錄的資料有名稱、作者、年代、報告編號、摘要、及內文，為了了解該報告使用哪些技術，因此需要紀錄其技術列表。

技術/解法關鍵字列表

在技術報告庫中，有許多報告運用相同的技術或解法，透過技術、解法的關鍵字列表，可查詢到此技術/解法之其他運用。關鍵字列表的儲存資料為編碼、名稱、及所有使用此技術/解法之技術報告，編碼由 1 一直往上加，由編碼可容易找到對應名稱。反之，名稱查詢編碼時，當技術關鍵字數量不多時，可利用 Radix Tree 方式來做分類，例如將前三個字母相同的關鍵字儲存在一個列表，列表之每一個 Entry 儲存資料為關鍵字名稱及編碼，且依其關鍵字之大小排序，搭配 Binary Search，即可找出名稱與編碼的對應關係。當編碼數量太多時利用 Radix Tree 方法查詢將非常耗時，可採用如圖 3-9 兩層的雜湊表(Hash table)

來儲存，名稱的前 10 個字元拿來作第一層 Hash 的 Key，其 Value 為第二層的雜湊表物件；第二層雜湊表物件則以名稱當作 Key，編碼當作 Value，當要查詢名稱對應到編碼的關係時，先利用名稱的前 10 個字元取得其第二層雜湊表物件，然後在利用名稱當作第二層雜湊表物件的索引，取得對應的編碼。

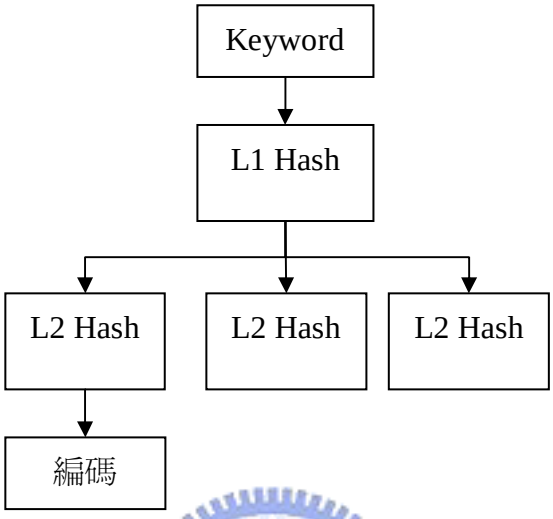


圖 3-9 解法列表查詢編碼示意圖

正向查詢索引資料之儲存方式

圖 3-10 為領域類別的知識列表，領域類別節點儲存的資料有領域類別名稱、簡介檔名、子知識列表指標，領域類別的簡介儲存在一個檔案，利用其檔名，即可列出簡介說明供使用者參考；領域類別之下的知識另外儲存成一個列表，並以子知識列表指標指向之；知識列表的資料包含：型別(說明其子知識節點為子領域或技術議題節點)、子領域/技術議題個數、Survey 議題個數、子領域/技術議題列表、Survey 議題列表指標。

Survey 議題節點儲存的資料有議題名稱、議題簡介檔名、Survey 報告列表指標，由於 Survey 議題之下的報告可能很多，因此另闢一個報告列表儲存，然後以報告列表指標指向之。Survey 報告節點儲存的資料有名稱、作者、年代、報告編號、摘要檔名、內文檔名、Survey 議題列表指標，Survey 報告的議題列表儲存的是 Survey 論文探討的大主題之下的小議題的資料，只有『未成熟領域

的 Survey 報告』才需要儲存小議題的資料，而『成熟領域的 Survey 報告』沒有小議題，而是解法，因此可分為兩種：

1. 未成熟領域的 Survey 報告：其小主題列表中儲存的資料為議題名稱、解法列表，解法列表為該議題之下的解法，儲存的資料為解法編碼、父解法編碼，儲存父解法編碼的原因是為了表示其大解法包含小解法的關係。
2. 成熟領域的 Survey 報告：如前所述，其主題專一，會列出解法並比較之，所以只需要儲存解法列表即可，其解法列表儲存的資料與『針對未成熟領域的論文』的小議題的解法列表一樣。

圖 3-10 領域類別知識列表儲存格式

如圖 3-11 所示，技術議題節點儲存的資料有議題名稱、簡介檔名、解法列表指標，解法節點儲存的資料有解法編碼、簡介檔名、繼承解法、技術報告列表指標，由於系統有建立解法的關鍵字列表，而且將關鍵字列表載入記憶體中，因

此不需要儲存解法名稱。技術報告節點需要儲存的資料包含：報告名稱、作者、年代、摘要檔名、內文檔名、報告編號、解法編碼、技術編碼列表，一篇技術報告的運用到的技術雖然很多，但是通常在 5、6 個技術之內，因此將報告的技術列表設定在最多使用 8 種技術。

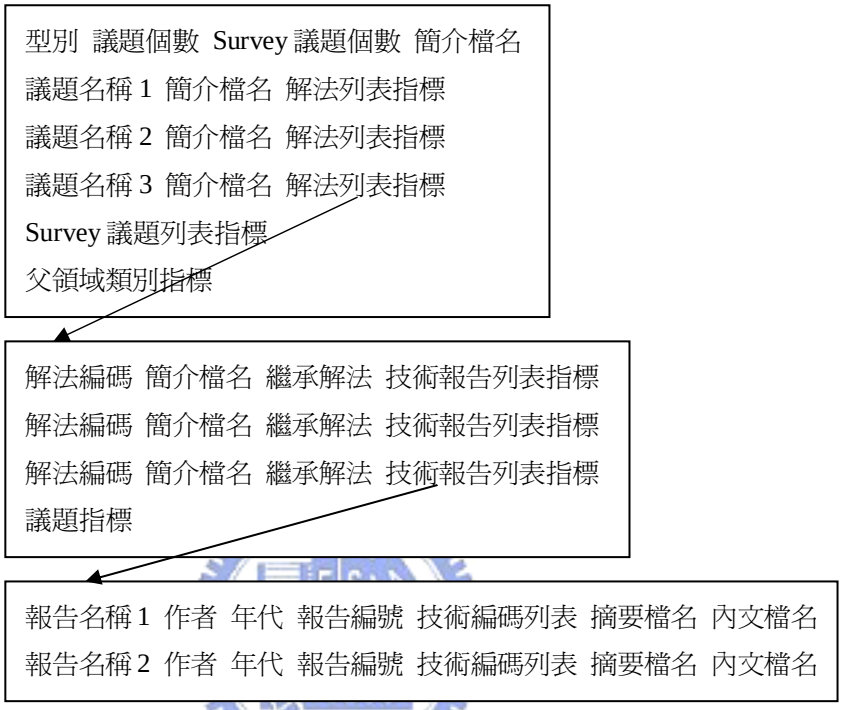


圖 3-11 技術報告儲存資料

知識編碼

逆向查詢的檢索資料，需要紀錄知識的完整路徑，因此需先制定知識的完整路徑編碼方式。領域類別架構為樹狀形式，ACM 的分類架構亦為樹狀形式，藉由 ACM 分類架構的子領域類別數目可估計領域類別的子領域類別數目。表 3-4 為 ACM 分類架構子領域數目之統計表，其子類別個數最多為 19，因此『問題-解法』知識分類架構中，每層領域類別(如圖 3-12 所示)可用 6Bit 表示，即最多可有 63 個，若不滿七層，則以下的層數均儲存 0，如 5-7-18-32-16-19-7 為第七層領域類別的編碼，其父領域類別編碼為 5-7-18-32-16-19-0。L8 代表技術議題的序號，L9 代表解法的序號，L10 代表技術報告的序號。L1 至 L10 共 60bit，如利用 64 位元機器，足以用一個 word 代表。

由於一領域類別之下可能同時有子領域或 Survey 議題，為了區分 Survey 議題與子領域/技術議題，以 63 來當做 Survey 議題的代號，如 5-7-18-32-16-63-0，則代表使用者要存取的是 5-7-18-32-16-0-0 這個領域類別之下的 Survey 議題，L8 代表 Survey 議題序號，L9 為 Survey 報告序號，由於 Survey 報告的子議題以及解法在顯示 Survey 報告時，以樹狀圖來表示，因此不需要編碼。

如此的編碼方式，最底層領域類別之下，可有 60 個議題、3,600 個解法、216,000 篇報告，如此的數量對於一般的領域類別已經足夠。然而，當一個新興領域(如量子電腦)，剛開始報告篇數不多，故將報告全放置於該領域類別之下，隨著時間的發展，該領域越來越成熟，發現無法將報告放置於領域類別之下，代表該領域需要切割成數個子領域以進行更細緻之分類。

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	18	19
第一層	0	0	0	1	0	3	3	0	2	2	0	0	0	0	0	0	0	0
第二層	2	0	1	7	14	9	8	1	4	3	1	2	1	0	1	0	1	0
第三層	9	22	24	26	33	19	16	10	9	6	2	4	2	1	0	1	0	1

表 3-4 ACM 子類別個數統計表

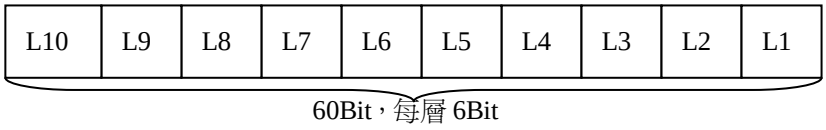


圖 3-12 知識編碼

逆向查詢檢索方式

有鑒於一般知識庫利用關鍵字查詢時列出一大堆知識，造成使用者不曉得哪些知識對他有幫助，因此本系統在逆向查詢時，系統將列出有哪些領域之議題下有該關鍵字之相關知識，再讓使用者挑選有興趣的領域及議題閱讀，系統才列出

該領域、議題下符合該關鍵字的知識，此對使用者有甚大方便。

使用者在查詢技術報告或 Survey 報告時，可利用『解法』或『技術』兩種關鍵字去查詢，圖 3-13 為『解法』、『技術』的逆向查詢檢索方式，系統在解法關鍵字列表中，儲存其逆向查詢資料指標，該資料儲存的型態有兩種：『技術報告編碼』、『Survey 報告編碼』。當使用者輸入解法關鍵字時，系統將關鍵字轉換成解法編碼，透過該編碼取得『逆向查詢資料指標』，經由該指標取得有哪些報告有談到該解法，依據領域類別排序，將同一議題的報告群組起來，並列出其領域類別路徑、議題名稱以及符合該解法關鍵字的報告列表。

技術關鍵字列表中，儲存了每一使用到此技術之技術報告編號，這些報告一編碼排序。當使用者輸入技術關鍵字作查詢時，系統先將關鍵字轉換成技術編碼與『逆向查詢資料指標』，取得所有運用該技術的技術報告，並依領域類別、技術議題、解法等順序一一列出。

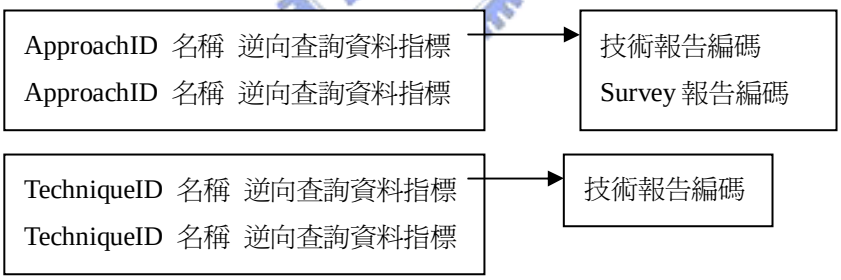


圖 3-13 論文/報告逆向查詢檢索資料

3.3.2 小型私人知識庫

小型私人知識庫與技術報告庫的差異有三：

1. 小型私人知識庫多了書本知識與技術論文
2. 知識來源可能是公司本身的知識，也可能來自於其他技術報告庫、期刊資料庫、或者網路上的資源，故需紀錄來源
3. Survey 論文來源多了期刊資料庫

知識節點	基本資料	額外資料
領域類別	名稱、簡介	子領域列表/技術議題列表、Survey 議題列表、書本知識列表
Survey 報告	名稱、作者、報告編號、年代、摘要、全文、報告庫	型別、子議題列表
技術報告	名稱、作者、報告編號、年代、摘要、全文、報告庫	型別、技術列表
Survey 論文	名稱、作者、期刊、年代、Volume、Issue、頁次、摘要、全文	型別、子議題列表
技術論文	名稱、作者、期刊、年代、Volume、Issue、頁次、摘要、全文	型別、技術列表
書本知識	名稱、作者、出版社、年代、簡介	書本議題列表
書本議題	名稱	父議題

表 3-5 私人知識庫知識節點儲存資料

表 3-5 為私人知識庫知識節點的儲存資料表，技術論文需要紀錄的資料有名稱、作者、年代、期刊、頁數、Volume、Issue、摘要、內文，為了與技術報告區分，因此需要加入型別這個欄位，及技術列表紀錄此論文所使用技術。技術報告需要紀錄來源報告庫。Survey 論文紀錄的資料有名稱、作者、年代、期刊、頁數、Volume、Issue、摘要、內文，為了與 Survey 報告區分，因此需要加入型別這個欄位，及加入 Survey 議題列表以紀錄論文討論的議題。如同技術報告，Survey 報告的來源不只一個，因此需要紀錄來源報告庫。由於期刊數量不多，因此可以將期刊名稱存在一個表格中，並且給予編碼，載入記憶體中，技術論文與 Survey 論文的期刊，只要儲存其期刊編碼即可。技術報告庫的做法也跟期刊相同。

書本知識節點儲存資料有名稱、作者、年代、出版社、簡介，為了儲存書本所探討的議題，因此需要加入書本議題列表，書本議題儲存的資料有議題名稱，為了表示議題包含的關係，書本議題需要加入父議題欄位。書本知識是介紹性的知識，其所探討的議題都是技術已經成熟的議題，在同一個領域類別下，可能會有好幾本書籍，例如 Operating System 之下就有 Sibberschatz 等人

的”Operating System Concepts”、Bach, M. J.的”The Design of the Unix Operating System”等書，也都會談到包含Process Synchronization、Memory Management 等等議題，因此書本議題名稱共用的情況非常多，故可儲存書本議題的關鍵字列表，其資料結構如同技術報告庫的解法、技術關鍵字列表，因此不在贅述。

由於私人知識庫多了書本知識，因此小型私人知識庫的領域類別節點的資料會多出書本知識列表指標，如圖 3-14 所示。

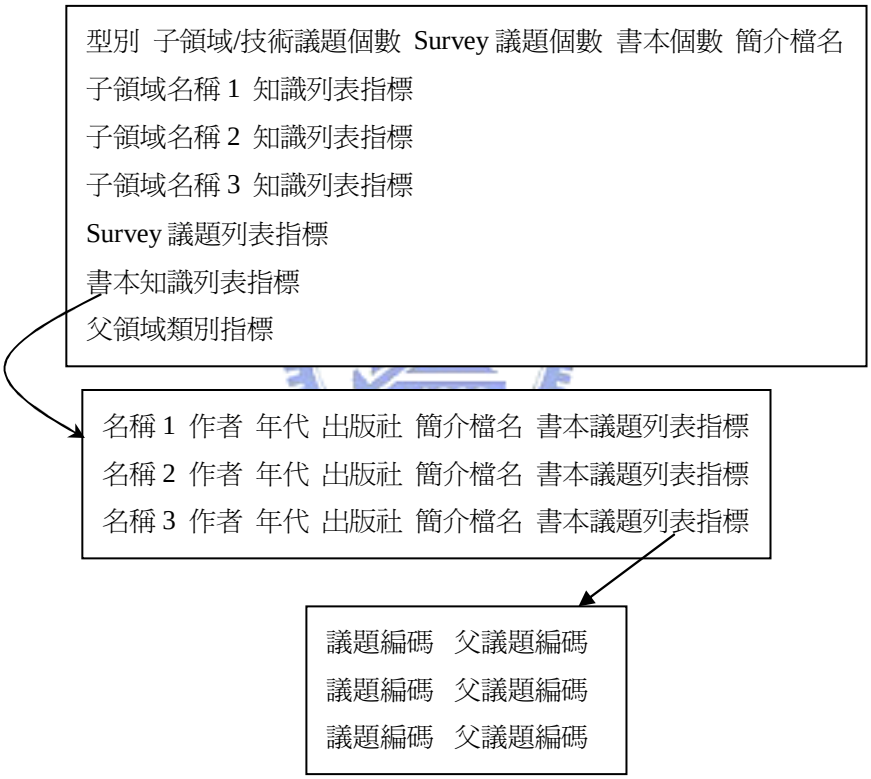


圖 3-14 書本知識資料

技術論文的編碼與技術報告的編碼相同，Survey 論文與 Survey 報告的編碼亦相同。由於一個領域類別之下可能同時有子領域、Survey 議題、書本知識，為了分辨其型態，以 62 來當做書本知識的代號，如 5-7-18-32-16-62-0，則代表使用者要存取 5-7-18-32-16-0-0 這個領域類別之下的書本知識。書本知識的範圍較廣，其所屬領域應該都在 L6 以上，如果書本知識所屬領域在 L5 或 L6，

如圖 3-15 所示，其下一層則為 62，L8 為書本序號，由於該書本位於 L5 或 L6，表示該書本的領域範圍很窄，書本數量應該不多，如此編碼是可行的。當書本知識所屬領域在 L4 以上時，其領域範圍較大，包含的書本知識可能會超過 64 本，因此用 L7、L8 來表示書本序號，亦即可有 3600 本書本，已經足以表示一個領域類別之下的書本。由於系統在顯示書本知識時，以樹狀圖顯示該書本知識的議題結構，因此不需將書本知識的議題納入編碼。

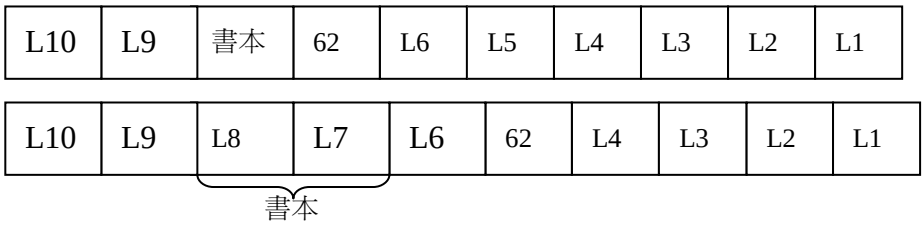


圖 3-15 書本知識編碼

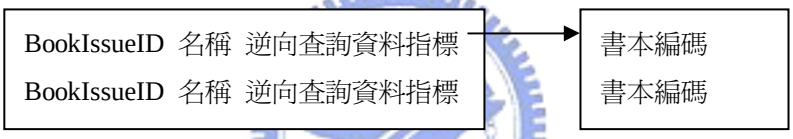


圖 3-16 書本知識逆向查詢資料

技術/Survey 論文的逆向查詢資料，與技術/Survey 報告的相同，不再贅述。系統在書本議題關鍵字列表中，儲存其逆向查詢資料指標，該資料儲存的型態為『書本編碼』，以領域類別排序。使用者在查詢書本時，可能會利用『書本議題』關鍵字去查詢，當使用者輸入『書本議題』時，系統將關鍵字轉換成書本議題編碼，透過該編碼取得『逆向查詢資料指標』，經由該指標取得有哪些書籍有談到該議題，並依照領域類別一一列出。

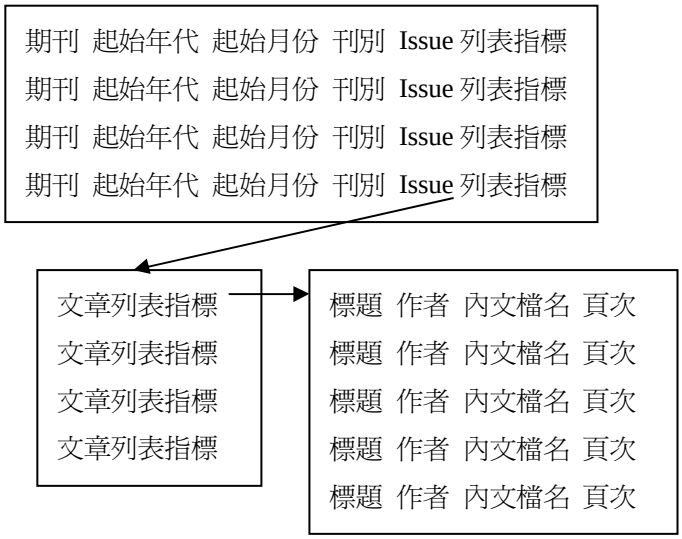


圖 3-17 期刊、Volume、Issue、論文之儲存資料

私人公司可能會訂閱期刊，因此使用者可能會查詢某 Volume 某 Issue 的論文有哪些。期刊通常以一年為一個 Volume，視其刊別(月刊、雙月刊、季刊)而分為有 12 個 Issue、6 個 Issue、4 個 Issue，因此只要紀錄其起始年代、起始月份、刊別，就可以計算出現在是第幾個 Volume、第幾個 Issue，如圖 3-17 所示，期刊列表的資料儲存刊明、起始年代、起始月份、刊別、其 Issue 列表指標，該 Issue 列表儲存了每一期期刊的文章列表，當使用者指定要第 5 個 Volume 第 2 個 Issue 時，假設期刊是季刊，因此就知道是要 4*4+2=第 18 個 Issue 的論文列表，就只要抓取相對應的檔案即可知道該期期刊的文章列表。論文列表中每一筆資料儲存著論文的標題、作者、頁次、內文檔名，其中內文檔名可用來分辨該文章是論文還是其他文章(如編輯者的話)，如果是論文的話，內文檔名儲存的資料為論文的編碼，如果是其他文章，則是該文章的檔名。

3.3.3 大型私人知識庫

當公司規模逐漸變大，知識越來越無法儲存在單一伺服器中，需要考慮利用多台伺服器來分散其資料，因此大型私人知識庫與小型私人知識庫的差異在於其知識量較大，需要分散式系統來處理。圖 3-18 為大型私人知識庫的架構，利用多台 Application Server 分散儲存知識資料，並由 Web Server 負責處理分散式

的查詢。由於領域類別編碼採用階層式的編碼，此種編碼方式適合分散式的架構，Web Server 只要儲存前幾層編碼與 Application Server 之間的對應關係，當 Web Server 收到使用者的查詢，透過領域類別編碼可得知需要轉交的 Application Server，因此不必改變領域類別編碼。

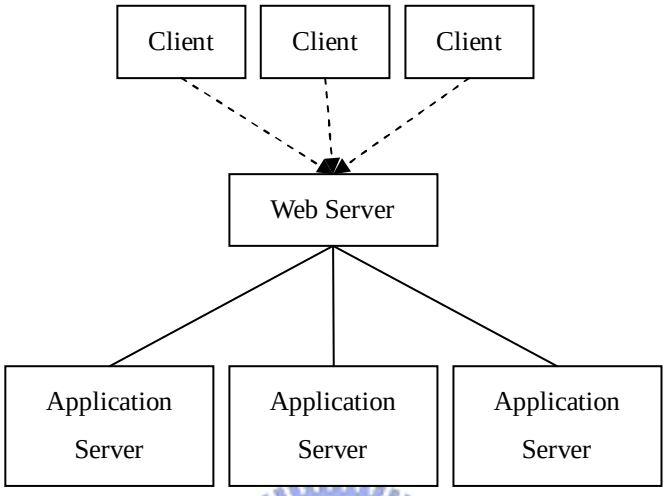


圖 3-18 大型私人知識庫架構

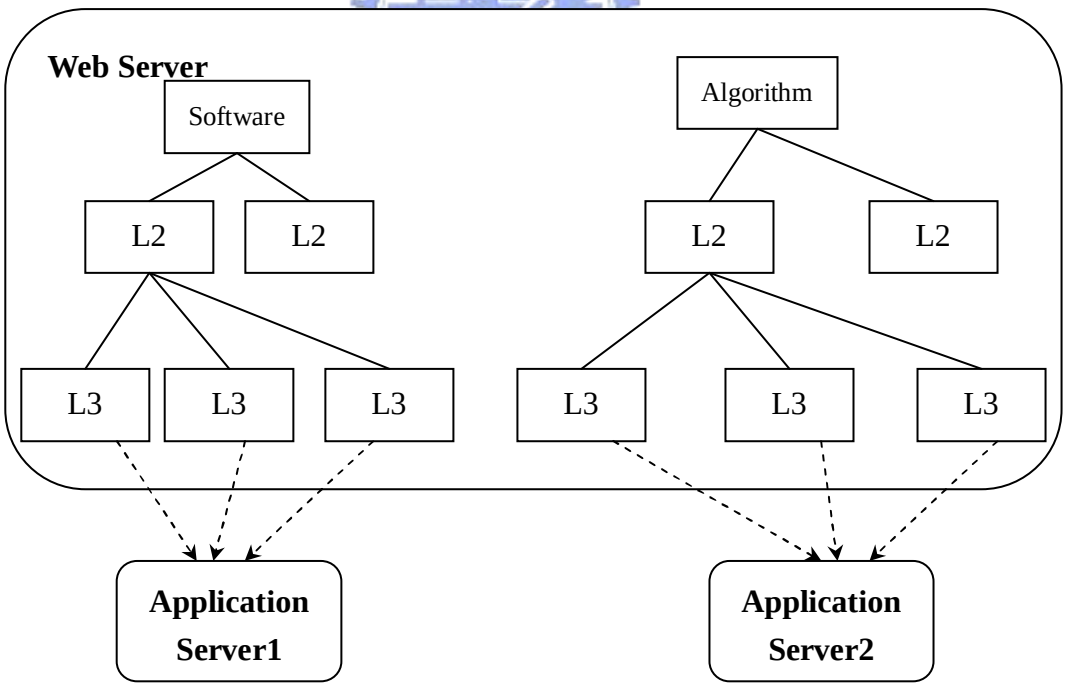


圖 3-19 知識分散範例

『問題-解法』知識分類法的查詢以『領域類別』為基礎，因此以領域類別為切割的基準，將同一個領域類別劃分在同一台伺服器，如圖 3-19 所示，

Software、Algorithm 為最上層領域類別，其下的領域類別資料一起放在 Web Server，當索引資料無法完全放在 Web Server 時，將部份索引資料分散在 Application Server。Web Server 儲存上層的領域類別資料，當 Web Server 啟動時，將領域類別的資料(如子領域/技術議題列表、Survey 議題列表、書本知識列表)載入至記憶體，當使用者查詢這些領域類別資料時，從記憶體中抓取資料回應，若該層領域類別之下的資料在第二層伺服器中，則在領域類別節點的子知識列表指標作一個註記(如全部為 1)，則代表該領域類別節點之下的知識位於 Application Server，當使用者要查詢其底下的知識時，Web Server 便轉交第二層伺服器查詢。為了知道該領域類別下的知識位於哪個伺服器中，Web Server 需要紀錄第一層領域類別或第二層領域類別底下的資料放在哪台 Application Server(隨著其資料大小而定，如果第一層領域類別下的資料無法完全放在一台伺服器中，必須分散到第二層領域類別)，如 Software 底下的知識放在 Server1，Algorithm 底下的知識放在 Server2，當使用者從 Software 開始找尋資料，到達某個領域類別節點時，發現該節點的子知識列表註記為儲存在第二層伺服器中，則 Web Server 將查詢轉交給該 Application Server。

Web Server 除了儲存上層領域類別資料以及轉交查詢至第二層伺服器以外，還需要處理逆向查詢所得資料的結果，例如使用者想查詢 Dynamic Programming 在 Software、Algorithm 領域下，有哪些論文運用該解法，Web Server 將 Dynamic Programming 關鍵字傳給儲存 Software、Algorithm 的 Application Server，Application Server 將結果傳回給 Web Server，Web Server 按照領域排序整合所得結果並回應給使用者，因此 Web Server 除需將上層領域類別資料載入至記憶體，尚需保留記憶體空間存放伺服器回傳的結果。

由於私人知識庫的知識有其隱密性，而有些知識只能限制某些使用者存取，因此 Web Server 必須將使用者身份的資訊轉交給 Application Server，讓

Application Server 判斷該使用者是否可以存取該知識。

3.3.4 期刊資料庫

期刊資料庫的知識量通常較為龐大，需要採取類似大型私人知識庫的架構，期刊資料庫與大型私人知識庫的最大差異在於其提交知識的流程，私人知識庫提交知識時，只要 Committer 負責審核即可，然而期刊論文需要多位 Reviewer 審核其論文，審核通過後，需要 Editor 負責將論文套上期刊的樣版(如加上期刊標誌、名稱、Volume、Issue、頁次等等)，然後在交由 Committer 負責將編輯好的論文上傳。此部分需要編輯系統，而編輯系統已經相當完善，且期刊資料庫的查詢系統也相當成熟，因此期刊資料庫的知識庫系統可分為兩套：『現有查詢系統』與『問題-解法系統』，使用者在查詢時，可先選擇要使用哪套系統。

由上述可知，期刊資料庫與大型私人知識庫的差異在於：

1. 期刊資料庫沒有書本知識及報告
2. 需要透過某種機制與現有系統整合

第一個差別在於其儲存資料不同，沒有書本知識，則領域類別不需要儲存書本列表及其個數，也不需要建立書本議題的關鍵字列表及其逆向查詢資料；沒有技術報告，解法的論文列表僅有技術論文，Survey 議題的論文列表也只有技術論文。第二個差異則讓知識庫需透過 DOI 機制向現有期刊系統取得論文全文。

圖 3-20 為期刊資料的系統架構，使用者在進入 Web Server 時，可選擇使用『現有期刊系統』或『問題-解法系統』，當使用者要下載論文全文時，『問題-解法』系統便透過 DOI 機制向現有期刊系統取得論文全文。

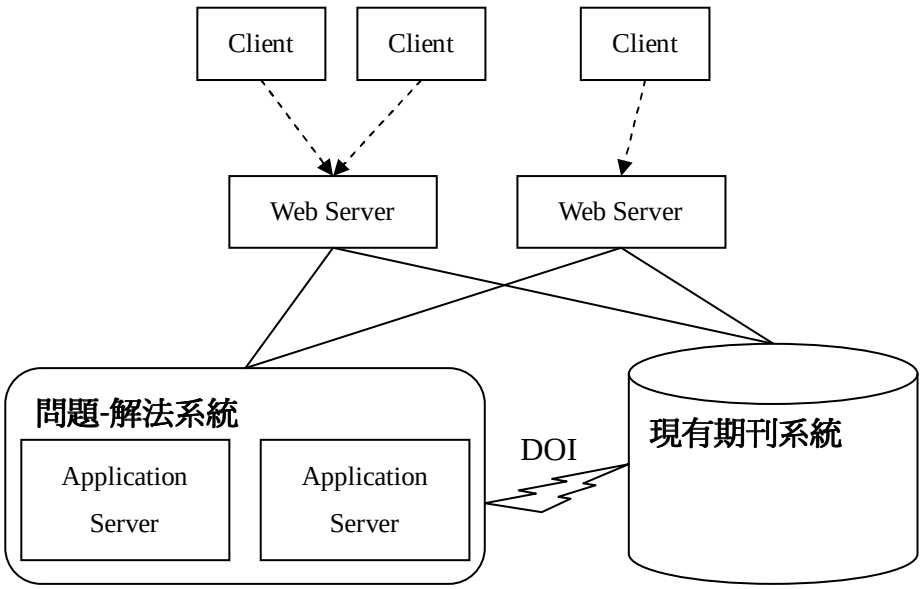


圖 3-20 期刊資料庫系統架構

3.3.5 全文資料庫

全文資料庫與期刊資料庫的差異僅在於其資料更新方式。全文資料庫為出版商安裝於私人機構中的期刊資料庫，其資料更新是由出版商負責，當出版商更新全文資料庫後，如果該全文資料庫支援『問題-解法』知識分類法，則直接使用該全文資料庫的檢索資料。如果該全文資料庫不支援『問題-解法』知識分類法，而機構想要將全文資料庫中的資料納入『問題-解法』分類法的查詢，則需要手動將知識資料建入資料庫中，所以需要提供一個介面，讓知識庫管理者能夠一一將新增的知識資料建立其『問題-解法』關鍵字。

第4章 技術報告庫之系統實作

第三章已說明各類知識庫面臨的問題及其解決策略，由第三章可知技術報告庫系統的設計最為簡單，依次是小型私人知識庫、大型私人知識庫、期刊資料庫、全文資料庫，因此本章先說明技術報告庫的知識庫系統之設計原理，第五章介紹其他知識庫系統的設計原理。

本章首先在 4.1 節說明技術報告庫的規格，在 4.2 節說明伺服器端使用的資料結構，4.3 節說明報告庫客戶端與伺服器端的設計。

4.1節 技術報告庫之規格

技術報告庫的資料量通常不多，與謝祖望學長所提出的雛型系統一樣可用一台伺服器儲存所有技術文件的相關資料，然而該雛型系統實際應用在技術報告庫有下列不足之處：

1. 技術報告庫沒有期刊論文及書本
2. 技術報告分『特定主題的研究或設計報告』及『Survey 類報告』，兩者性質不同
3. 雛型系統沒有逆向查詢功能
4. 雛型系統允許所有人都可以存取知識資料，但有些技術報告庫對人員使用有不同的使用限制

第一、二點可藉由修改系統內資料結構而解決，第三點則需在知識庫系統加入 Inverted File 機制，才能處理逆向的搜尋。第四點則需在知識庫系統內加入存取控制(Access Control)機制，依技術報告的限閱類別來決定使用者可否讀取。

為方便使用者在任何地方均可使用，本系統採用 Web-Based 架構，如圖 4-1 所示，使用者透過瀏覽器，經由 HTTP 協定，向系統的 Web Server 查詢，由 Application Server 取出所需技術報告，而使用者的帳號與密碼登資料儲存在 User DB Server。

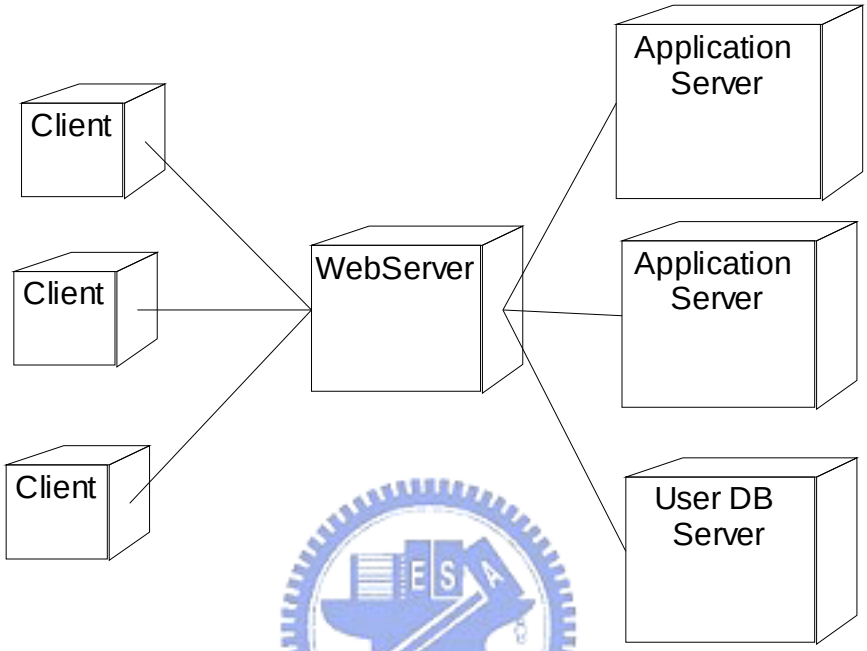


圖 4-1 系統架構初步規劃圖

使用案例

根據第三章，技術知識庫的使用者可分為四種：一般使用者、知識庫管理者、Submitter 及 Committer，圖 4-2 為系統的使用案例圖(UseCase Diagram)，當使用者要使用知識庫系統時，首先需透過身分認證機制確認身分，此機制詢問使用者帳號、密碼，使用者輸入帳號、密碼，系統驗證該帳號密碼是否正確，若是則讀取使用者資料，並依照使用者的帳號設定該使用者的身分，往後的查詢或修改知識庫資料都是以該使用者身份進行操作。

一般使用者的使用案例可分為一般查詢類與輸入資料查詢類，一般查詢為按照領域類別架構樹由上而下依序的查詢，輸入資料查詢則為輸入關鍵字、作者、

或年代等資料後進行查詢；使用者除了查詢以外，還可以訂閱某議題下新增的報告通知，當系統新增報告時，便會依據使用者的訂閱而將該資料 Email 給使用者。。知識管理者的使用案例套件為『知識管理』，Submitter 的使用案例套件為『提交報告』，Committer 的使用案例套件為『審核報告』。

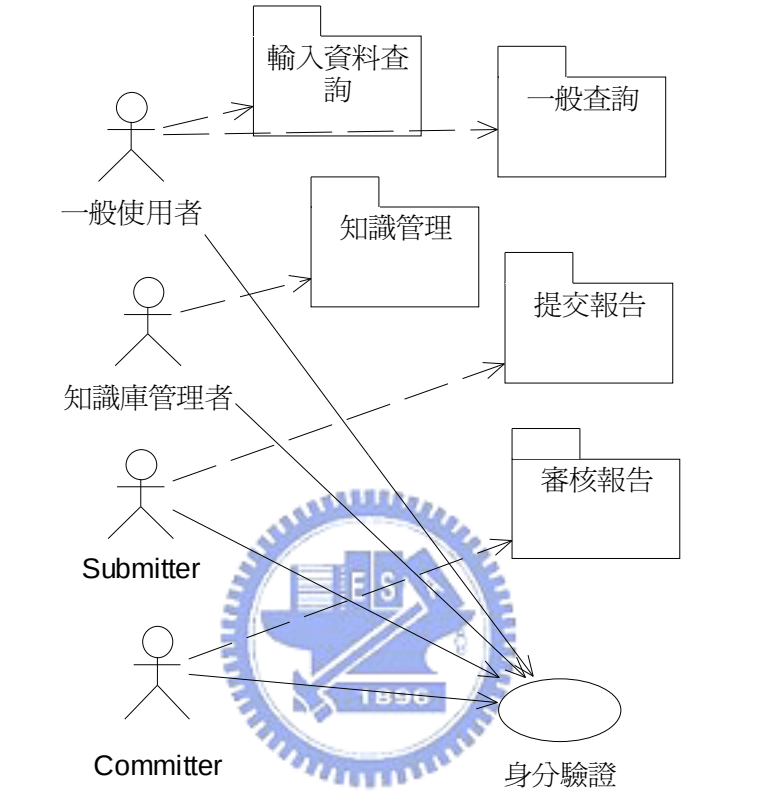


圖 4-2 使用案例圖

制定一般使用者、Submitter、Committer、知識管理者的使用案例前，首先應確定每一個知識的儲存格式，方能設計知識庫應執行的工作。假設技術報告庫有 12500 個領域類別，其中最下層有 10000 個領域類別，每一個領域類別下有 10 個技術議題、5 個 Survey 議題，每一技術議題下有 5 個解法，每一解法下有 5 篇報告，每一 Survey 議題下有 5 篇報告，依各節點所需儲存資料可估算其所需儲存的資料量，如表 4-1 所示。每個知識節點都有其儲存的基本資料，以及其他儲存資料(如簡介、摘要、全文等等)，領域類別、技術議題、Survey 議題、解法節點的基本資料數量不多，可個別用一個檔案來存放所有資料，『知識資料』資料夾下有『領域類別』、『技術議題』、『Survey 議題』、『解法』等四個檔案分

別儲存領域類別、技術議題、Survey 議題、解法節點的基本資料。為了增加查詢速度，這些資料有必要分別放置於主記憶體之領域類別節點、技術議題節點、Survey 議題節點、解法節點，並建立其索引機制。

領域類別、技術議題、Survey 議題、或解法的簡介資料量較大，若儲存在主記憶體，所佔空間龐大，因此以檔案儲存，其儲存方式依據知識編碼而定，假設一個解法的第一層領域類別編碼為 10，第二層為 5，第三層為 20，技術議題編碼為 8，解法編碼為 4，其第一層領域簡介儲存在『知識資料/10』資料夾下的檔案 intro 中，第二層領域簡介儲存在『知識資料/10/5/』資料夾下的檔案 intro 之中，技術議題的簡介儲存在『知識資料/10/5/20/Issue8/』資料夾下的檔案 intro 之中，解法簡介儲存在『知識資料/10/5/20/Issue8/4/』資料夾下的檔案 intro 之中。

知識	資料大小	數量	總合	其他儲存資料
領域類別	64	12500	0.8MB	簡介(1KB)，共 12.5MB
技術議題	64	10 萬	6.4MB	簡介(1KB)，共 100MB
Survey 議題	64	5 萬	3.2MB	簡介(1KB)，共 50MB
解法	32	50 萬	16MB	簡介(1KB)，共 500MB
技術報告	256	250 萬	640MB	摘要(2KB)，共 5GB 全文(500KB)，共 1.25TB
Survey 報告	256	25 萬	64MB	摘要(2KB)，共 500MB 全文(500KB)，共 125GB 子議題列表(512Byte)，共 128MB

表 4-1 知識儲存資料量估計表

如表 4-1 所示，技術報告的基本資料較為龐大(640MB)，無法完全載入記憶體之中，為了加快存取報告節點的資料，將同一個解法節點下的技術報告資料放在一起，其報告列表儲存方式，依據其領域類別路徑、技術議題、解法、報告編碼而定，假設一個解法的第一層領域類別編碼為 10，第二層為 5，第三層為 20，技術議題編碼為 8，解法編碼為 4，則其報告列表儲存在『知識資料/10/5/20/Issue8/4』資料夾的檔案 reportList，若是 Survey 報告列表，則為

『知識資料/10/5/20/Survey8』資料夾的檔案 reportList。報告資料以區塊的方式儲存在報告列表檔案之中，區塊大小為 256Byte，假設報告的編碼為 3，則該報告的資料儲存在報告列表的第三個區塊。新增報告時，將報告資料附加在報告列表檔案的檔尾，並以該區塊的編號當作報告的編號。

報告摘要以一個摘要一個檔案的方式儲存，會產生 250 萬個檔案，而報告摘要都是小檔案，在分割儲存報告列表的硬碟分割區時，可以選擇較小的區塊大小(Block Size)，例如 1KB，這樣一個分割區可以容納許多檔案。報告摘要的儲存方式，依據其領域類別路徑、技術議題、解法、報告編碼而定，假設一篇報告的第一層領域類別編碼為 10，第二層為 5，第三層為 20，技術議題編碼為 8，解法編碼為 4，報告編碼為 3，則其摘要儲存在『知識資料/10/5/20/Issue8/4』資料夾的檔案 3.abs，若是 Survey 報告摘要，則為『知識資料/10/5/20/Survey8/』資料夾的檔案 3.abs。

Survey 報告的子議題列表並不儲存在 Survey 報告的基本資料中，而是另外儲存集中在一個檔案儲存，一篇 Survey 報告的子議題列表約可用 512Byte 儲存，其儲存方式與報告摘要雷同，一個子議題列表為一個檔案，一其路徑儲存，舉以上的例子，該 Survey 報告的子議題列表儲存在『知識資料/10/5/20/8/』資料夾下的檔案 3.subissue 中。

由上述可知，『知識資料』這個資料夾依據領域類別為分類儲存著領域類別簡介、技術議題簡介、Survey 議題簡介、解法簡介、解法的技術報告列表、Survey 議題的 Survey 報告列表、技術報告的摘要、Survey 報告的摘要、及 Survey 報告的子議題列表等資料，以這種方式儲存的好處再於當需要切割成多台伺服器時，只要將某個領域類別資料夾下的檔案複製到另外一台即可。另外，以這種方式儲存知識資料的話，大約會產生 400 萬個檔案，而且這些檔案都是小檔案，因

此特別針對儲存『知識資料』的檔案系統作調整，檔案系統的邏輯區塊大小(Logic Block Size)需要調整為 1KB，且檔案系統的最大檔案個數要調整為可容納 500 萬個以上的檔案。

技術報告的全文與 Survey 報告全文資料龐大(共 1.4TB)，需要三顆 500GB 的硬碟儲存，由於報告的全文都是單獨下載，而且檔案量較大，因此以檔案儲存全文資料。報告的全文檔案，共有 275 萬個檔案，加上其檔案總大小約為 1.4TB，在小型技術報告庫中，一開始可能沒有這麼多檔案，所以不會利用多顆硬碟去儲存，但是報告越來越多時，就需要額外配置硬碟儲存。為此，報告全文依據其路徑儲存，舉上例來說，該報告的全文檔案放在『全文/10/5/20/Issue8/4』資料夾下的檔案 3，如果是 Survey 報告，則放在『全文/10/5/20/Survey8/』資料夾的檔案 3。假設一開始系統報告不多，領域 A 與領域 B 下的報告都放在磁碟 C，但是當報告越來越多時，必須加掛磁碟 D，因此只要將領域 B 的資料夾搬到磁碟 D 即可。

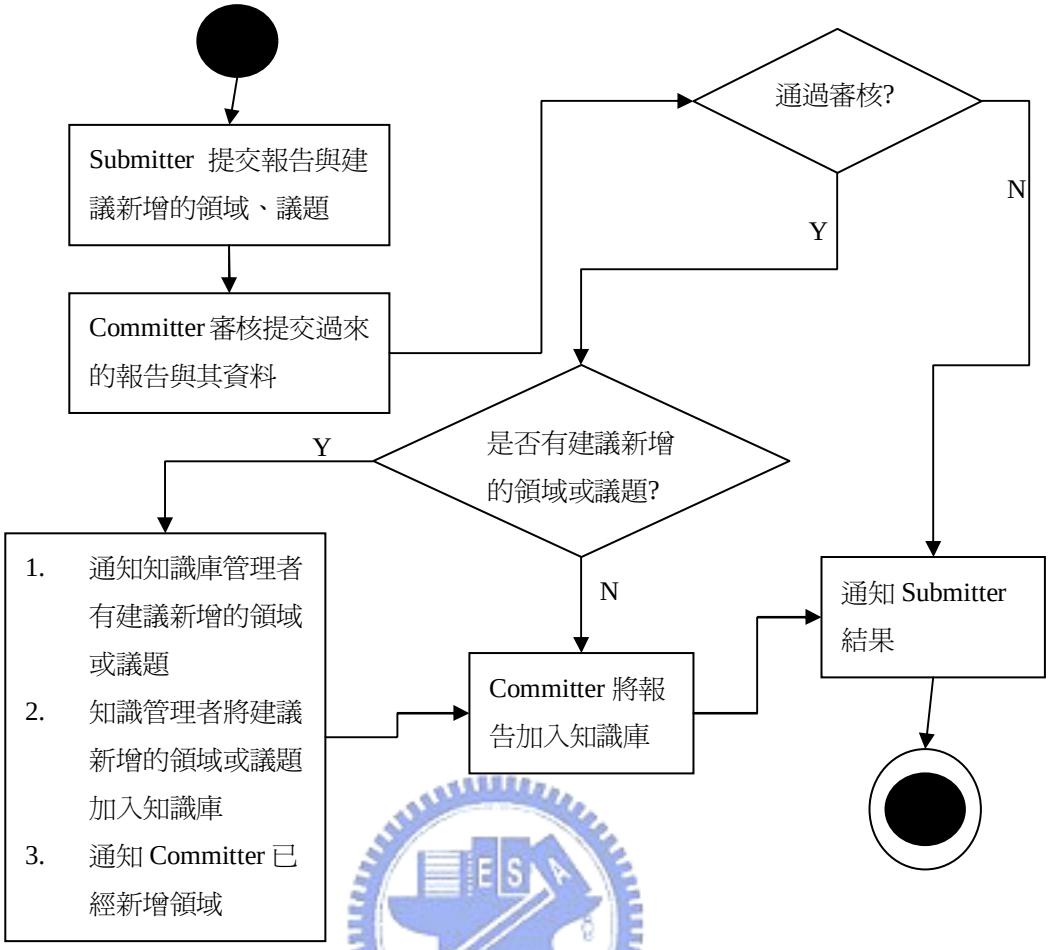


圖 4-3 新增報告流程

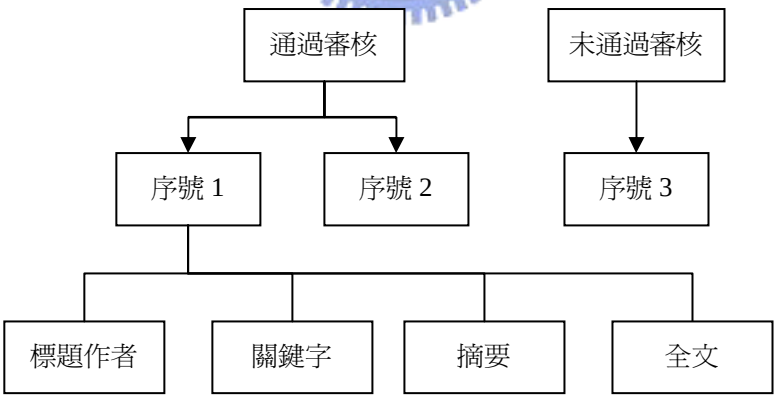


圖 4-4 新增報告流程所產生的檔案

圖 4-3 為新增報告的流程，Submitter 在提交報告時，若發現知識庫沒有適合的領域或議題時，則需將建議新增的領域或議題資料一併上傳。圖 4-4 是報告新增流程所產生的檔案，Submitter 將報告提交過來時，依據目前時間給予該報告一個序號，根據序號在『未通過審核』資料夾下建立資料夾，該資料夾下有

『標題作者檔案』、『關鍵字檔案』、『摘要檔案』、『全文檔案』，關鍵字檔案的格式如圖 4-5 所示，若該報告是技術報告則內容包括領域類別路徑、技術議題名稱、解法名稱、繼承解法名稱(若該報告的解法有繼承的解法)、技術名稱列表，如果該報告是 Survey 報告，則內容包括領域類別路徑、Survey 議題名稱、子議題名稱列表及子議題下的解法樹狀關係，若有建議新增的知識，則會有 new 這個標籤，其簡介資料以 introduction 標籤標示。當 Committer 接受該報告後，該報告會移動至『通過審核』資料夾，如果有建議新增領域類別、技術議題、或 Survey 議題，則分別會產生『add_category』、『add_issue』、『add_surveyissue』三個檔案，並通知系統管理者將建議新增的知識加入知識庫中。當知識庫管理者加入建議新增的領域類別、技術議題、或 Survey 議題後，系統自動移除對應的 add_category、add_issue、add_surveyissue 三個檔案並產生『add.done』檔案。當 Committer 欲執行將報告加入知識庫時，系統會去掃描『通過審核』資料夾下的報告資料夾是否有『add.done』檔案，若有則提醒 Committer 將該報告加入知識庫，加入工作完成後，系統自動將該報告自資料夾移除。

如前所述，使用者可以訂閱某議題下新增報告通知，因此在 Committer 加入報告後，需將該報告編碼加入到『本日新增報告列表』檔案中，而使用者的訂閱紀錄，則依據領域類別分類，如一個議題的第一層領域類別編碼為 10，第二層為 5，第三層為 20，技術議題編碼為 8，則該議題的訂閱紀錄放在『訂閱紀錄/10/5/20/』中的 Issue8 檔案中，該檔案儲存使用者編碼。每到晚上時，系統會先將『本日新增報告列表』檔案複製一份，並將『本日新增報告列表』檔案清空，然後透過副本取得所有新增的報告編碼列表，先將該列表排序後，依據領域類別、議題編碼，取得該議題的訂閱使用者列表，然後透過使用者編碼查詢使用者的 Email，並將新增報告的訊息傳送給使用者。

```
<categorypath>
   <category>
      <name>Database</name>
   </category>
   <category>
      <name>Text Database</name>
      <new/>
      <introduction>…      </introduction>
   </category>
</categorypath>
<issue>
      <name>String Data Structure</name>
      <new/>
      <introduction>…</introduction>
</issue>
<approach>
      <name>Burst tries</name>
      <new/>
      <inherent>Ternary search Tree</ inherent >
      <introduction>…</introduction>
</approach>
<techniques>
      <technique>Splay Trees</technique>
      <technique>Tries</technique>
      <technique>Binary Trees</technique>
      <technique>Hash Table</technique>
</techniques>
```

圖 4-5 關鍵字檔案格式

一般查詢的使用案例

一般查詢的使用案例包括『檢視領域類別』、『檢視技術議題』、『檢視解法』、『檢視技術報告』、『檢視 Survey 議題』、『閱讀知識簡介』、『檢視 Survey 報告』、『下載報告全文』、『訂閱議題新增報告通知』，分別說明如下：

- 檢視領域類別

Precondition：畫面上列出某一領域類別之所有子領域名稱後，使用者點選其中一子領域

一般使用者	系統
1. 點選領域類別的子領域類別其中一個	
	2. 讀取子領域類別編號 3. 利用子領域類別編號從目前知識節點指標中找出對應的子節點指標 4. 更新目前知識節點指標 5. 讀取領域類別名稱，並顯示出來 6. 讀取子領域類別/技術議題列表並顯示出來 7. 檢查是否有 Survey 議題，若是，則讀取 Survey 議題列表並顯示

- 檢視技術議題

Precondition：使用者已經瀏覽到最下層的領域類別節點，畫面上列出其技術議題列表。

一般使用者	系統
1. 點選技術議題列表中其中一個技術議題	
	2. 讀取技術議題編號 3. 利用技術議題編號從目前知識節點指標中找出對應的技術議題節點指標 4. 更新目前知識節點指標 5. 讀取解法列表並顯示

- 檢視解法

Precondition：使用者已經瀏覽到技術議題節點，畫面上列出其解法列表。

一般使用者	系統
1. 點選解法列表其中一個解法	
	2. 讀取解法編號 3. 依解法編號從目前知識節點找出對應的解法節點指標 4. 更新目前知識節點指標 5. 透過解法節點的知識編碼計算出報告列表檔名，從報告列表檔案讀取報告列表資料(名稱、作者、年代)並顯示

- 檢視技術報告

Precondition：使用者已經瀏覽到解法節點，畫面上顯示其報告列表。

一般使用者	系統
1. 點選技術報告列表其中一篇報告	
	2. 讀取報告編號 3. 利用報告編號從目前解法節點指標中找出該報告節點 4. 檢查使用者是否有閱覽報告的權限，若無則顯示沒有權限閱讀，結束 Use Case 5. 更新目前知識節點指標 6. 讀取節點中的技術編碼列表 7. 從技術關鍵字列表將該報告所運用的技術列舉出來 8. 利用報告編碼計算出摘要檔名，從摘要檔案讀取摘要 9. 顯示摘要、技術列表等資料，並顯示[下載全文]連結

- 檢視 Survey 議題

Precondition：使用者瀏覽到領域類別節點，畫面上顯示其 Survey 議題列表。

一般使用者	系統
1. 點選 Survey 議題列表中其中一個議題	
	2. 讀取 Survey 議題編號 3. 利用 Survey 議題編號從目前知識節點指標中找出對應的 Survey 議題節點指標 4. 更新目前知識節點指標 5. 利用 Survey 議題編碼計算出報告列表檔名，從報告列表檔案讀取報告列表資料(作者、名稱、年代)並顯示

- 閱讀知識簡介

Precondition：使用者已經瀏覽到領域類別、技術議題、Survey 議題、或解法節點

一般使用者	系統
1. 點選[閱讀簡介]	
	2. 從目前知識節點指標中讀取知識編碼 3. 透過知識編碼計算出簡介檔案名稱，從簡介檔案中讀取資料

	4. 顯示簡介資料

- 檢視 Survey 報告

Precondition：使用者瀏覽到 Survey 議題節點，畫面上列出所有 Survey 報告。

一般使用者	系統
1. 點選 Survey 報告列表其中一篇報告	
	2. 讀取報告編號 3. 利用報告編號從目前知識節點指標中找出該報告節點 4. 檢查使用者是否有閱覽報告的權限，若無則顯示沒有權限閱讀，結束 Use Case 5. 更新目前知識節點指標 6. 利用報告編碼計算出子議題列表檔名,並從該檔案讀取子議題列表(包含名稱、其下的解法列表) 7. 利用報告編碼計算出摘要檔名，從摘要檔案讀取摘要 8. 顯示摘要、子議題列表等資料，並顯示[下載全文]連結

- 下載報告全文

Precondition：使用者已經瀏覽到技術報告、或 Survey 報告節點。

一般使用者	系統
1. 點選[下載全文]	
	2. 從目前知識節點取得報告編碼 3. 利用報告編碼計算全文檔名，讀取該檔案的資料 4. 顯示報告全文

- 訂閱議題新增報告通知

Precondition：使用者已經瀏覽到技術議題節點或 Survey 議題節點。

一般使用者	系統
1. 點選[訂閱新增報告通知]	
	2. 從目前知識節點指標中讀取知識編碼 3. 透過知識編碼計算出訂閱使用者列表檔案名稱，將該使用者編號加到該檔案檔尾

輸入資料查詢的使用案例

輸入資料查詢包括『利用解法關鍵字查詢報告』、『利用技術關鍵字查詢報

告』、『利用作者查詢報告』、『列出報告庫某年代的報告』等 4 種使用案例，分別說明如下：

- 利用解法關鍵字查詢報告

  Precondition:使用者瀏覽到『技術議題』、『解法』、『技術報告』、『Survey 報告』節點，畫面上出現『利用解法關鍵字查詢』連結

一般使用者	系統
1. 點選『利用解法關鍵字查詢』	
	2. 依據目前知識節點指標，列出所有解法(議題節點取出該議題下的解法，解法、技術報告節點則直接取出其解法，Survey 報告則取出該報告所列舉的解法)
3. 選擇一解法	
	4. 讀取解法名稱 5. 列出目前知識節點的第一層領域類別、第二層領域類別…一直到最該知識節點的父領域類別
6. 選擇其中一個領域	
	7. 從該領域類別編碼，找出該領域類別下的子領域類別列表 8. 列出子領域類別列表
9. 重複步驟 6~8，直到使用者點選『設定領域範圍』按鈕	
	10. 讀取領域範圍編碼 11. 依據解法名稱，至解法關鍵字列表中找出該解法對應的逆向查詢索引 12. 根據逆向查詢索引找出有用到該解法的報告編碼 13. 比較報告編碼是否落於領域範圍之中，若是則依據該編碼取得報告節點，並透過報告節點取得該報告的名稱、作者 14. 將所有報告依據領域類別、議題排序 15. 依領域及議題順序，顯示其領域類別名稱、議題名稱及該議題下有用到該解法的報告之標題及作者

- 利用技術關鍵字查詢報告

  Precondition：使用者瀏覽到『技術報告』節點，畫面上出現『利用技術關鍵字查詢』連結

一般使用者	系統
1. 點選『利用技術關鍵字查詢』	

	2. 據目前知識節點，取出該技術報告節點所使用的技術並顯示之
3. 選擇一技術	
	4. 讀取技術名稱 5. 列出目前知識節點的第一層領域類別、第二層領域類別…一直到最該知識節點的父領域類別
6. 選擇其中一個領域	
	7. 從該領域類別編碼，找出該領域類別下的子領域類別列表 8. 列出子領域類別列表
9. 重複步驟 6~8，直到使用者點選『設定領域範圍』按鈕	
	10. 讀取領域範圍編碼 11. 依據技術名稱，從技術關鍵字列表中找出該技術對應的逆向查詢索引 12. 根據逆向查詢索引找出有用到該技術的報告編碼 13. 比較報告編碼是否落於領域範圍之中，若是則利用報告編碼取得該報告節點，並讀取報告名稱、作者 14. 將所有報告依領域類別、議題排序 15. 依領域及議題順序，顯示其領域類別名稱、議題名稱、及該議題下有用到該技術的報告之標題及作者

- **利用作者查詢報告**

一般使用者	系統
1. 點選[利用作者查詢]	
	2. 顯示作者欄位
3. 輸入『作者』	
	4. 讀取作者名稱 5. 依據作者名稱，從作者列表中找出該作者所有的報告編碼 6. 利用報告編碼找出該報告的節點並讀取該報告的標題、作者、年代，並顯示之

- **列出報告庫某年代的報告**

一般使用者	系統
1. 點選[利用年代查詢]	
	2. 依據報告庫列表顯示所有的報告庫
3. 點選其中一個報告庫	
	4. 讀取報告庫 5. 依據年代列表顯示所有的年代
6. 點選其中一個年代	
	7. 讀取年代

	8. 依據報告庫與年代，取得該年代的報告列表檔案 9. 從報告列表檔案讀取報告編碼，並利用報告編碼找出該報告的節點並取出報告的標題、作者 10. 顯示報告的標題及作者

Submitter 負責的使用案例有二：提交新技術報告、提交新 Survey 報告，分別說明如下：

- 提交新技術報告

Precondition：當 Submitter 瀏覽到最末端領域類別，畫面上出現『提交技術報告』連結

Submitter	系統
1. 點選[提交技術報告]	
	2. 詢問是否要建議新增領域類別
3. 若不新增領域類別，跳到步驟 13	
	4. 顯示名稱與簡介欄位
5. 輸入領域名稱與簡介	
	6. 讀取領域名稱與領域簡介 7. 顯示議題名稱與簡介欄位
8. 輸入議題名稱與簡介	
	9. 讀取議題名稱與議題簡介 10. 顯示解法名稱與簡介欄位
11. 輸入解法名稱與簡介	
	12. 讀取解法名稱與解法簡介，跳至步驟 28 13. 列出該領域下的議題，並詢問是否要建議新增技術議題
14. 若不新增技術議題，則選擇一個議題，跳到步驟 21	
	15. 顯示議題名稱與簡介欄位
16. 輸入議題名稱與簡介	
	17. 讀取議題名稱與議題簡介 18. 顯示解法名稱與簡介欄位
19. 輸入解法名稱與簡介	
	20. 讀取解法名稱與解法簡介，並跳至步驟 28 21. 讀取技術議題編碼 22. 列出該議題下的解法，詢問是否要新增解法
23. 若不新增解法，則選擇其中一個	

解法，跳到步驟 27	
	24. 顯示解法名稱、簡介、繼承解法欄位
25. 輸入解法名稱、簡介、繼承解法	
	26. 讀取解法名稱、解法簡介、繼承解法，跳至步驟 28 27. 讀取解法編碼 28. 顯示八個『技術』欄位
29. 輸入技術關鍵字	
	30. 讀取技術關鍵字 31. 顯示標題、作者、摘要、全文等欄位
32. 輸入標題、作者、摘要，並選擇全文檔案上傳	
	33. 讀取報告的標題、作者、摘要、全文檔案 34. 依據目前時間取得報告序號 35. 在『未通過審核』資料夾下建立名稱爲報告序號的資料夾 36. 將報告全文資料寫入報告資料夾的全文檔案 37. 將報告摘要資料寫入報告資料夾的摘要檔案 38. 將報告的標題、作者寫入報告資料夾的標題作者檔案 39. 將領域類別路徑寫入報告資料夾的關鍵字檔案，若有建議新增領域，則加入 new 標籤，並以 introduction 標籤標示該領域的簡介 40. 將技術議題名稱寫入關鍵字檔案，若爲建議新增領域，則加入 new 標籤，並以 introduction 標籤標示該議題的簡介 41. 將解法名稱及其繼承解法名稱寫入關鍵字檔案，若爲新解法，則加入 new 標籤，並以 introduction 標籤標示該解法的簡介 42. 將技術名稱列表寫入關鍵字檔案 43. Email 通知 Committer 有報告提交 44. 顯示完成提交報告

- 提交新 Survey 報告

Precondition：當 Submitter 瀏覽到領域類別，畫面會出現『提交 Survey 報告』連結

Submitter	系統

1. 點選[提交 Survey 報告]	
	2. 詢問是否要建議新增領域類別
3. 若不新增領域類別，跳到步驟 10	
	4. 顯示名稱與簡介欄位
5. 輸入領域名稱與簡介	
	6. 讀取領域名稱與領域簡介 7. 顯示議題名稱與簡介欄位
8. 輸入議題名稱與簡介	
	9. 讀取議題名稱與議題簡介，並跳至步驟 16 10. 列出該領域類別下的 Survey 議題，並詢問是否要新增 Survey 議題
11. 若不新增 Survey 議題，則選擇一個 Survey 議題，跳至步驟 15	
	12. 顯示議題名稱與簡介欄位
13. 輸入名稱與簡介	
	14. 讀取議題名稱與議題簡介 15. 紀錄議題編碼 16. 顯示子議題欄位
17. 輸入子議題	
	18. 讀取子議題資料 19. 顯示解法欄位
20. 輸入解法資料	
	21. 讀取解法資料 22. 顯示標題、作者、摘要、全文等欄位
23. 輸入標題、作者、摘要，並選擇全文檔案上傳	
	24. 讀取報告的標題、作者、摘要、全文檔案等資料 25. 依據目前時間取得報告序號 26. 在『未通過審核』資料夾下建立名稱爲報告序號的資料夾 27. 將報告全文資料寫入報告資料夾的全文檔案 28. 將報告摘要資料寫入報告資料夾的摘要檔案 29. 將報告的標題、作者寫入報告資料夾的標題作者檔案 30. 將領域類別路徑寫入報告資料夾的關鍵字檔案，若有建議新增領域，則加入 new 標籤，並以 introduction 標籤標示該領域的簡介 31. 將 Survey 議題名稱寫入關鍵字檔案，若爲建議

	新增 Survey 議題，則加入 new 標籤，並以 introduction 標籤標示該議題的簡介 32. 將子議題列表及其解法寫入關鍵字檔案 33. Email 通知 Committer 有報告提交 34. 顯示完成提交報告

Committer 負責審核 Submitter 提交的報告，其使用案例包括『審核報告』、『將報告加入知識庫』、『建議新增領域』等 3 個使用案例，說明如下：

- 審核報告

審核報告的流程較為複雜，其流程如圖 4-6 所示。

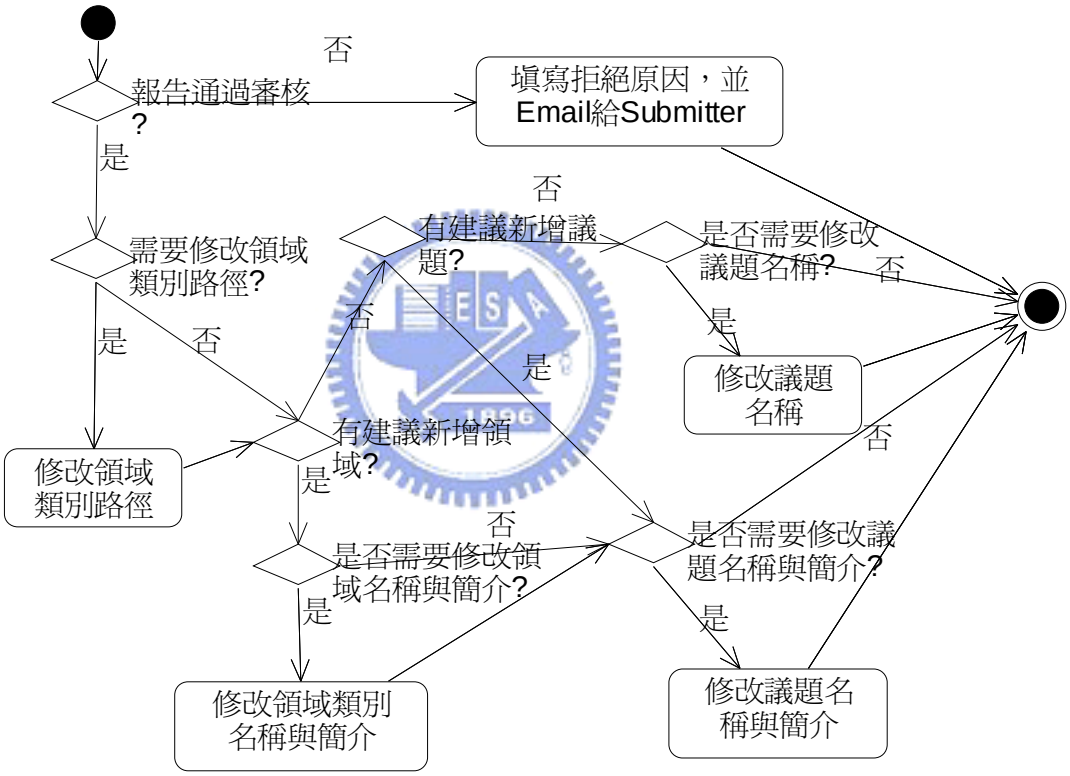


圖 4-6 審核報告流程

Precondition：如果系統有尚待審核的報告，則在 Committer 使用系統時，會出現『審核報告』

Committer	系統
1. 點選[審核報告]	
	2. 從『待審核報告』目錄中，讀取每一個報告資料夾的標題作者檔案，並顯示標題、作者

3. 點選其中一篇報告	
	4. 讀取報告序號 5. 依據報告序號從該報告資料夾中讀取其全文資料，並顯示之 6. 顯示『接受』與『拒絕』按鈕
7. 若接受該報告，跳到步驟 13	
	8. 顯示『拒絕原因』欄位
9. 輸入拒絕原因	
	10. 刪除該報告資料夾 11. 將拒絕原因 Email 給 Submitter 12. 顯示刪除完畢，並結束 Use Case 13. 從該報告的關鍵字檔案讀取關鍵字資料(包含領域類別名稱、議題名稱、解法名稱、技術列表，若有建議新增的知識則一併讀取)，並顯示領域類別路徑，詢問是否需要修改領域類別路徑
14. 若不需修改，跳至步驟 17 15. 修改領域類別路徑	
	16. 讀取領域類別路徑，更新關鍵字資料中相對應的領域類別名稱 17. 檢查該報告是否有建議新增的領域類別，若否，則跳至步驟 26 18. 顯示建議新增的領域類別名稱、簡介
19. 若不修改則跳至步驟 22 20. 修改建議新增領域類別名稱與簡介資料	
	21. 讀取建議新增領域名稱、簡介，並更新關鍵字資料中建議新增的領域名稱、簡介 22. 顯示建議新增的議題名稱、簡介
23. 若不修改則跳至步驟 35 24. 修改建議新增議題名稱與簡介資料	
	25. 讀取建議新增議題名稱、簡介，更新關鍵字資料中建議新增議題名稱、簡介，跳至步驟 35 26. 檢查該報告是否有建議新增的議題，若否，則跳至步驟 31

	27. 顯示建議新增的議題名稱、簡介
28. 若不修改則跳至步驟 35 29. 修改建議新增議題名稱與簡介資料	
	30. 讀取建議新增議題名稱，更新關鍵字資料的建議新增議題名稱、簡介，跳至步驟 35 31. 顯示報告的議題名稱，詢問是否需要修改
32. 若不需修改，跳至步驟 35 33. 修改議題名稱	
	34. 讀取議題名稱，並更新關鍵字資料中的議題名稱 35. 將修改過的關鍵字資料寫入報告資料夾中的關鍵字檔案 36. 將報告資料夾搬移到『通過審核』資料夾 37. 若有建議新增的領域類別，則在報告資料夾中加入 add_category 檔案 38. 若有建議新增的技術議題，則在報告資料夾中加入 add_issue 檔案 39. 若有建議新增的 Survey 議題，則在報告資料夾中加入 add_surveyissue 檔案 40. 若沒有加入任何 add 類型的檔案，則在報告資料夾中加入 add.done 檔案 41. 若有任何 add 類類型的檔案，則 Email 通知知識管理者需將建議新增的知識加入知識庫系統 42. 顯示『審核完畢』

- 將報告加入知識庫

Precondition：有報告已經通過審核且其建議新增的領域類別、議題都已經加入系統，Committer 在使用系統時，會出現『加入報告』連結

Committer	系統
1. 點選[加入報告]	
	2. 從『通過審核』資料夾中，找出有 add.done 檔案的報告資料夾，並從該報告資料夾的標題作者檔案讀取報告的標題、作者
3. 點選其中一篇報告	
	4. 讀取報告序號 5. 列出報告庫欄位

	6. 顯示『加入』按鈕
7. 選擇報告庫	
	8. 讀取報告庫名稱 9. 從目前時間取得年代 10. 從關鍵字檔案讀取領域類別路徑、議題名稱、解法名稱、技術名稱列表等資料，並依據路徑與議題名稱找到技術議題/Survey 議題節點 11. 若是技術報告，則檢查該解法是否爲新解法，若是則從技術議題節點的解法列表中取得新的解法編碼，並將解法關鍵字編碼、解法編碼寫入解法資料檔案。 12. 若是技術報告，則依據解法名稱從議題節點的解法列表取得解法節點，並將解法節點設爲父節點 13. 從父節點的報告列表，找出新的報告編碼 14. 建立報告節點，將標題、作者、年代、報告庫、報告編碼等資料填入 15. 利用報告編碼，計算出摘要檔名，並將報告資料夾的摘要檔案更改檔名至計算出來的摘要檔名 16. 利用報告編碼，計算出全文檔名，並將報告資料夾的全文檔案更改檔名至計算出來的全文檔名 17. 從父節點取得節點編碼，透過節點編碼計算出報告列表檔名，將報告資料寫入報告列表檔案 18. 依據報告所使用的解法，找出解法的逆向查詢資料，並將報告編碼加入該資料中，並以編碼順序排序後，將逆向查詢資料寫回檔案 19. 依據報告所使用的技術，找出技術的逆向查詢資料，並將報告編碼加入該資料中，並以編碼順序排序後，將逆向查詢資料寫回檔案 20. 依據報告的作者，找出該作者的逆向查詢資料，將報告編碼加入該資料後寫回檔案 21. 依據報告的報告庫及年代，找出該報告庫某年代的逆向查詢資料，將報告編碼加入該資料後寫回檔案 22. 刪除該報告資料夾 23. 將報告節點指標加入其父節點的報告節點指標陣列之中，並更新父節點的報告數量 24. 將父節點的資料寫入檔案 25. 顯示『已成功加入報告』

- 建議新增領域

Precondition：Committer 瀏覽某一領域類別，欲建議知識管理者在此領域類別新增一子領域

Committer	系統
1. 點選[新增子領域]	
	2. 顯示『子領域名稱』與『簡介』欄位
3. 輸入『名稱』與『簡介』	
	4. 讀取名稱與簡介 5. 檢查目前知識節點是否在第七層，若是則回應『該領域類別位於第七層，無法新增子領域』，並結束 Use Case 6. 檢查目前知識節點的子領域類別是否太多，若是則回應子領域類別過多 7. 從目前知識節點(即該節點的父類別節點)取得知識編碼，並將該知識編碼轉成領域類別路徑 8. 將新領域類別名稱、簡介、所屬領域類別路徑寫入至『建議新增領域』資料夾下中以目前時間爲檔名的檔案 9. Email 通知知識管理者有建議新增領域類別

知識管理者負責維護知識庫的領域類別、議題等資料，知識管理者的使用案例包括：『新增領域類別』、『新增技術議題』、『新增 Survey 議題』、『搬移領域』、『搬移技術議題』、『搬移 Survey 議題』、『分割領域類別』等 7 種使用案例，分別說明如下：

- 新增領域類別

Precondition：知識管理者收到系統通知，要新增某些領域類別，知識管理者使用系統時，會顯示『新增領域』連結

知識庫管理者	系統
1. 點選[新增領域]	
	2. 從『建議新增領域』資料夾中的每一個檔案讀取其路徑、名稱，並顯示之 3. 從『通過審核』資料夾找尋有 add_category 檔案的報告資料夾，並從報告資料夾中的關鍵字檔案讀取建議新增的領域類別路徑與名稱，並顯示之
4. 點選其中一個建議新增領域	

	5. 讀取名稱、簡介、路徑 6. 依據路徑找出父領域類別節點 7. 檢查父領域類別節點的子領域類別是否太多，若是則回應子領域類別過多，結束 Use Case 8. 從父領域類別節點的子領域類別節點指標陣列中，找尋沒有用到的編號，將該編號指派給新加入的子領域類別編碼 9. 建立新的領域類別節點，將名稱、領域類別編碼、父領域類別指標等欄位填妥 10. 將子領域類別資料寫入領域類別資料檔案檔尾，並讀取寫入的紀錄編號，將該編號填入子領域類別節點的資料區塊編號 11. 利用領域類別編碼，計算出領域類別簡介檔名，將簡介資料加入該檔案 12. 將新增的子領域類別節點指標加入父領域類別的子領域類別節點指標陣列，並將父領域類別的子領域節點數量加一 13. 將父領域類別節點資料寫回領域類別檔案中對應的紀錄欄 14. 如果該領域類別透過『通過審核』資料新增，則移除報告關鍵字檔案中對應的領域類別資料中的 new 標籤以及 introduction 標籤，並移除 add_category 檔案 15. 如果該領域類別是透過 Committer『建議新增知識』新增，則移除對應的領域類別資料檔案 16. 回應新增成功

- 新增技術議題

Precondition：知識管理者收到系統通知，要新增某些技術議題，知識管理者使用系統時，會顯示『新增技術議題』連結

知識庫管理者	系統
1. 點選[新增技術議題]	
	2. 從『通過審核』資料夾中的每一個報告資料夾，找尋有 add_issue 檔案的報告資料夾，並從報告資料夾中的關鍵字檔案讀取建議新增的技術議題路徑與名稱，並顯示之
3. 點選其中一個建議新增技術議題	
	4. 讀取名稱、簡介、路徑 5. 依據路徑找出父領域類別節點 6. 檢查父領域類別節點的技術議題是否太多，若是則回應技術議題過多，結束 Use Case 7. 從父領域類別節點的技術議題節點指標陣列中，找尋沒有用到的編號，將該編號指派給新加入的技術議題編碼

	8. 建立新的技術議題節點，將名稱、父領域類別指標、技術議題編號等欄位填妥 9. 將技術議題資料寫入技術議題資料檔案檔尾，讀取紀錄編號，將該編號填入技術議題節點的資料區塊編號欄位 10. 利用技術議題編碼，計算出技術議題簡介檔名，將簡介加入該檔案 11. 將新增的技術議題節點指標加入父領域類別的技術議題節點指標陣列，並將父領域類別的技術議題數量加一 12. 將父領域類別節點資料寫入至檔案 13. 移除報告關鍵字檔案中對應的技術議題中的 new 標籤以及 introduction 標籤，並移除 add_issue 檔案及建立 add.done 檔案 14. 回新增成功

- 新增 Survey 議題：

Precondition：知識管理者收到系統通知，要新增某些 Survey 議題，知識管理者使用系統時，會顯示『新增 Survey 議題』連結

知識庫管理者	系統
1. 點選[新增 Survey 議題]	
	2. 從『通過審核』資料夾中的每一個報告資料夾，找尋有 add_surveyissue 檔案的報告資料夾，並從報告資料夾中的關鍵字檔案讀取建議新增的 Survey 議題的路徑與名稱，並顯示之
3. 點選其中一個建議新增 Survey 議題	
	4. 讀取名稱、簡介、路徑 5. 依據路徑找出父領域類別節點 6. 檢查父領域類別節點的 Survey 議題是否太多，若是則回應 Survey 議題過多，結束 Use Case 7. 從父領域類別節點的 Survey 議題節點指標陣列中，找尋沒有用到的編號，將該編號指派給新加入的 Survey 議題編碼 8. 建立新的 Survey 議題節點，並將名稱、Survey 議題編碼、父領域類別指標等欄位填妥 9. 將 Survey 議題資料寫入 Survey 議題資料檔案檔尾，並讀取該紀錄編號，將該編號填入 Survey 議題節點的資料區塊編號 10. 利用 Survey 議題編碼，計算出 Survey 議題簡介檔名，將簡介加入該檔案

	11. 將新增的 Survey 議題節點指標加入父領域類別的 Survey 議題節點指標陣列，並將父領域類別的 Survey 議題數量加一 12. 將父領域類別節點資料寫入至檔案 13. 移除報告關鍵字檔案中對應的 Survey 議題中的 new 標籤以及 introduction 標籤，並移除 add_surveyissue 與建立 add.done 檔案 14. 回應新增成功

- 搬移領域類別

Precondition：知識管理者已經瀏覽到一個領域類別節點，欲將該領域類別搬移到其他領域類別之下

知識庫管理者	系統
1. 點選[搬移領域]	
	2. 讀入『要搬移的領域類別編碼』 3. 列出該領域類別的第一層領域類別、第二層領域類別直到其父領域類別，讓管理者選擇『舊領域與新領域的共同祖先』
4. 選擇其中一個領域	
	5. 從領域類別編碼，找出該領域類別下的子領域類別列表 6. 列出子領域類別列表
7. 重複步驟 4~6，直到管理者點選『搬移領域』按鈕	
	8. 讀入『搬移目的領域類別編碼』 9. 依據搬移目的領域類別編碼，找出搬移目的領域類別節點 10. 檢查搬移目的領域類別的子領域類別是否太多，若是則回應子領域類別過多，並結束 Use Case 11. 依據要搬移的領域類別編碼，取得要搬移領域類別節點 12. 檢查要搬移領域類別節點的深度是否小於搬移目的領域類別節點的深度，若是則回應不能往比該領域類別層級還深的領域搬移，並結束 Use Case 13. 將搬移領域類別節點從其父領域節點的子領域類別列表移除，將父節點子領域類別數量減一，並將該節點資料寫入領域類別檔案中對應的紀錄欄 14. 從搬移目的領域類別的子領域類別節點指標陣列中，找尋沒有用到的編號，將該編號指派給搬移的領域類別編碼 15. 修改搬移領域類別節點的編碼、父節點指標 16. 將原來領域類別節點下的知識節點，執行下列動作直到步驟 19 爲止

	17. 修改知識節點的知識編碼，並將節點資料寫入對應的知識資料檔案中的紀錄欄 18. 如果該知識是報告，則執行下列步驟 甲、 依據報告的作者找出其逆向查詢資料，並將對應的報告編碼更新後寫入 乙、 依據報告庫、年代找出其逆向查詢資料，並將對應的報告編碼更新後寫入 丙、 依據報告的解法找出逆向查詢資料，並將對應的報告編碼更新後寫入 丁、 依據報告的技術指出逆向查詢資料，並將對應的報告編碼更新後寫入 19. 若該知識節點有子知識，針對其子知識重複執行步驟 17、18 20. 將搬移領域節點指標加入搬移目的領域類別的子領域節點指標陣列 21. 將搬移目的領域類別節點的子領域數量加一，並將該節點資料寫入領域類別檔案中對應的紀錄欄 22. 回應搬移成功

- 搬移技術議題

Precondition：知識管理者已經瀏覽到最末端的領域類別節點，畫面上列出其技術議題列表，欲將其中一個技術議題搬移到其他領域類別之下

知識庫管理者	系統
1. 點選 [搬移技術議題]	
	2. 將顯示技術議題供管理者點選要搬移的技術議題
3. 點選要搬移的技術議題	
	4. 讀入要搬移的技術議題編碼 5. 列出該領域類別的第一層領域類別、第二層領域類別直到其父領域類別，讓管理者選擇『舊領域與新領域的共同祖先』
6. 選擇其中一個領域	
	7. 從該領域類別編碼，找出該領域類別下的子領域類別列表 8. 列出子領域類別列表
9. 重複 6~8，直至管理者點選『搬移按鈕』	
	10. 讀入『搬移目的領域類別編碼』 11. 依據搬移目的領域類別編碼，找出搬移目的領域類別節點

	12. 檢查搬移目的領域類別是否在最末端，若否則回應『非末端領域類別節點』，並結束 Use Case 13. 檢查搬移目的領域類別的技術議題是否太多，若是則回應技術議題過多，並結束 Use Case 14. 依據技術議題編碼，取得要搬移技術議題節點 15. 將搬移技術議題節點從其父領域類別節點的技術議題陣列移除，將父節點技術議題數量減一，並將該節點資料寫入領域類別資料檔案對應的紀錄欄 16. 從搬移目的領域類別的技術議題節點指標陣列中，找尋沒有用到的編號，將該編號指派給搬移的技術議題編碼 17. 更新搬移技術議題節點的知識編碼與父類別節點指標 18. 將原來技術議題節點下的知識(包含解法、與技術報告)，執行下列動作直到步驟 21 爲止 19. 修改知識節點的知識編碼，並將節點資料寫入對應的知識資料檔案中的紀錄欄 20. 如果該知識是報告，則執行下列步驟 ● 依據報告的作者找出其逆向查詢資料，並將對應的報告編碼更新後寫入 ● 依據報告庫、年代找出其逆向查詢資料，並將對應的報告編碼更新後寫入 ● 依據報告的解法找出逆向查詢資料，並將對應的報告編碼更新後寫入 ● 依據報告的技術指出逆向查詢資料，並將對應的報告編碼更新後寫入 21. 若該知識節點爲解法，重複執行步驟 19、20 繼續更新該解法下的報告 22. 將搬移技術議題節點資料寫入技術議題資料檔案對應的紀錄欄 23. 將搬移技術議題節點指標加入目的領域類別的技術議題節點指標陣列，將搬移目的領域類別的技術議題數量加一，並將該節點資料寫入領域類別資料檔案對應的紀錄欄 24. 回應搬移成功

- 搬移 Survey 議題

Precondition：知識管理者已經瀏覽到領域類別節點，畫面上列出其 Survey 議題列表，欲將其中一個 Survey 議題搬移到其他領域類別之下

知識庫管理者	系統
1. 點選[搬移 Survey 議題]	

	2. 將顯示 Survey 議題供管理者點選要搬移的 Survey 議題
3. 點選要搬移的 Survey 議題	
	4. 讀入搬移的技術議題編碼 5. 列出該領域類別的第一層領域類別、第二層領域類別…一直到其父領域類別，讓管理者選擇『舊領域與新領域的共同祖先』
6. 選擇其中一個領域	
	7. 從該領域類別編碼，找出該領域類別下的子領域類別列表 8. 列出子領域類別列表
9. 重複 6~9，直至管理者點選『搬移按鈕』	
	10. 讀入搬移目的領域類別編碼 11. 依據搬移目的領域類別編碼，找出搬移目的領域類別節點 12. 檢查搬移目的領域類別節點的 Survey 議題是否太多，若是則回應 Survey 議題過多，並結束 Use Case 13. 依據 Survey 議題編碼，取得要搬移 Survey 議題節點 14. 將搬移 Survey 議題節點從其父領域類別節點的 Survey 議題陣列中移除，將父節點 Survey 議題數量減一並將領域類別資料檔案對應的紀錄欄 15. 從搬移目的領域類別的 Survey 議題節點指標陣列中，找尋沒有用到的編號，將該編號指派給搬移的 Survey 議題編碼 16. 更新搬移 Survey 議題節點的知識編碼與父領域類別編碼 17. 將原來 Survey 議題節點下的所有報告，執行下列動作直到步驟 21 爲止 18. 修改知識節點的知識編碼，並將節點資料寫入對應的知識資料檔案中的紀錄欄 19. 依據報告的作者找出其逆向查詢資料，並將對應的報告編碼更新後寫入 20. 依據報告庫、年代找出其逆向查詢資料，並將對應的報告編碼更新後寫入 21. 依據報告的解法找出逆向查詢資料，並將對應的報告編碼更新後寫入 22. 將搬移 Survey 議題節點資料寫入 Survey 議題資料檔案對應的紀錄欄 23. 將新增的 Survey 議題節點指標加入目的領域類別的 Survey 議題節點指標陣列，搬移目的領域類別的 Survey 議題數量加一，並將該節點資料寫入領域類別資料檔案的對應的紀錄欄 24. 回應搬移成功

- 分割領域類別

這個使用案例應用於末端領域類別技術太多，因此需作分割放置在二個以上之領域類別，但領域類別下中子領域與技術議題不能共存，因此需新增數個子領域，再將其技術議題分配到這些子領域中。

Precondition：知識庫管理者瀏覽至末端領域類別，發現其技術議題太多，欲將此領域類別分成數個小領域

圖 4-7 分割領域類別介面 1

知識庫管理者	系統
1. 點選[分割類別]	
	2. 顯示如圖 4-7 的介面
3. 填入要新增的子領域類別名稱與簡介	
	4. 讀入每一個要新增的子領域類別名稱、簡介 5. 檢查目前知識節點是否在第七層，若是則回應無法分割領域類別，並結束 Use Case 6. 建立新的父領域類別，從目前知識節點複製名稱、簡介區塊編號、資料區塊編號、父領域類別節點指標，將型態設爲『有子領域的領域類別』，配置子領域類別列表 7. 將目前知識節點從其父領域類別的子領域類別列表中移除 8. 對於每一個新增的子領域類別，執行下列動作直至步驟 13 爲止 9. 取得領域類別編碼，該編碼從 1 開始編碼 10. 建立新的領域類別節點，將名稱、父領域類別指標、領域類別編碼填入 11. 將子領域類別簡介寫入領域類別簡介檔案檔尾，並讀取該紀錄編號，將該紀錄編號填入節點的簡介編號欄位 12. 將子領域類別資料寫入領域類別資料檔案檔尾，並讀取該紀錄編號，將該編號填入子領域類別節點的資料區塊編號

	13. 將新增的子領域類別節點指標加入父領域類別的子領域類別節點指標陣列，並將父領域類別的子領域節點數量加一 14. 將父領域類別節點資料寫入至領域類別資料檔案對應的紀錄欄 15. 從目前知識節點讀取其技術議題列表與 Survey 議題列表，並詢問每一個 Survey/技術議題要搬移到哪一個子領域中
16. 爲每一個議題選擇要搬移的子領域類別目的	
	17. 讀入每個要搬移的議題編碼與其搬移目的 18. 從目前知識節點讀取 Survey 議題列表與技術議題列表，針對每一個要搬移的議題節點，依序執行下列步驟直至步驟 28 爲止 19. 依據編碼取得議題節點 20. 將議題節點從其父領域類別節點的議題列表移除 21. 根據搬移目的領域類別編碼，取得搬移目的領域類別節點 22. 從搬移目的領域類別節點的議題列表取得新的議題編碼 23. 更新議題節點的知識編碼與父類別節點指標 24. 將議題節點加入搬移目的領域類別節點的議題列表，並將搬移目的領域類別節點的議題個數加一 25. 針對議題下的每一個知識，執行下列步驟直至步驟 27 爲止 26. 更新知識編碼，並知識節點資料寫入資料檔案 27. 如果該知識是報告，則執行下列步驟 甲、 依據報告的作者找出其逆向查詢資料，並將對應的報告編碼更新後寫入 乙、 依據報告庫、年代找出其逆向查詢資料，並將對應的報告編碼更新後寫入 丙、 依據報告的解法找出逆向查詢資料，並將對應的報告編碼更新後寫入 丁、 依據報告的技術指出逆向查詢資料，並將對應的報告編碼更新後寫入 28. 將議題資料寫入對應的議題資料檔案紀錄欄 29. 將每一個分割的子領域類別資料寫入領域類別資料檔案中對應的紀錄欄 30. 將新的父領域類別加入其父領域類別的子領域類別列表中 31. 將新的父領域類別資料寫入檔案 32. 將目前知識節點更新到新的父領域類別節點 33. 詢問是否要更新領域類別的名稱與簡介

34. 若不修改，則跳至步驟 44	
	35. 透過目前知識節點取得知識編號，依據該編號計算出簡介檔名，從該檔案讀取簡介 36. 顯示名稱與簡介
37. 修改名稱與簡介	
	38. 修改目前知識節點的名稱 39. 將目前知識節點資料寫入領域類別檔案 40. 將目前知識節點簡介寫入簡介檔案 41. 回應分割成功

系統架構

本系統採用三層式架構，使用 Tomcat[15]當作 Web Server，以 JSP（Java Server Page）、Java Servlet 處理使用者的請求，知識庫伺服器(DB Server)以 Java 實作，使用者資料庫採用 MySQL[16]當作資料庫伺服器，由於技術報告庫資料量不大，因此將這些伺服器裝置在同一台電腦中。

如圖 4-8 所示，本系統分為『身分認證模組』、『查詢模組』、『管理知識模組』、『新增知識模組』、『記憶體快取模組』、『關鍵字列表模組』等 6 個模組，『身分認證模組』負責驗證使用者的帳號與密碼，並且賦予其特定功能的使用權利；查詢模組負責處理一般使用者的查詢，管理知識模組負責新增、搬移、分割知識，新增知識模組負責處理 Submitter 提交的報告，以及提供 Committer 審核報告的功能。『新增知識暫存區』則是用來暫時存放 Submitter 提交過來的報告資料以及建議新增的領域類別、技術議題、Survey 議題等資料。

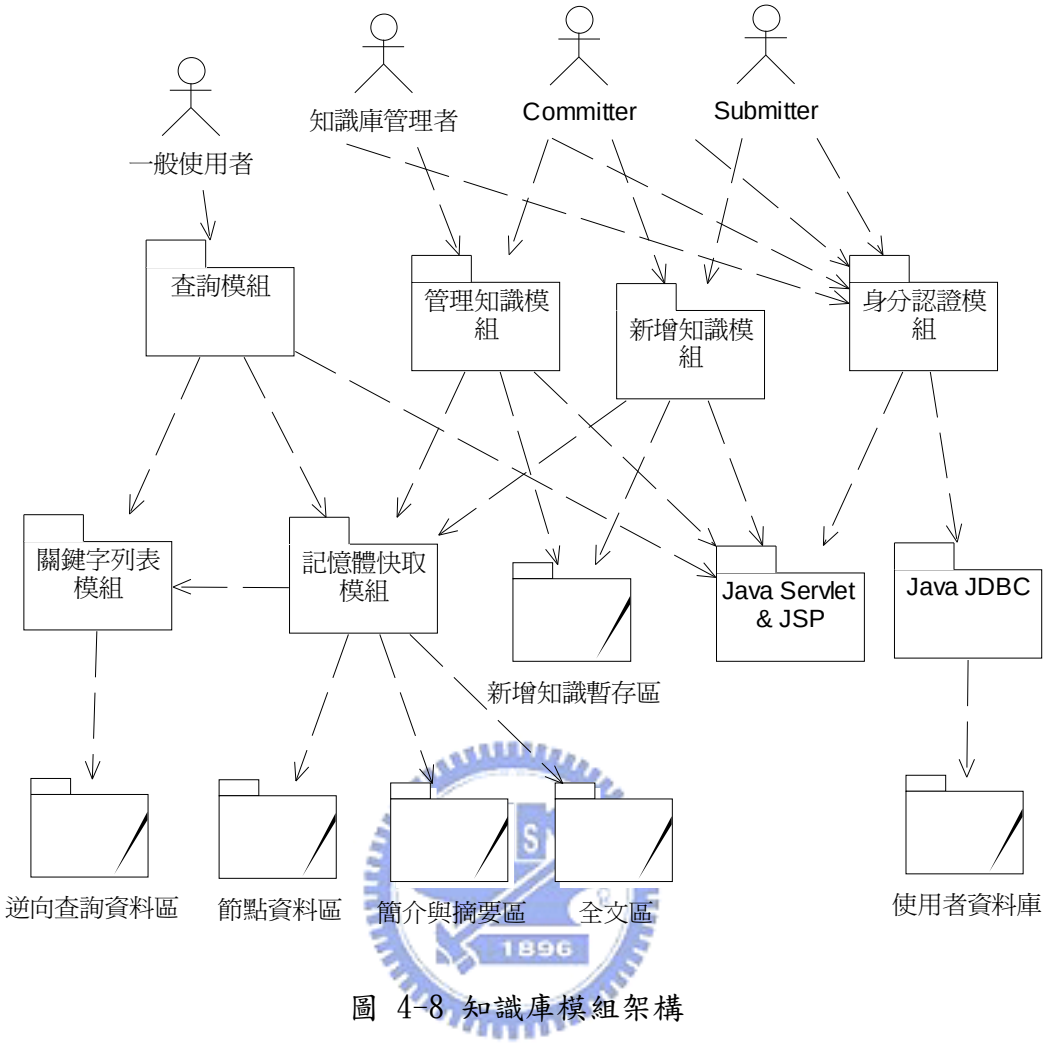


圖 4-8 知識庫模組架構

為了加快查詢速度，因此將領域類別、技術議題、解法、Survey 議題等節點資料儲存在記憶體之中，此部分為『記憶體快取模組』，而節點資料備份在硬碟中的資料為『節點資料區』，節點資料區除了上述的節點資料外，還儲存了技術報告節點、Survey 報告節點的資料。知識的簡介資料與報告的摘要資料儲存在『簡介與摘要區』，報告的全文資料儲存在『報告全文區』。報告的逆向查詢資料，如技術、解法、作者、年代等，儲存在『逆向查詢資料區』。如第三章所述，知識庫系統有『技術關鍵字列表』、『解法關鍵字列表』，這一部分的功能在『關鍵字列表模組』，關鍵字列表的資料會關聯到逆向查詢資料的技術、解法的逆向查詢資料。『使用者資料庫』儲存使用者的帳號、密碼、Email、身分(一般使用者、知識庫管理者、Submitter、Committer)等。

在管理者作『搬移領域類別』、『搬移技術議題』、『搬移 Survey 議題』、『分割領域類別』等動作時，知識庫會更動許多資料，如果系統在更動資料時發生當機，則知識庫會處於不一致的狀態，為了解決此問題，乃採用 Dual-System 的做法，將這些資料同時儲存在 A 與 B 系統之中，如果是一般的新增，則 A、B 同時更新，若是搬移或分割，則將其中一個先做搬移或分割的動作，另一個用來服務，當搬移分割完成後，在交換過來。當系統發生當機時，可由另一個完整的系統將資料重新更新，因此不會有不一致的問題。本論文不深入探討此問題。

4.2節 伺服器端之資料結構

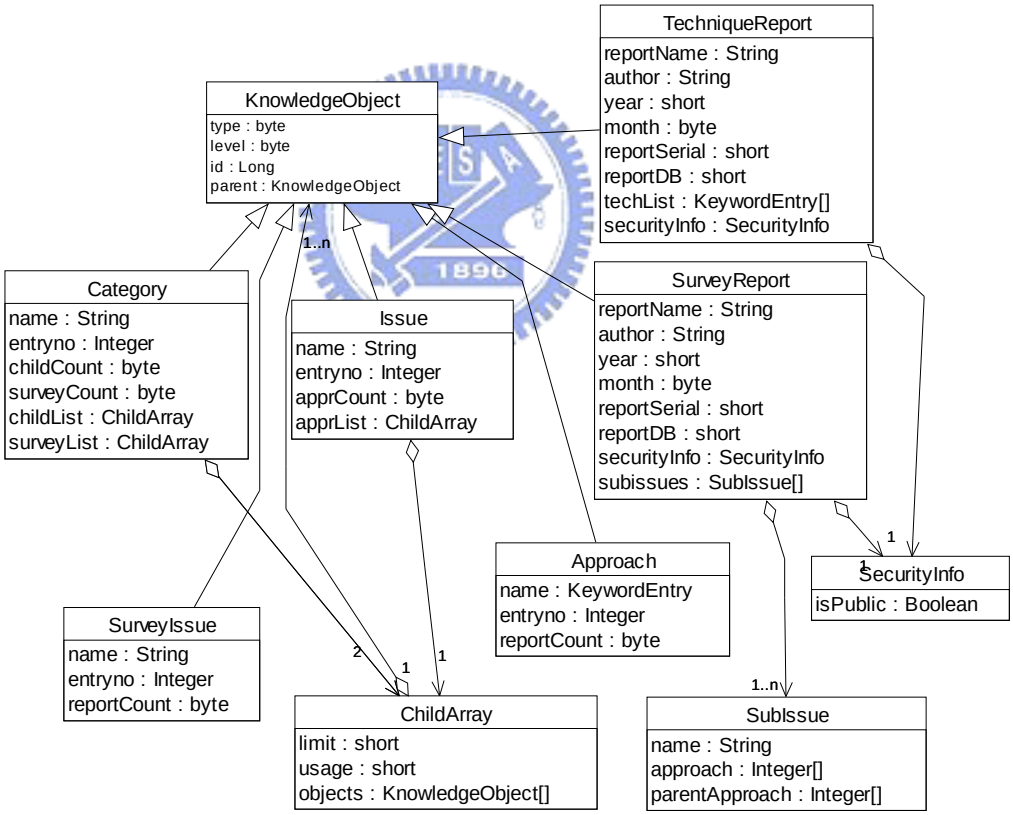


圖 4-9 記憶體快取模組類別圖

圖 4-9 為記憶體快取模組的類別圖，KnowledgeObject 為各知識節點的共同父類別(Parent Class)，所有的知識物件類別都繼承自此類別，技術報告庫的知

識物件類別有 Category、Issue、SurveyIssue、Approach、TechniqueReport、SurveyReport 等 6 種類別，分別代表領域類別、技術議題、Survey 議題、解法、技術報告、Survey 報告等知識節點，其中 Category、Issue 類別有 ChildArray 類別的物件以紀錄儲存節點下的子知識節點指標及處理子節點指標陣列的各種運算。ChildArray 的 limit 紀錄該陣列的最大長度，usage 紀錄該陣列有多少知識節點指標，objects 為儲存子知識節點指標的陣列，ChildArray 的 Set(I, KnowledgeObject)方法負責設定第 I 個子知識節點的節點指標，如果 I 之值超過 limit 的值，則將陣列大小加倍，Get(I)方法負責取得第 I 個子知識節點指標，Append(KnowledgeObject) 負責將知識物件加到陣列第一個空的地方，Remove(I)則為將第 I 個知識節點指標從陣列中移除，getNextFreeIndex()負責取得陣列第一個空的位置。

KnowledgeObject 的 type 為該知識節點物件的型別，level 為該知識節點在第幾層，id 為其知識編碼，parent 為該知識物件的父知識節點指標，KnowledgeObject 是一個抽象類別，其大部分的方法都沒有實作，待其繼承類別(如 Category、Issue、SurveyIssue、Approach、TechniqueReport、SurveyReport 等類別)實作，以下是 KnowledgeObject 的方法說明：

1. writeData()：將知識的基本資料寫入檔案
2. writeIntro(intro)：將知識的簡介資料(或者是摘要)寫入檔案
3. getIntro()：取得該知識的簡介資料
4. getChildTypes()：取得該知識節點下有哪些子知識節點型態
5. getChildList(Type)：取得該知識節點下，某個型態的子知識節點，與 getChildTypes 合作，可以取得該知識下所有的子知識。
6. checkPermission(User)：檢查 User 是否可閱讀該知識，KnowledgeObject 預設允許所有人讀取知識，而 SurveyReport、TechniqueReport 會修改此方法，以檢查該報告是否可以給使用者讀

取。TechniqueReport、SurveyReport 有 SecurityInfo 物件，該物件儲存著該報告是否公開等資訊，若需要作其他檢查(如報告的可閱讀使用者列表)，可修改 SeucrityInfo 類別的 checkRead(User)方法，以及增加 SecurityInfo 類別的屬性。

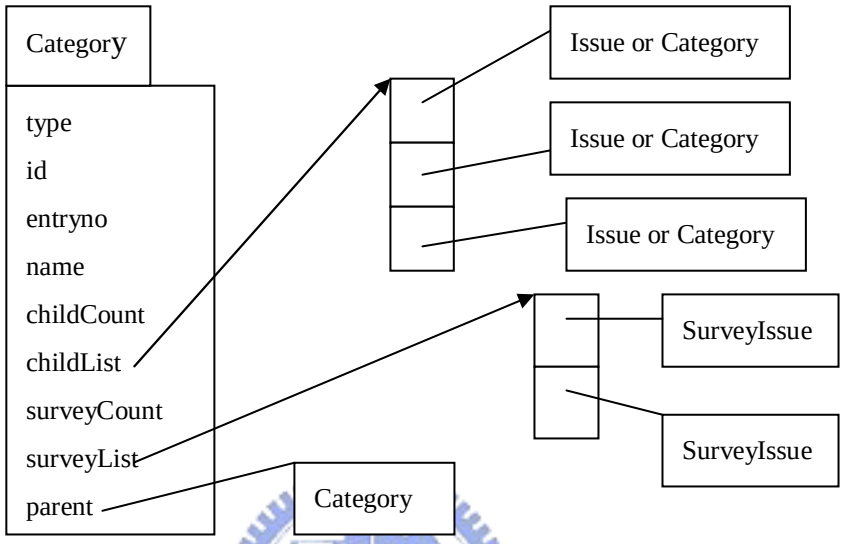


圖 4-10 領域類別節點記憶體儲存資料示意圖

圖 4-10 為領域類別節點(Category)在記憶體中的儲存資料，記憶體中的儲存資料介紹如下：

1. type 為該領域類別的型態，可分為『有技術議題』與『無技術議題』兩種領域類別
2. id 為領域類別的知識編碼
3. entryno 儲存該節點對應檔案之紀錄(Record)序號，所有的領域類別檔案儲存在同一個檔案中，當該領域類別的資料有變動時(如增加子領域/技術議題、Survey 議題等)，可依據 entryno 將資料寫入對應的紀錄欄
4. name 為領域類別的名稱
5. childCount 為子領域或技術議題個數
6. childList 為子領域或技術議題列表，視領域類別的型別而定，如果為『有議題的領域類別』，則 childList 陣列為技術議題節點陣列，如果

是『無議題的領域類別』，則為領域類別節點陣列。以 ChildArray 類別實作

7. surveyCount 為 Survey 議題個數
8. SurveyList 為 Survey 議題的列表，儲存資料為 Survey 議題節點指標。以 ChildArray 類別實作
9. parent 為父領域類別的物件指標，當逆向查詢、搬移領域類別時，會需要父領域類別指標。

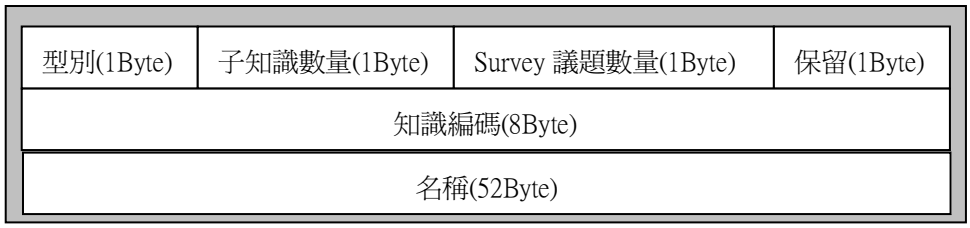


圖 4-11 領域類別節點儲存在硬碟的資料格式

圖 4-11 為領域類別儲存在硬碟中的資料，型別以 1 個 Byte 儲存，子知識數量(技術議題/子領域類別)以 1Byte 儲存，Survey 議題數量以 1 個 Byte 儲存，根據第三章介紹，一個領域類別之下最多 63 個技術議題/子領域，63 個 Survey 議題，因此以 1Byte 表示已經足夠。知識編碼以 8Byte 表示。領域類別名稱長度 52Byte，由於英文單字大多數長度都在 10 左右，以 52Byte 表示，則可以有 4~5 個英文單字，此足以表示領域類別名稱。領域類別節點在記憶體中的資料，有數個欄位並未儲存在硬碟資料中，分別是 entryno、childList、surveyList、parent，entryno 為該節點資料儲存在檔案中的對應編號，此欄位在讀取檔案時便可得知，並不需要特別在檔案中紀錄，childList、surveyList 為子領域/技術議題、Survey 議題的節點指標陣列，在之後繼續讀取技術議題、Survey 議題時會將其關係連結起來，因此不需紀錄在檔案中，而 parent 為該領域節點的父類別指標，可經由 id 找尋其父領域類別節點，故不需紀錄在檔案中。

由於領域類別資料是一直附加在檔案末端，因此對於每一個領域類別而言，

其父領域類別的資料一定在該領域類別資料之前，從檔案開頭開始讀取，每一次讀取一個區塊，依據所讀取的資料建立領域類別物件，並根據該領域類別的知識編碼，找出其父領域類別，如圖 4-12 所示，假設該類別編碼為 2-3-4-0-0-0-0-0-0-0，由於前面的領域類別的資料已經讀取，因此之前的領域類別資料已經在領域類別架構樹中，可藉由從根節點(root)開始，找到該領域類別的父領域類別節點，並建立連結關係，如此便可以將技術報告庫的領域類別架構樹建立起來，提供使用者查詢。

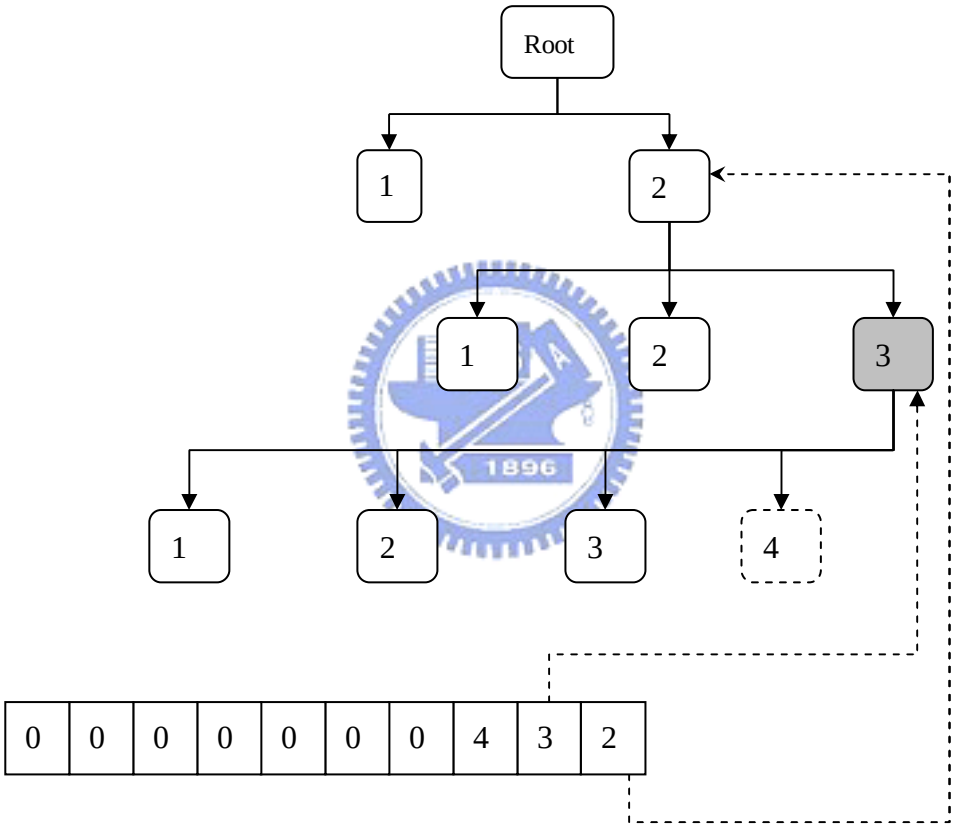


圖 4-12 利用知識編碼建立領域類別節點之間的連結

圖 4-13 是技術議題節點在記憶體中儲存資料的示意圖，id、entryno、name 等欄位與領域類別節點相同，type 為知識節點的型別，其值為『技術議題』，apprCount 為解法個數，apprList 為解法節點指標陣列，以 ChildArray 實作，parent 為該技術議題的父領域類類別節點指標。圖 4-14 則為其儲存在硬碟中的格式，如同領域類別節點，系統啟動時循序讀取技術議題檔案，以類似建立領域

類別樹的方式將技術議題加到領域類別樹狀圖中。

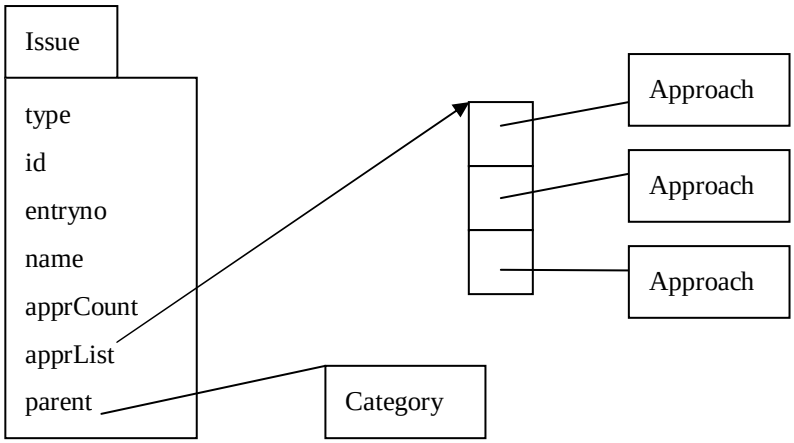


圖 4-13 技術議題節點儲存資料示意圖

圖 4-14 技術議題儲存在硬碟中的格式

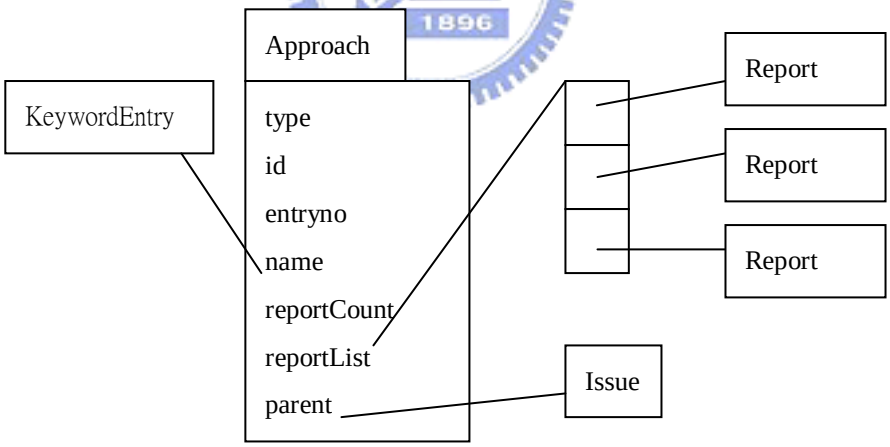


圖 4-15 解法節點儲存資料示意圖

圖 4-15 為解法節點在記憶體中的儲存示意圖，id、entryno 欄位與領域類別相同，type 為節點型別，其值為『解法』，name 紀錄此解法名稱的關鍵字列表編碼，如第三章所述，技術報告庫的解法通常有許多是共用的，建立解法關鍵字列表，因此只需要紀錄其關鍵字編碼即可，可透過該編碼找到對應的關鍵字 Entry，關鍵字 Entry 以 KeywordEntry 類別實作，其細節之後會談到；reportCount

負責紀錄該解法目前的報告數量，parent 為該解法的技術議題節點指標。至於解法的報告列表，由於報告數量龐大，加上只有在列舉解法下的報告時才會用到，因此當使用者點選到解法時，透過解法的知識編碼計算出報告列表檔名，讀取報告列表檔案以取得報告的名稱、作者、年代等資料，並建立 reportList，當使用者離開解法節點時，便將該 reportList 歸還。圖 4-16 為解法儲存在檔案中的格式。

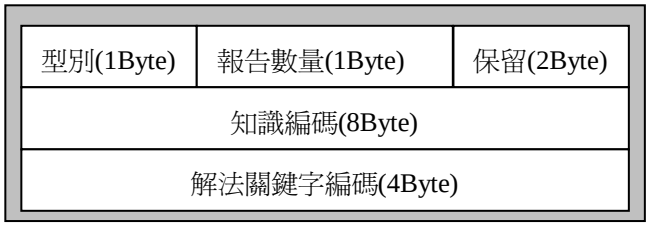


圖 4-16 解法節點儲存在硬碟中的格式

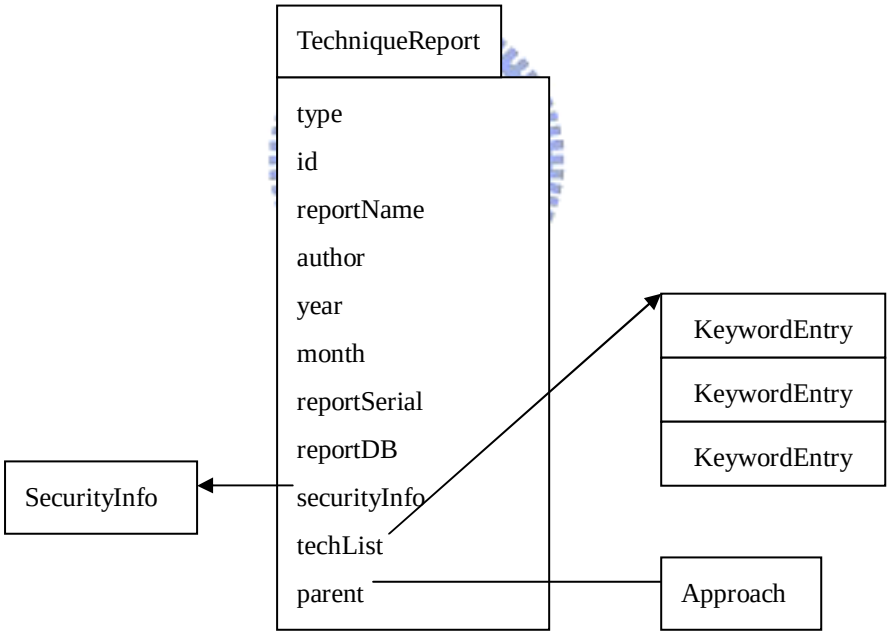


圖 4-17 技術報告節點儲存資料示意圖

圖 4-17 為技術報告的儲存在記憶體中的示意圖，type 為知識節點型態，其值為『技術報告』，id 為技術報告的知識編碼，reportName 為報告名稱，author 為作者名稱，year 代表年代，month 代表月份，reportSerial 為該報告在該年度的序號，reportDB 為報告庫的代號，securityInfo 除存著該報告的權限設定，以 SecurityInfo 類別實作，techList 為該報告所使用的技術列表，為

KeywordEntry 物件指標陣列，指向技術關鍵字列表中的 KeywordEntry，parent 為該技術議題的解法節點指標。

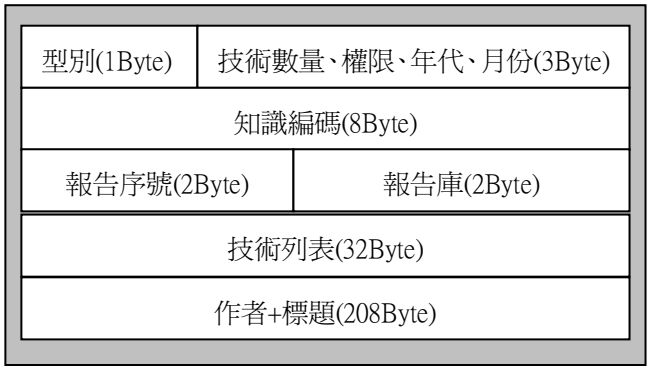


圖 4-18 技術報告儲存在硬碟中的格式

圖 4-18 為技術報告儲存在硬碟中的格式，其內容包括：

1. 型別為 1Byte
2. 權限設定以 1Bit 儲存，在技術報告庫中，只需要一個 Bit 表示該技術報告是否開放即可。技術數量以 4Bit 儲存，月份以 4Bit 儲存，年代以 12Bit 儲存，由於年代都是四位數，以 12Bit 儲存已經足夠。共佔 21Bit，故以 3Byte 儲存。
3. 報告的知識編碼以 8Byte 儲存
4. 報告序碼為技術報告庫給予的序號，一般技術報告庫的編號通常是『報告庫+年代+序號』，如 TR2000-381 這類的編號，因此以 2Byte 儲存該序號，然後以報告庫、年代、序號欄位當作整篇報告的編號。像 NASA 這種大型組織，其技術報告庫有很多個，即可使用這個欄位以分辨該報告從哪一個報告庫中出來。
5. 如第三章所述，技術報告庫的技術有許多會重複，因此建立技術的關鍵字列表，技術報告所用到的技術，只需要紀錄該關鍵字的編碼即可(每一個關鍵字編碼佔 4Byte，如第三章所述，一篇報告的技術最多 8 個，以 32Byte 儲存技術列表)。
6. 標題、作者名稱長度 208Byte，首先儲存作者，作者以逗號分隔，最後

加一個 0，接著儲存標題。

圖 4-19 是 Survey 議題節點在記憶體中儲存資料的示意圖，id、entryno、name、parent 與技術議題相同，type 為知識節點型別，其值為『Survey 議題』，reportCount 儲存其 Survey 報告個數，如同解法節點，Survey 議題下的報告列表只有當使用者點選到 Survey 議題節點時，透過議題節點的知識編碼計算出報告列表檔名，讀取報告列表檔案以取得報告的名稱、作者、年代等資料，並建立 reportList，當使用者離開議題節點時，便將該 reportList 歸還。圖 4-20 為 Survey 議題儲存在硬碟中的格式，如同領域類別節點，所有的 Survey 議題都儲存在同一個檔案，在系統啟動時循序讀取技術議題檔案，以類似建立領域類別樹的方式將技術議題加到領域類別樹狀圖中。

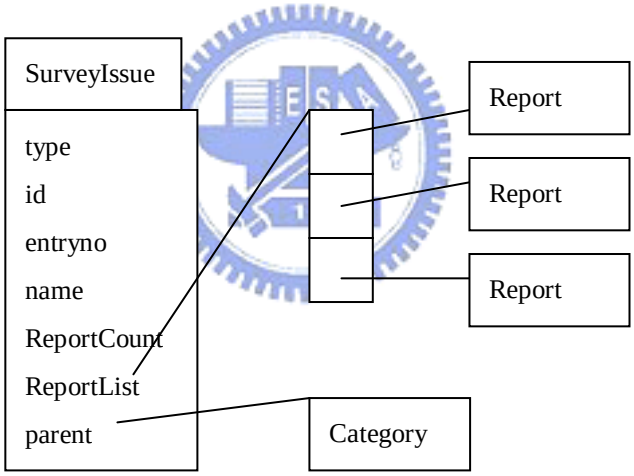


圖 4-19 Survey 議題節點儲存資料示意圖

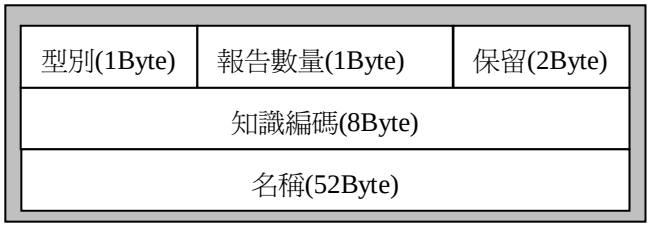


圖 4-20 Survey 議題儲存在硬碟中的格式

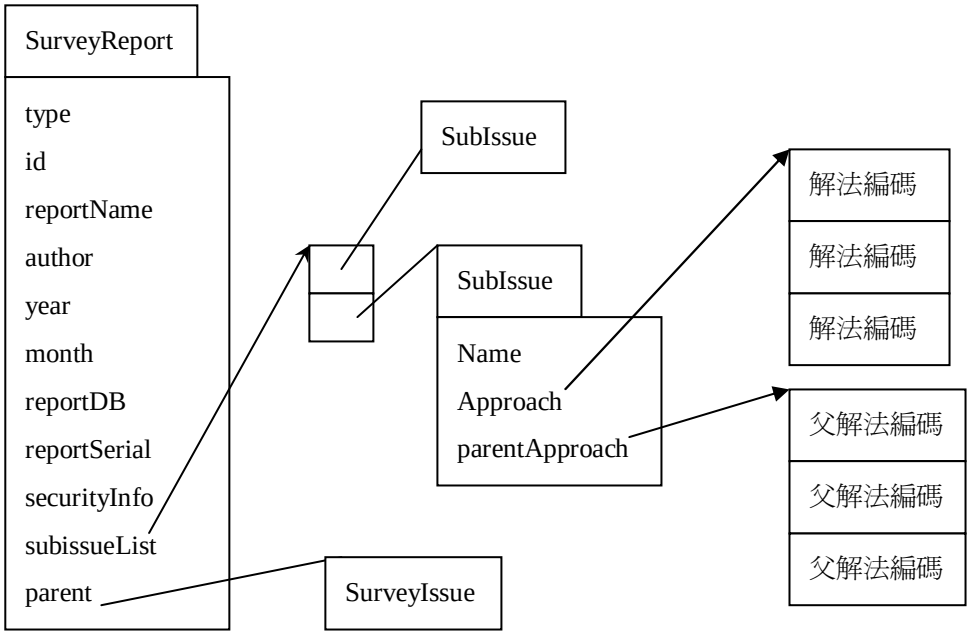


圖 4-21 Survey 報告節點儲存資料示意圖

圖 4-21 為 Survey 報告的儲存在記憶體中的示意圖，id、reportName、author、year、month、reportDB、reportSerial、securityInfo 等資料與技術報告節點相同，type 為知識節點型別，以 Survey 報告而言，其值為『Surve 報告』，subissueList 代表其子議題列表，為一 SubIssue 物件陣列，parent 為 Survey 報告的 Survey 議題節點指標。圖 4-22 為其儲存在硬碟中的格式。Survey 報告的子議題及其下的解法樹狀圖的資料儲存在子議題列表檔案中，子議題列表檔案格式如圖 4-23 所示，子議題以換行(newline)相隔，緊接著 0，然後是子議題之下解法的樹狀圖儲存資料，子議題編碼以 1Byte 儲存，解法及其父解法則以關鍵字列表的編碼儲存。

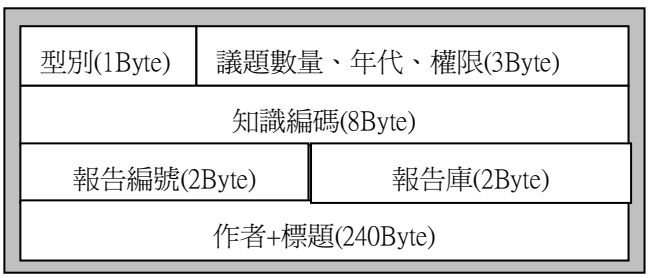


圖 4-22 Survey 報告儲存在硬碟中的格式

```
子議題名稱\n
子議題名稱\n
子議題名稱 0
子議題編碼 解法編碼 父解法編碼
0 1 0
0 2 0
0 3 1
…
```

圖 4-23 子議題列表檔案格式

圖 4-24 為關鍵字列表儲存的資料，每一個關鍵字的儲存資料為關鍵字編碼(4Byte，從 1 開始累加)、名稱(60Byte)。每一個關鍵字的逆向查詢資料，依據第一層領域類別編碼及其關鍵字前四個字元分配，如 Dynamic Programming 技術關鍵字在領域 10 底下的逆向查詢資料儲存在『逆向查詢/10/D/Y/N/A』底下的檔案 Dynamic Programming_T 檔案之中(_T 代表技術關鍵字，_A 代表解法關鍵字)，逆向查詢檔案中儲存的資料為報告節點的知識編碼。

圖 4-25 為關鍵字列表模組的類別圖，KeywordTable 類別的 array 以 Java 的 Vector 物件實作，負責儲存關鍵字編碼與關鍵字名稱的對應，其索引為關鍵字編碼，值為 KeywordEntry 物件，KeywordEntry 的 id 為關鍵字編碼，name 為關鍵字名稱，若要查詢編碼與關鍵字的對應，則呼叫 KeywordTable 的 getEntryById(id)方法，該方法會以 id 為 array 的索引，取出 KeywordEntry，然後從 KeywordEntry 取出 name 即得到關鍵字名稱。KeywordTable 類別的 radixTree 負責處理關鍵字名稱反查編碼的部分，radixTree 以 RadixTree 類別實作，該類別屬性 root 為 RadixTree 的根節點，以 RadixTreeNode 類別實作，該類別有兩個屬性：type 與 childnodes，childnodes 為一陣列，該陣列大小為 26，用來表示 A~Z，type 用來指定該 Node 是否為中間節點，若是中間節點，則 childnodes 為一 RadixTreeNode 節點陣列，用來指向其下一層 RadixTreeNode

指標，若 type 為末端節點，則 childnodes 為儲存 Hashtablee 的陣列，該 Hashtable 以關鍵字名稱(name)當作 key，KeywordEntry 當作 value。當使用者利用關鍵字作逆向查詢時，系統先呼叫 KeywordTable 的 getEntryByName(name)方法，該方法會將 name 轉交給 RadixTree 的 get(name)方法，RadixTree 利用 name 的前幾個字先作 Radix Search，到了最底端的 Hashtable 時，以 name 當作索引找出對應的 KeywordEntry。

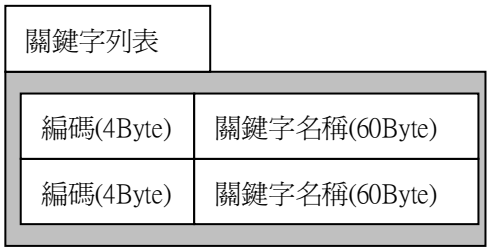


圖 4-24 關鍵字列表與逆向查詢資料

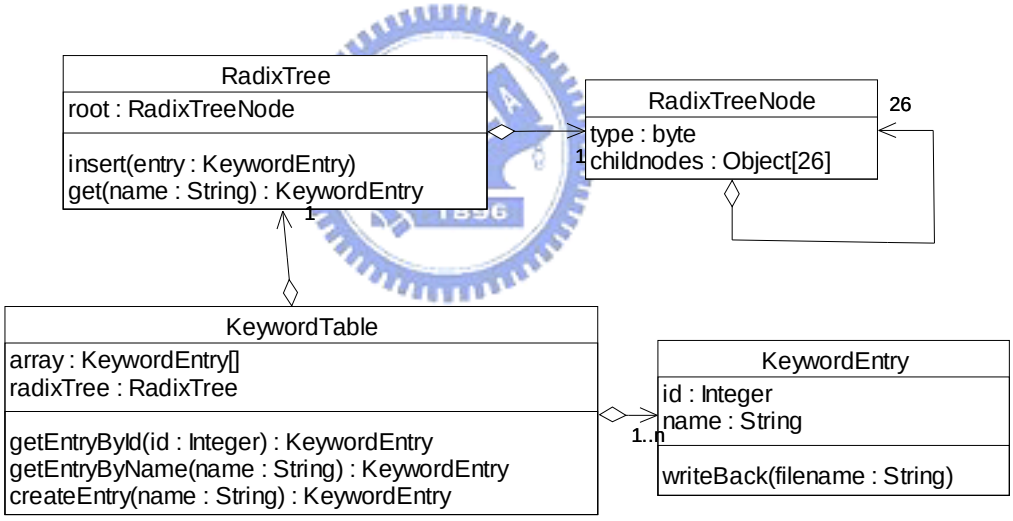


圖 4-25 關鍵字列表模組類別圖

作者的逆向查詢資料儲存方式為作者的前四個英文字母(只抓取英文字母)當作資料夾，舉例來說，T. Doeppner 的報告列表儲存在『作者/T/D/O/E/T. Doeppner』檔案中，檔案儲存的資料為報告編碼。報告庫的某年代的報告列表的儲存方式為以『報告庫』名稱為資料夾，在『報告庫』的資料夾中以年代為名稱的檔案儲存該報告庫在該年代的報告列表，儲存資料為報告編碼。

如前所述，技術報告庫有 12500 個領域類別，其中最下層有 10000 個領域類別，每一個領域類別下有 10 個技術議題、5 個 Survey 議題，每一技術議題下有 5 個解法，每一解法下有 5 篇報告，每一 Survey 議題下有 5 篇報告，系統為了加快查詢速度，因此將領域類別、技術議題、Survey 議題、解法等節點資料儲存在記憶體中，領域類別基本資料大小為 36byte(從 KnowledgeObject 繼承 16byte：type(1)、level(1)、id(8)、parent(4)，name 4byte，entryno 4Byte，childCount 1Byte、surveyCount 1Byte，childList 4Byte，surveyList 4Byte，共 34Byte，但為了對齊記憶體邊界，需 36Byte)，假設上層領域類別下平均有 5 個子領域類別，故上層領域類別的子領域類別列表需花費 28byte(usage(2)、limit(2)、objects(4)，5 個領域類別節點指標共需 20byte)儲存，因此上層領域類別佔 64*2500=160KB，而下層領域類別的技術議題列表需花費 48Byte，Survey 議題列表需花費 28Byte，因此下層領域類別佔 112*10000=1.12MB，領域類別名稱則佔 52*12500=650KB 記憶體，因此領域類別節點資料大約需花費 2MB 記憶體。

技術議題節點基本資料需 32byte(從 KnowledgeObject 繼承 16byte，name 4byte，entryno 4Byte，apprCount 1Byte、apprList 4Byte，共 29Byte，但為了對齊記憶體邊界，需 32Byte)，其解法列表需花費 28Byte，因此技術議題節點共需花費 60*100,000=6MB 記憶體空間，而議題名稱需要 52*100,000=5.2MB，因此共需 11.2MB。

解法節點基本資料需花費 32Byte(從 KnowledgeObject 繼承 16byte，keywordEntry 4byte，entryno 4Byte，reportCount 1Byte，reportList 4Byte，共 29Byte，但為了對齊記憶體邊界，需 32Byte)，然而由於其報告列表並不永遠在記憶體中，不納入計算，故可知解法節點佔 32*500,000=16MB 記憶體。

Survey 議題節點基本資料需 32byte(從 KnowledgeObject 繼承 16byte,name 4byte，entryno 4Byte，reportCount 1Byte，reportList 4Byte，共 29Byte，但為了對齊記憶體邊界，需 32Byte)，而其 Survey 報告列表並不永遠在記憶體中，不納入計算，故可知 Survey 議題節點需 32*50,000=1.6MB 記憶體，而 Survey 議題名稱需要 52*50,000=2.6MB，因此 Survey 議題需 4.2MB 記憶體。因此可得知記憶體快取模組靜態儲存在記憶體中的約佔 34MB。

除了靜態記憶體之外，還需要處理使用者查詢的緩衝區(Buffer Area)，假設每個使用者為 8KB(處理簡介摘要等資料應該足夠)，假設系統同時有 1000 位使用者，那麼緩衝區佔 8MB 記憶體。當使用者查詢到解法節點、Survey 議題節點時，會將報告列表載入至記憶體中，技術報告節點需花費 256byte，故可知共需 1000*64*256=16MB 記憶體暫存報告節點。故可知動態記憶體的部分約需 24MB。

假設關鍵字有 50 萬個，而每一個關鍵字需 64Byte，故可知一個關鍵字列表需花費 500,000*64=32MB 儲存，而有解法與技術關鍵字列表，故需花費 64MB。除了列表以外，還需儲存 Radix Tree 的部分，假設 Radix Tree 的層數為 4 層，有 10,000 個 RadixTreeNode，而每一個 RadixTreeNode 需要 112Byte(type 4Byte、childnodes 4Byte，26 個下一層的指標 26*4)，故可得知 RadixTree 需要 10,000*112=1.12MB，而有解法與技術的 RadixTree，故需 2.24MB。由以上可關鍵字列表模組約需 67MB 記憶體。

綜合以上所述,若參照表 4-1 的知識數量,本知識庫系統知識資料需約 125MB 記憶體。

4.3節 技術報告庫系統之設計

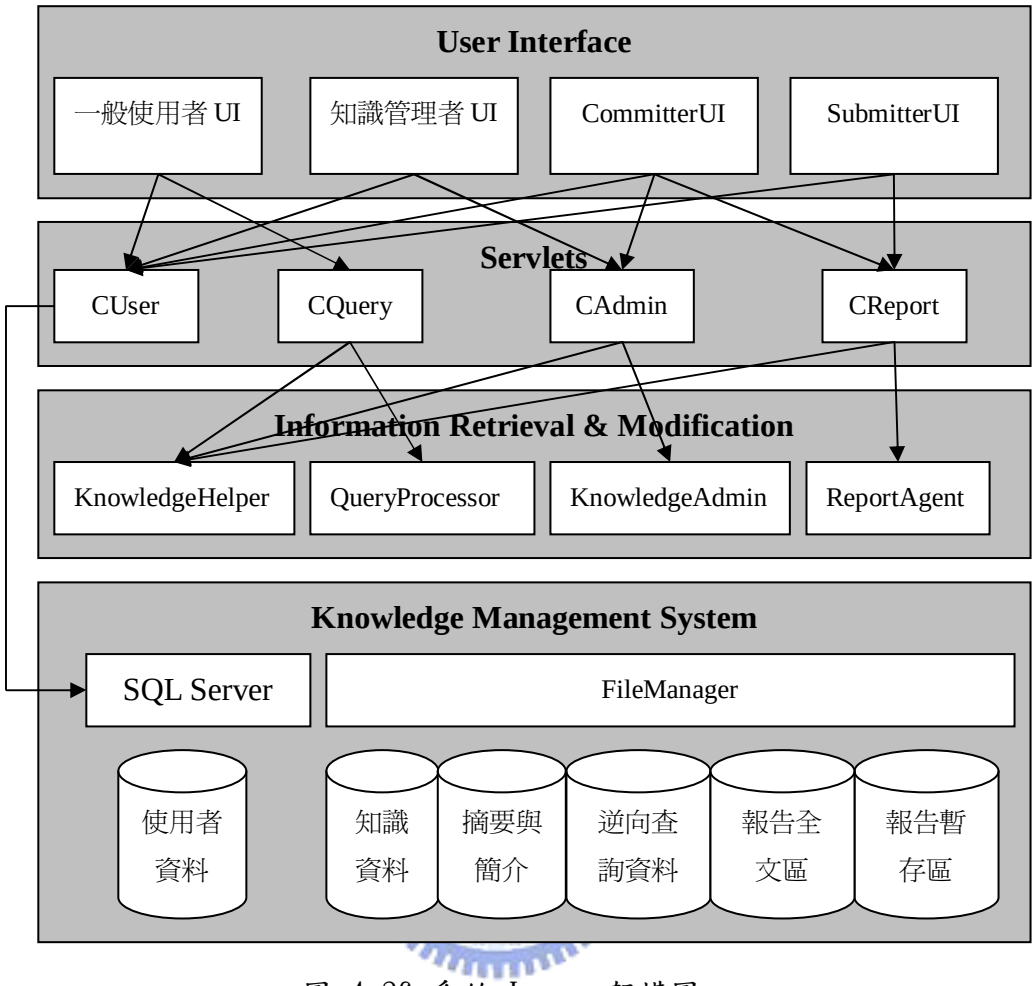


圖 4-26 系統 Layer 架構圖

本知識庫採用 Web-Based 架構，客戶端透過瀏覽器從網頁伺服器瀏覽知識，網頁伺服器執行知識庫系統程式，該程式透過 Java 及 JDBC 等向知識庫取得所需資料。圖 4-26 為系統的 Layer 架構圖，第一層 User Interface 提供使用者的介面，使用者透過該介面與系統互動；第二層 Servlets 則是負責接收使用者傳送的命令，第三層 Information Retrieval & Modification 則是從知識庫中找尋資料以及新增知識，Knowledge Management System 則負責儲存知識資料及其相關的逆向查詢資料。

User Interface Layer

User Interface Layer 主要組成為 JSP 檔案，以及負責讓使用者與瀏覽器互

動的 JavaScript 檔案，共用的 JSP 檔案包括 index.jsp 及 Login.jsp，index.jsp 負責提供樣版，由 Servlet Layer 控制 index.jsp 要 include 哪些 JSP 檔案，Login.jsp 提供使用者登入的畫面。User Interface Layer 依據使用者的類型，包含四個模組：一般使用者 UI、Submitter UI、Committer UI、知識管理者 UI。一般使用者 UI 包括 ViewKnowledge.jsp、ViewReport.jsp、SearchKeyword.jsp、SearchAuthor.jsp、及 SearchYear.jsp。ViewKnowledge.jsp 負責顯示關鍵字知識的資料，如其名稱、路徑、子知識列表等等；ViewReport.jsp 則是負責顯示報告的資料，SearchKeyword.jsp 讓使用者能依關鍵字找尋報告，SearchAuthor.jsp 則是找尋該作者的所有報告，SearchYear.jsp 則是列出特定報告庫中某年代的所有報告。

Submitter UI 包括 SubmitTechniqueReport.jsp 及 SubmitSurveyReport.jsp，提供 Submitter 提交技術報告及 Survey 報告。

Committer UI 包括 SubmitCategory.jsp、ListSubmittedReport.jsp 、ReviewTechniqueReport.jsp、ReviewSurveyReport.jsp 、ListCommittedReport.Jsp、AddTechniqueReport.jsp、及 AddSurveyReport.jsp，SubmitCategory.jsp 提供 Committer 建議新增領域類別，ListSubmittedReport.jsp 負責列出 Submitter 提交過來且尚未審核通過的報告，ReviewTechniqueReport.jsp 與 ReviewSurveyReport.jsp 提供 Committer 審核報告之用，ListCommittedReport.jsp 則列出已通過審核的報告，AddTechniqueReport.jsp 及 AddSurveyReport.Jsp 供 Committer 將報告加入知識庫中。

知識管理者 UI 包括 ListSubmittedKnowledge.jsp 、ViewSubmittedKnowledge.jsp、MoveKnowledge.jsp、及 SplitCategory.jsp。ListSubmittedKnowledge.jsp 提供知識管理者列出待新增的知識(包括領域類

別、技術議題、Survey 議題)，ViewSubmittedKnowledge.jsp 提供知識管理者將建議新增知識加入知識庫中，MoveKnowledge.jsp 提供知識管理者在領域之間搬移知識的介面，SplitCategory.jsp 提供知識管理者分割領域類別的介面。

Servlets Layer

第二層 Servlet Layer 負責接收使用者傳送的參數，並針對該參數執行對應的動作，本系統透過 action 參數來判斷使用者需要做些什麼事情，表 4-2 為各 Servlet 類別所應處理的 action 參數對照表。CUser 負責處理使用者登入/登出；CQuery 則是負責一般使用者的查詢動作，CReport 負責提交報告及將報告新增至知識庫，CAdmin 則是負責建議新增領域、新增知識、搬移知識、分割領域等動作。

本知識庫利用 Java Servlet 所提供的 HttpSession 類別追蹤使用者狀態，透過 user 及 helper 為索引分別儲存使用者物件(User)及 KnowledgeHelper 物件，當需要取得使用者身份或目前知識節點時，只要透過對應索引從 HttpSession 取得即可。

Class	Action 參數	參數意義	Actor	使用案例
CUser	Login	登入	所有 Actor	身分驗證
	Logout	登出		
CQuery	viewKnowledge	檢視知識資料	一般使用者	檢視領域類別、檢視技術議題、檢視 Survey 議題、檢視解法
	getIntro	檢視知識的簡介資料		閱讀知識簡介
	viewReport	檢視報告資料		檢視技術報告、檢視 Survey 報告
	downloadReport	下載報告全文		下載報告全文
	addFavorite	訂閱議題下的新增報告通知		訂閱議題新增報告通知
	searchAuthor	查詢某作者的報告，		利用作者查詢

		參數有 name		報告
	searchTech	查詢運用某技術的報告		利用技術關鍵字查詢報告
	searchAppr	查詢運用某解法的報告		利用解法關鍵字查詢報告
	searchYear	查詢運用某報告庫某年代的報告 r		列出報告庫某年代的報告
CReport	submitReport	提交報告	Submitter	提交新技術報告、提交新 Survey 報告
	listSubmittedReport	列出提交的報告	Committer	審核報告
	reviewReport	閱讀審核報告		
	rejectReport	拒絕報告		
	commitReport	接受報告		
	addReport	將報告加入知識庫		將報告加入知識庫
CAdmin	submitCategory	建議新增領域	Committer	建議新增領域
	listSubmittedKnowledge	列出建議新增的知識	知識庫管理者	新增領域類別、新增技術議題、新增 Survey 議題
	viewSubmittedKnowledge	檢視建議新增的知識		
	commitCategory	將建議新增領域類別加入知識庫		新增領域類別
	commitIssue	將建議新增技術議題加入知識庫		新增技術議題
	commitSurveyIssue	將建議新增 Survey 議題加入知識庫		新增 Survey 議題
	moveKnowledge	搬移知識		搬移領域、搬移技術議題、搬移 Survey 議題
	splitCategory	分割領域類別		分割領域類別

表 4-2 各 Servlet 處理訊息

Information Retrieval & Modification Layer

Information Retrieval & Modification Layer 的類別包括 KnowledgeHelper、QueryProcessor 、ReportAgent、KnowledgeAdmin、XMLParser

以及 4.2 節所提到的 KnowledgeObject、Category、Issue、SurveyIssue、Approach、TechniqueReport、SurveyReport、KeywordTable、KeywordEntry 等等類別。KnowledgeHelper 類別負責存取知識架構樹狀圖、關鍵字列表、以及目前使用者所在節點之資料，並提供以知識編碼找尋知識節點的功能。QueryProcessor 負責處理利用關鍵字查詢報告、利用年代查詢報告、及利用作者查詢報告等功能，ReportAgent 負責處理 Submitter 提交報告以及 Committer 審核報告，而 KnowledgeAdmin 則是負責處理新增知識(如領域類別、技術議題、Survey 議題)、搬移知識、分割領域類別等工作，XMLParser 負責處理報告的關鍵字 XML 檔案。

KnowledgeHelper 有三個靜態變數(static attribute)：root、apprTbl、techTbl，root 指向領域類別樹狀圖的根節點(Root Node)，apprTbl、techTbl 變數分別指向解法關鍵字列表及技術關鍵字列表，此外，KnowledgeHelper 有 current 變數指向使用者目前所在的知識節點，KnowledgeHelper 有下列方法：

1. initCategory()：讀取領域類別檔案資料並建立領域類別架構樹
2. initIssue()：讀取技術議題檔案資料並技術議題節點掛在架構樹
3. initApproach()：讀取解法檔案資料並將解法節點掛在架構樹
4. initSurveyIssue()：讀取 Survey 議題檔案資料並將技術議題節點掛在架構樹
5. initKeywordTable(KeywordTable tbl, String filename)：讀取關鍵字列表檔案資料，並建立關鍵字列表
6. getKnowledge(long id)：取得知識編碼為 id 的知識節點指標
7. getKeywordTable(byte type)：取得對應型態的關鍵字列表
8. setCurrent(KnowledgeObject o)：將知識節點物件設定為目前知識節點。

QueryProcessor 有三個方法：

1. searchKeyword(name, type, category_id):name 為指定的關鍵字,type 為要找的關鍵字型態(解法或技術)，category_id 則為限定領域範圍的編碼，searchKeyword 會開啟該關鍵字對應的逆向查詢檔案，並從中讀取報告編碼，並比對是否在範圍中，若是，則呼叫 KnowledgeHelper 的 getKnowledge 取得報告節點。
2. searchAuthor(author):author 為要查詢的作者名稱,從該作者名稱所對應的逆向查詢資料檔案中讀取報告編碼，並透過 KnowledgeHelper 取得對應的報告節點。
3. searchYear(db, year)：db 為哪一個報告庫，year 為年代，從該報告庫該年代的逆向查詢資料檔案中讀取報告編碼，並透過 KnowledgeHelper 取得對應的報告節點。

ReportAgent 有下列 6 個方法：

1. SubmitTechniqueReport(Hashtable hash, Vector uploadFileItems, Category category, User user)：hash 為 Submitter 所填寫表單(Form)中，每一個輸入所對應的值(如作者名稱、標題等等)，用來取得 Submitter 所填寫的資料，uploadFileItems 為一陣列，陣列中所儲存的物件是上傳的檔案(目前僅有全文檔案)，category 則為 Submitter 所選擇提交的領域類別，user 則為 submitter 所代表的使用者物件。此方法會建立一『尚未通過審核』的報告資料，並依據技術報告的關鍵字架構建立關鍵字 XML 檔案。
2. SubmitSurveyReport(Hashtable hash, Vector uploadFileItems, Category category, User user)：與 SubmitTechniqueReport 類似，只是關鍵字處理採用 SurveyReport 的關鍵字。
3. listSubmittedReport()：列出尚未通過審的報告

4. listCommittedReport()：列出已通過審的報告
5. commitReport(String serial)：serial 為 Submitter 提交報告時，依據當時時間所產生的序號，可由此序號得知 Committer 要 commit 哪一篇報告，此方法會將該報告的資料夾從『尚未通過審核』的資料夾搬移至『通過審核』的資料夾。
6. addReport(String serial)：將報告加入至知識庫。

KnowledgeAdmin 有下列 7 個方法：

1. submitCategory(String name, String intro, Category parent, User user)：name 為建議新增領域的名稱，intro 為其簡介，parent 則為其父領域類別節點，user 代表 Committer 的使用者物件，此方法會在『建議新增領域』資料夾中加入一個檔案，檔案內容包含名稱、簡介、父領域類別編碼、Committer 代號。
2. listSubmittedKnowledge(type)：type 為要列出的知識型態(包括領域類別、技術議題、Survey 議題)，此方法列出要新增至知識庫中的知識。
3. commitCategory(int type, String serial)：type 用來判斷是 Committer 直接建議新增領域或者是由 Submitter 提交報告時建議新增領域，serial 則為當時的時間，此方法會將對應的領域類別加入至知識庫，若是 Submitter 建議新增的領域類別，則會更改對應報告資料夾下的關鍵字 XML 檔案。
4. commitIssue(String serial)：此方法會將對應的技術議題加入至知識庫，並更改對應報告資料夾下的關鍵字 XML 檔案。
5. commitSurveyIssue(String serial)：此方法會將對應的 Survey 議題加入至知識庫，並更改對應報告資料夾下的關鍵字 XML 檔案。
6. moveKnowledge(KnowledgeObject o, Category to)：將知識節點 o 搬移至領域類別 to 之下，並更新其底下的逆向查詢資料。

7. splitCategory(Category parent, LinkedList childs)：parent 為要分割的領域類別，childs 其新的領域類別，childs 中的每一個 node 都紀錄著其名稱與簡介，此方法會將 parent 底下產生數個領域類別，並將原本 parent 底下的技術議題另外暫存起來等待分割領域類別第二階段使用。
8. splitCategory2(Hashtable relation)：relation 紀錄著議題要搬移至哪一個領域類別之下，此方法會將技術議題/Survey 議題節點搬移至對應的領域類別之下，並更新其下的報告之逆向查詢資料。

Knowledge Management System Layer

Knowledge Management System Layer 主要的類別為 FileManager，負責存取知識檔案，此類別有下列 15 個方法：

1. readIntroData(long id)：讀取知識編碼為 id 的簡介檔案
2. writeIntroData(long id,String intro)：id 為知識編碼，intro 為簡介，將 intro 寫入 id 對應的簡介檔案
3. writeCategoryData(Category)：將領域類別的資料寫入領域類別資料檔案
4. writeIssueData(Issue)：將技術議題的資料寫入技術議題資料檔案
5. writeSurveyIssueData(SurveyIssue)：將 Survey 議題的資料寫入 Survey 議題資料檔案
6. writeApproachData(Approach)：將解法的資料寫入解法資料檔案
7. writeReportData(Report)：將報告的基本資料寫入檔案
8. updateReportInvertedFile(Report)：將報告的逆向查詢資料更新，此方法用在新增報告
9. updateReportInvertedFile(Report, long oldid)：將報告的逆向查詢資料更新，此方法用在搬移報告
10. openInvertedFile(String name, byte type, long id)：type 指定關

鍵字型態(解法、技術)，name 為關鍵字名稱，id 為限制的領域範圍，此方法會開啟對應的逆向查詢檔案，並回傳此檔案的 RandomAccessFile 物件。

11. closeInvertedFile(RandomAccessFile f)：將 openInvertedFile 回傳的檔案關閉。
12. openAuthorFile(String name)：name 為作者稱，此方法會開啟該作者對應的逆向查詢檔案，並回傳此檔案的 RandomAccessFile 物件。
13. closeAuthorFile(RandomAccessFile f)：將 openAuthorFile 回傳的檔案關閉。
14. openYearFile(int year,String db)：year 為年代，db 為報告庫，此方法會開啟該報告庫該年代所對應的逆向查詢檔案，並回傳此檔案的 RandomAccessFile 物件。
15. closeYearFile(RandomAccessFile f)：將 openYearFile 回傳的檔案關閉。

以下將針對本知識庫重要的幾項查詢/更新流程做說明，包括：登入流程、瀏覽知識流程、檢視報告流程、以關鍵字查詢報告流程、提交技術報告流程、建議新增領域流程、審核技術報告流程、加入技術報告流程、加入領域類別流程、搬移領域類別流程、分割領域類別流程，由於 Survey 報告的提交/審核/加入流程與技術報告流程類似，加入技術議題/Survey 議題與加入領域類別流程類似，搬移技術議題/Survey 議題與搬移領域類別流程類似，故不再贅述。

登入流程

如圖 4-27 所示，使用者透過 Login.jsp 輸入帳號與密碼，CUser 透過 JDBC 連線到使用者資料庫取得使用者的資訊(如編號、帳號、密碼、Email 等)，並檢查密碼是否正確，若是則產生使用者物件(User)，該物件持有使用者的帳號、編號、權限等資料，並將該物件以 user 為索引指派給目前的 HttpSession 物件；

除此之外，當使用者登入之後，會將根節點(Root Node)設定為目前的知識節點，並產生該使用者專屬的 KnowledgeHelper 物件，以供以後查詢使用。

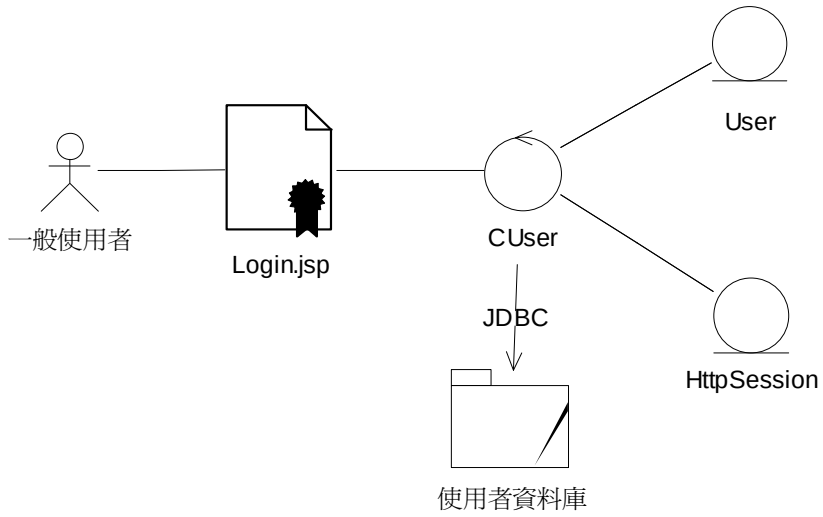


圖 4-27 使用者登入流程

瀏覽知識流程

如圖 4-28 所示，使用者透過 index.jsp 點選知識，index.jsp 傳送知識編碼給 CQuery，CQuery 先從 HttpSession 取得該使用者對應的 KnowledgeHelper 物件，然後呼叫 KnowledgeHelper 的 getKnowledge 取得該知識節點指標，並呼叫 setCurrent 設定目前知識節點。最後 index.jsp 將知識節點轉交給 ViewKnowledge.jsp 顯示其結果。

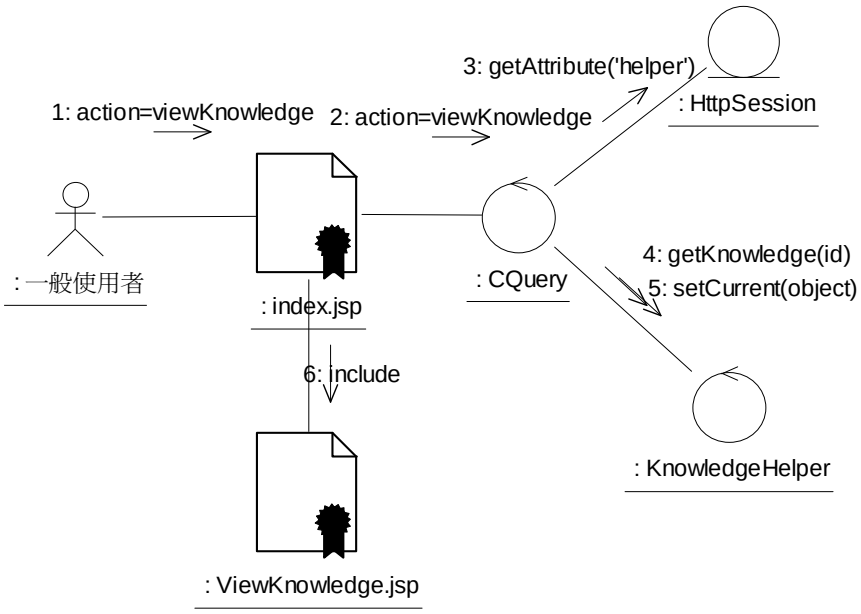


圖 4-28 瀏覽知識流程

檢視報告流程

如圖 4-29 所示，使用者透過 index.jsp 點選知識，index.jsp 傳送報告編碼給 CQuery，CQuery 先從 HttpSession 取得該使用者對應的 KnowledgeHelper 物件，然後呼叫 KnowledgeHelper 的 getKnowledge 取得該報告節點指標，並呼叫 setCurrent 設定目前知識節點。最後 index.jsp 將知識節點轉交給 ViewReport.jsp 顯示其結果。

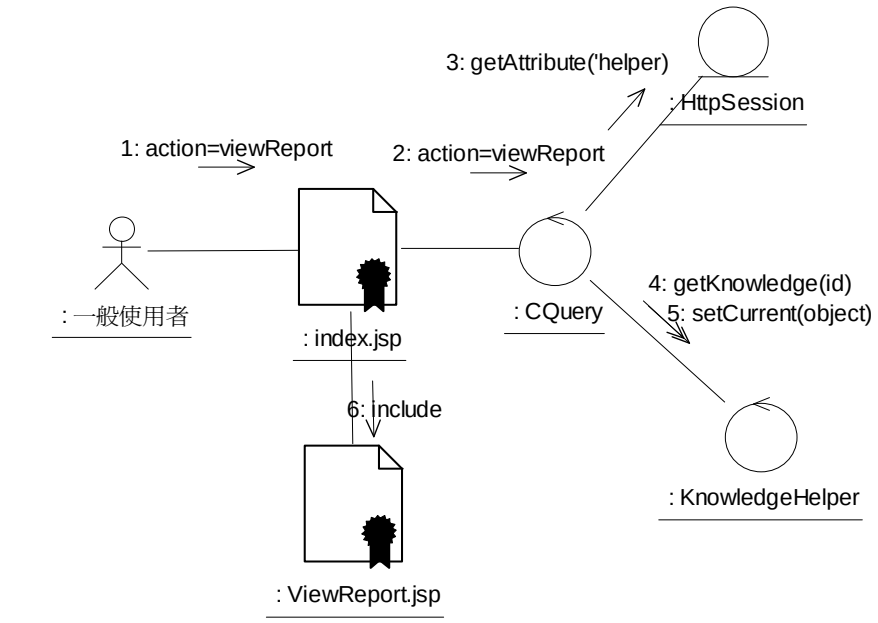


圖 4-29 檢視報告流程

以關鍵字查詢報告流程

如圖 4-30 所示，使用者透過 searchKeyword.jsp 輸入關鍵字及查詢範圍，searchKeyword.jsp 傳送關鍵字與查詢範圍編碼給 CQuery，CQuery 先產生 QueryProcessor 物件，然後呼叫 QueryProcessor 的 searchKeyword 方法，該方法產生 FileManager 後呼叫 FileManager 的 openInvertedFile 以開啟該關鍵字名稱對應的逆向查詢檔案，然後從中讀取資料，檢查是否落於查詢範圍中，若是則呼叫 KnowledgeHelper 的 getKnowledge 取得報告節點指標，最後 CQuery 將查詢結果交由 searchKeyword.jsp 顯示。

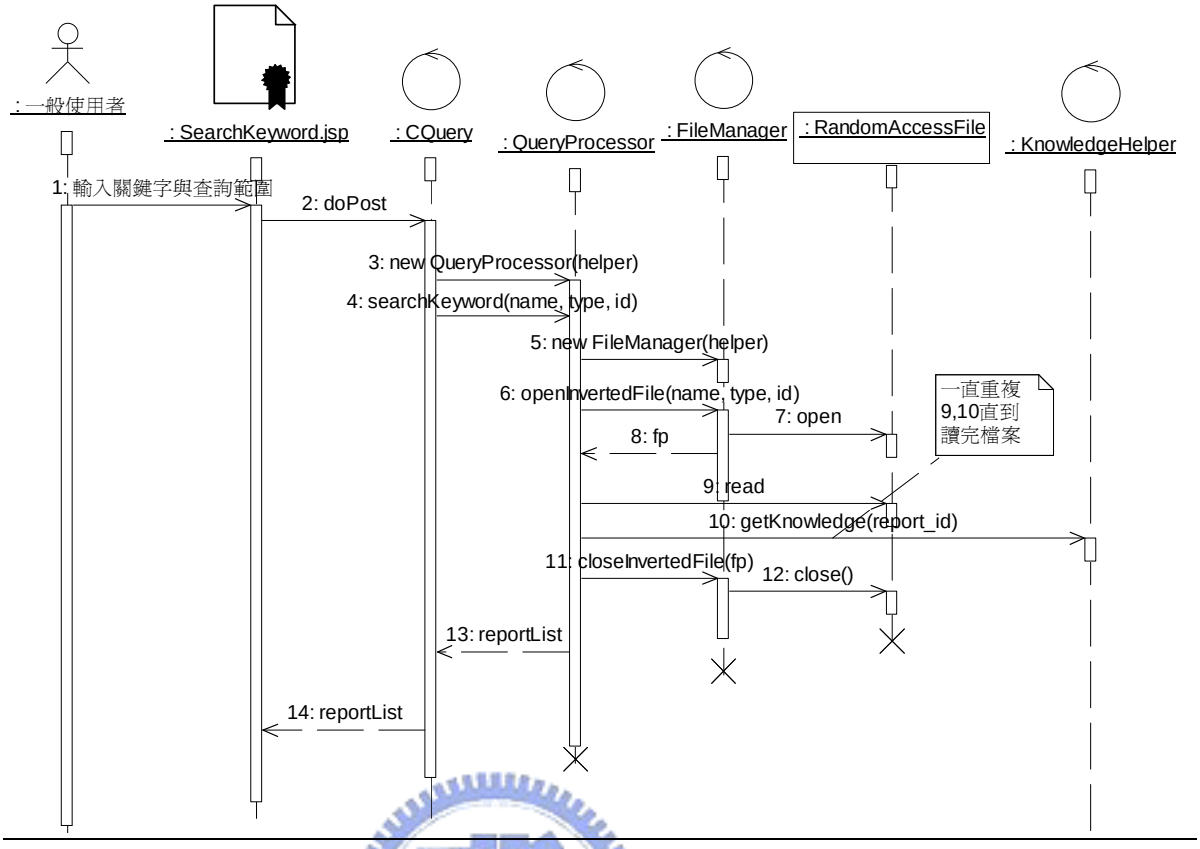


圖 4-30 以關鍵字查詢報告流程

提交技術報告流程

如圖 4-31 所示，Submitter 透過 submitTechniqueReport.jsp 輸入該報告的關鍵字、標題、作者、摘要，並選擇全文檔案上傳，submitTechniqueReport.jsp 將資料轉交給 CReport，CReport 先產生 ReportAgent 物件，呼叫其 submitTechniqueReport 方法，該方法先建立 XMLParser，設定 XML 關鍵字檔案所需的關鍵字路徑、建議新增知識名稱與簡介，而後呼叫 XMLParser 的 write 方法將關鍵字資料寫入 XML 檔案，接著 submitTechniqueReport 依序寫入標題作者檔案、摘要檔案，並將上傳的全文檔案搬移到暫存目錄中。

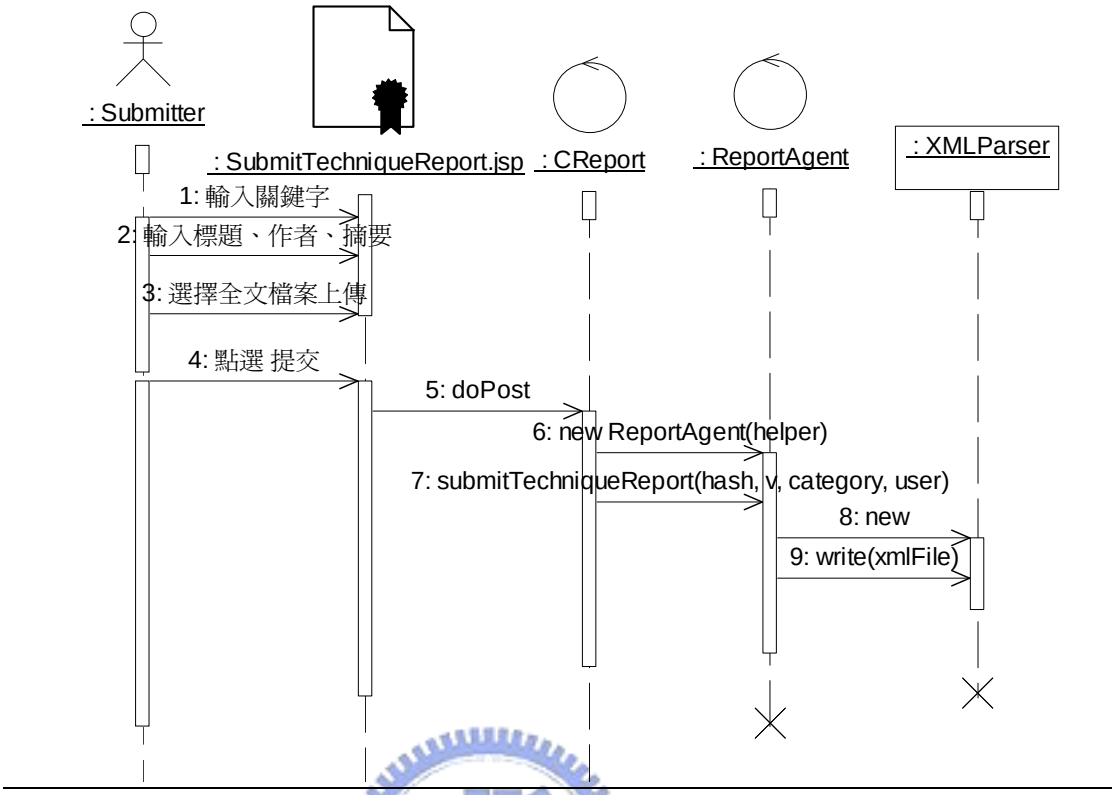


圖 4-31 提交技術報告流程

建議新增領域流程

如圖 4-32 所示，Committer 透過 submitCategory.jsp 輸入領域類別的名稱、簡介，並選擇其父領域類別，submitCategory.jsp 將資料轉交給 CAdmin，CAdmin 先產生 KnowledgeAdmin 物件，然後呼叫其 submitCategory 方法。最後，透過 JDBC 與使用者資料庫連線，取得知識管理者的 Email，並寄發『建議新增領域通知信』給知識管理者。

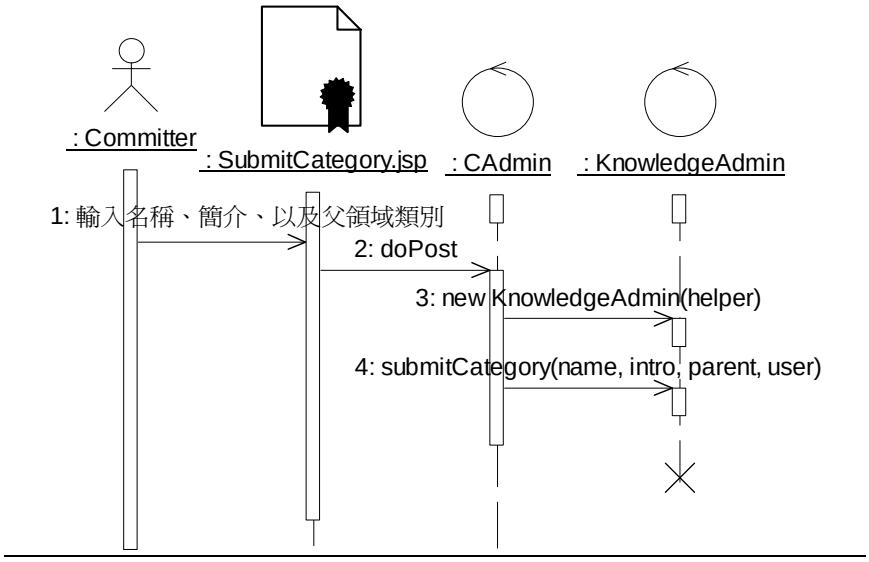


圖 4-32 建議新增領域流程

審核技術報告流程

如圖 4-33 所示，當 Committer 點選『接受』時，reviewTechniqueReport.jsp 向 CReport 下 commitReport 命令，CReport 先產生 ReportAgent 物件，然後呼叫其 commitReport 方法，該方法將報告資料夾從『尚未通過審核』資料夾搬移至『通過審核』資料夾，依據是否有建議新增知識產生 add_category、add_issue、add_surveyissue、add_done 等檔案。若有建議新增領域、議題，透過 JDBC 與使用者資料庫連線，取得知識管理者的 Email，並寄發『建議新增領域/議題通知信』給知識管理者。

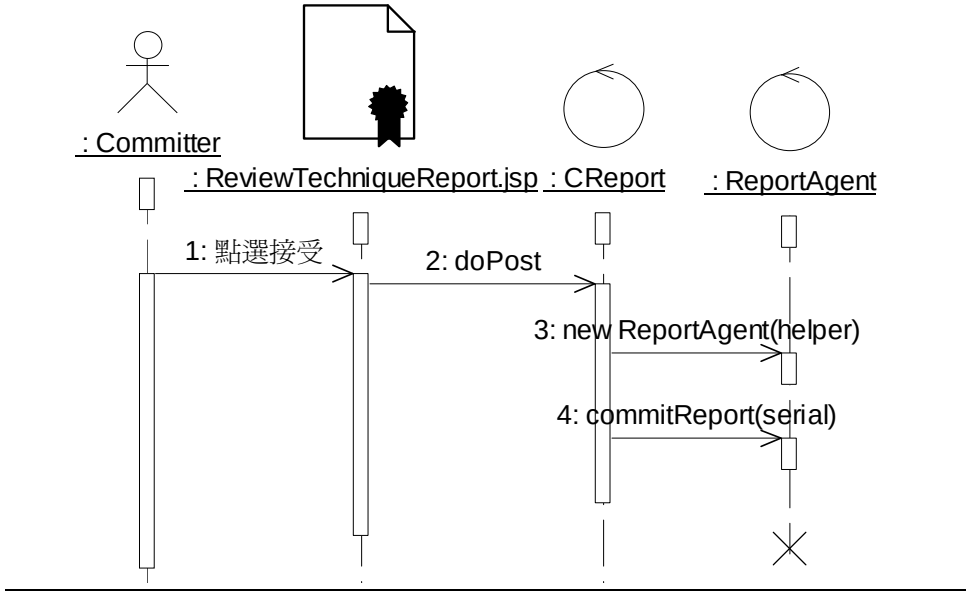


圖 4-33 審核技術報告流程

加入技術報告流程

如圖 4-34 所示，Committer 透過 AddTechniqueReport.jsp 輸入報告庫名稱，AddTechniqueReport.jsp 將資料轉交給 CReport，CReport 先產生 ReportAgent 物件，然後呼叫其 addReport 方法。addReport 依據傳過來的序號開啟關鍵字 XML 檔案，並檢查是否有建議新增解法(圖 4-34 的狀況為有建議新增解法)，若有則產生 Approach 物件，並將名稱、技術議題等資料填入及建立連結後，產生 FileManager 並呼叫 FileManager 的 writeApproachData 及 writeIntro 以寫入解法的資料及簡介。addReport 接著產生 TechniqueReport 節點，並將報告的資料填入後呼叫 FileManager 的 writeReportData 及 updateReportInvertedFile 以寫入報告的資料及逆向查詢資料。addReport 接著將摘要及全文檔案從暫存區搬移至報告區，最後將暫存區的資料夾刪除。最後，透過 JDBC 與使用者資料庫連線，取得 Submitter 的 Email，並寄發『其報告已經加入知識庫通知信』給 Submitter。

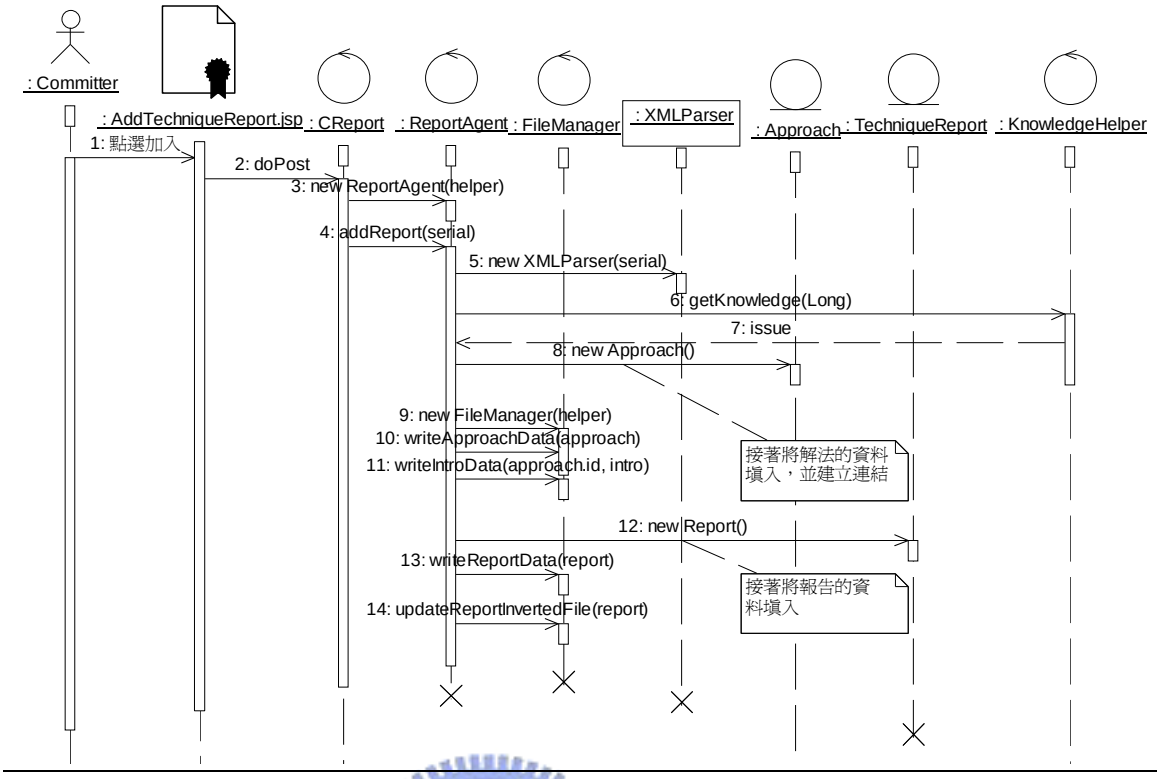


圖 4-34 加入技術報告流程

加入領域類別流程

如圖 4-35 所示，知識管理者透過 CommitCategory.jsp 將領域類別加入知識庫中，CommitCategory.jsp 將資料轉交給 CAdmin，CAdmin 產生 KnowledgeAdmin 物件，並呼叫其 commitCategory 方法。commitCategory 方法先產生 FileManager 物件，並呼叫 KnowledgeHelper 的 getKnowledge 方法以取得該節點的父領域類別，然後產生 Category 物件，將名稱及父領域類別節點指標填入後，呼叫 FileManager 的 writeCategoryData 及 writeIntroData 將領域類別的資料及簡介寫入檔案。如果該領域類別是從 Committer 建議新增過來的，則移除該建議新增的領域類別檔案，並透過 JDBC 與使用者資料庫連線，取得該 Committer 的 Email，通知該 Committer 已經加入將領域類別加入知識庫中。如果該領域類別是透過 Submitter 新增報告時建議新增的，則修改該關鍵字 XML 檔案，並移除 add_category 檔案及加入 add_issue/add_surveyissue 檔案。加入技術議題及加入 Survey 議題流程與加入領域類別類似，只是最後要透過 JDBC 與使用者資料

庫連線以取得 Committer 的 email，寄發『已將報告建議新增知識加入知識庫通知信』給該 Committer。

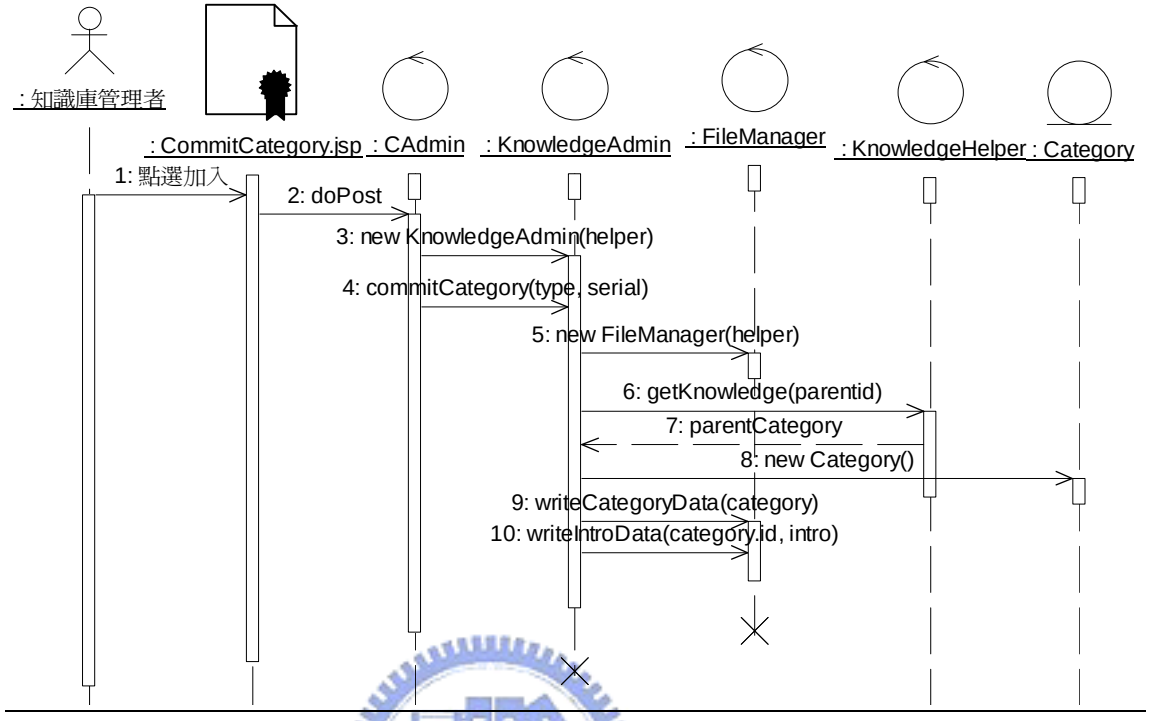


圖 4-35 加入領域類別流程

搬移領域類別流程

如圖 4-36 所示，知識管理者透過 MoveKnowledge.jsp 選擇搬移的來源及目的，MoveKnowledge.jsp 將資料轉交給 CAdmin，CAdmin 先呼叫 KnowledgeHelper 的 getKnowledge 方法以取得要搬移的領域類別及目的領域類別，然後產生 KnowledgeAdmin 物件並呼叫其 moveKnowledge 方法。moveKnowledge 產生 FileManager 物件，計算來源領域類別新的知識編碼，並移除來源領域類別與其父領域類別的連結，然後建立來源領域類別與目的領域類別的連結，接著呼叫 FileManager 的 writeCategoryData 將來源領域類別、目的領域類別、原來父領域類別的資料寫入檔案。寫入檔案後，moveKnowledge 針對來源領域類別下的每一個子領域/技術議題及 Survey 議題，呼叫 cascadeMoveKnowledge(KnowledgeObject o, FileManager fm)，o 為子知識物件，由於子知識物件與其父親的關係並沒有打斷，因此 cascadeMoveKnowledge 只要更新子知識的編碼，並將子知識節點資料寫入檔案即可，如果子知識為報

告，則呼叫 FileManager 的 updateReportInvertedFile 方法更新報告的逆向查詢資料，否則繼續針對子知識下的每一個知識物件呼叫 cascadeMoveKnowledge。

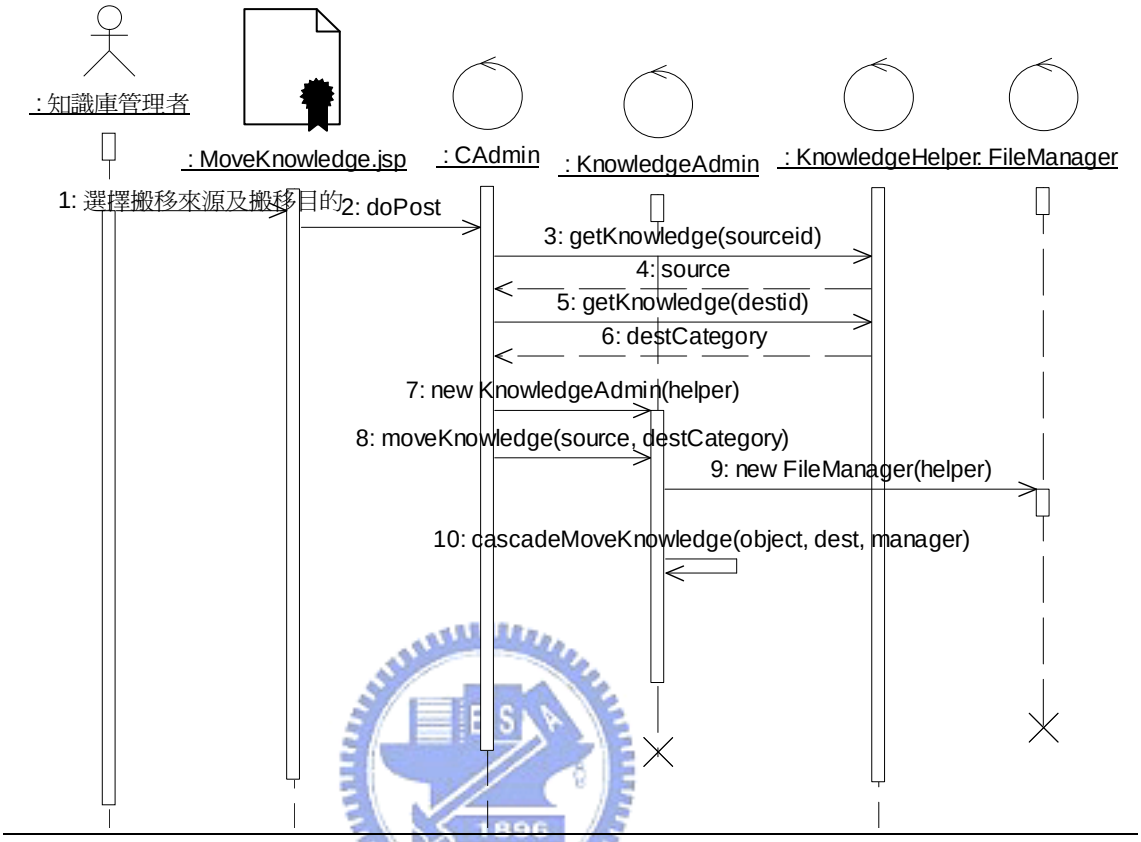


圖 4-36 搬移領域類別流程

分割領域類別流程

如圖 4-37 所示，知識管理者透過 SplitCategory.jsp 選擇要分割的領域類別，並輸入領域類別的名稱、簡介，SplitCategory.jsp 將資料轉交給 CAdmin，CAdmin 先呼叫 KnowledgeHelper 的 getKnowledge 取得要分割的領域類別物件，並將該物件的技術議題列表暫存在 Session 物件中，然後產生 KnowledgeAdmin 物件並呼叫其 splitCategory 方法，splitCategory 方法會產生 FileManager，splitCategory 接著依據知識管理者的設定，產生一至多個 Category 物件，並建立與要分割的領域類別節點的連結，然後呼叫 FileManager 的 writeCategoryData、writeIntroData 將這些子領域類別資料寫入。接著知識管理者再選擇原領域類別下的技術議題、Survey 議題要搬移到哪一個子領域，CAdmin 會透過 KnowledgeHelper 的 getKnowledge 方法取得議題節點與子領域類

別節點，並將其要搬移的關係建立起來放入串列(LinkedList)之中，然後呼叫 KnowledgeAdmin 物件的 splitCategory2 方法，該方法會將議題節點的知識編碼更新，並呼叫 FileManager 的 writeIssueData/writeSurveyIssueData 將資料寫入檔案，然後針對議題下的知識呼叫 cascadeMoveKnowledge 方法將議題下的知識一起搬移至新的領域類別之下。

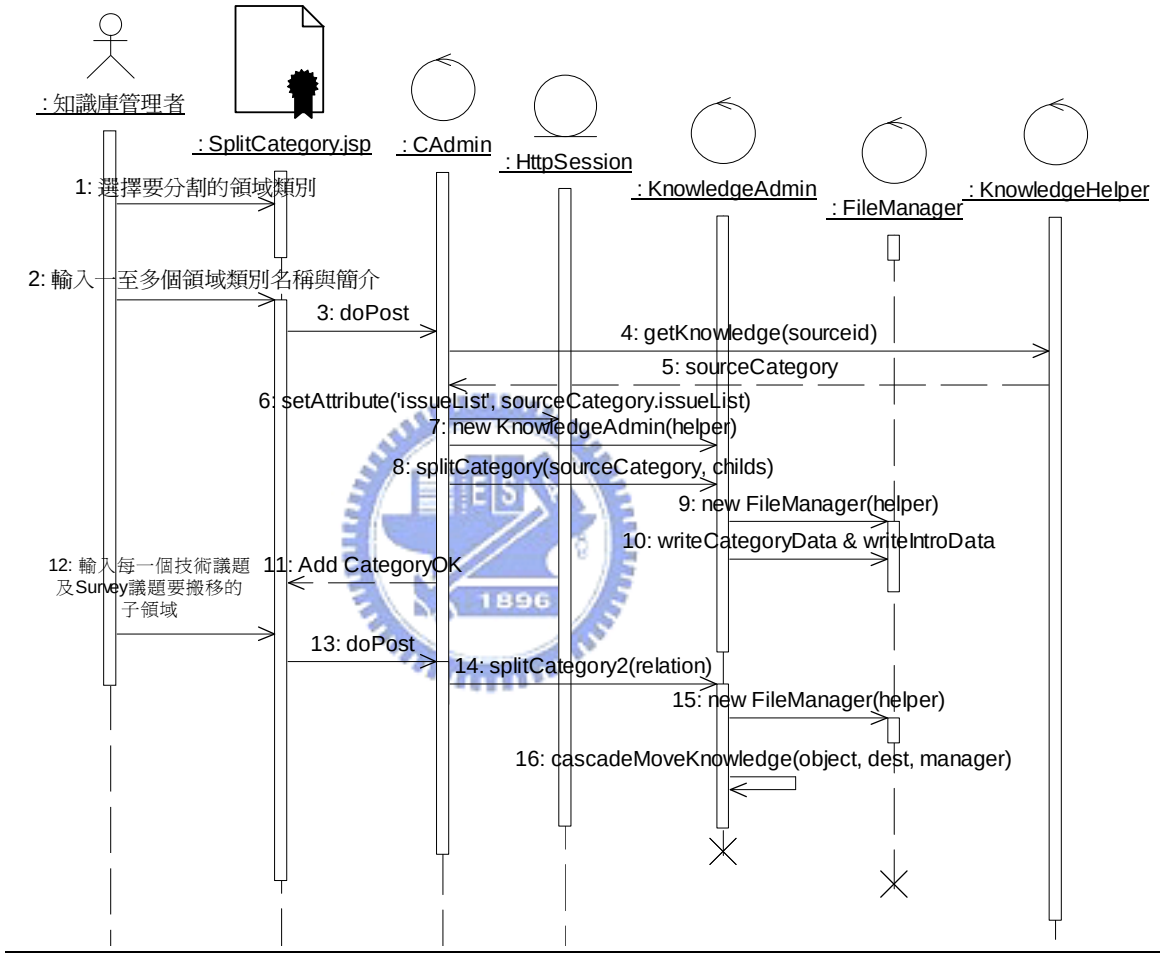


圖 4-37 分割領域類別流程

第5章 其他類型知識庫系統之設計

第四章說明技術報告庫的知識庫系統之設計細節，本章將介紹私人知識庫、期刊資料庫、全文資料庫、以及大型知識庫系統之設計。

5.1節 私人知識庫之設計

如第三章所述，私人知識庫與技術報告庫的差異在於：

1. 私人知識庫的知識種類多了論文知識與書本知識
2. 私人知識庫會訂閱期刊，因此需要依據期刊查詢的功能

私人知識庫的技術論文及 Survey 論文的來源有二：訂閱的期刊及從網路上下載的論文；而其技術報告/Survey 報告的來源有二：從網路上下載的報告及公司內部所產生的技術報告/Survey 報告。公司訂閱的期刊、從網路上下載的論文及報告都可開放給公司員工閱讀;但是公司內部所產生的報告並非所有員工均可閱讀，因此需要透過報告的存取控制列表(Access Control List，簡稱 ACL)來作管制，即採用層級及部門來做控管，層級可分為：公司核心主管、部門主管、高階主管、主管、一般員工，而 ACL 可針對報告設定該報告可給予哪些部門的哪些層級的員工存取，此 ACL 檔案根據報告編碼存放，假設一篇報告的第一層領域類別編碼為 10，第二層為 5，第三層為 20，技術議題編碼為 8，解法編碼為 4，報告編碼為 3，則其 ACL 檔案為『知識資料/10/5/20/Issue8/4/3.acl』，若此檔案不存在，則為所有員工皆可存取。

技術論文/Survey 論文的儲存方式與技術報告/Survey 報告一樣，故不再贅述。書本知識的儲存方式與解法的論文/報告列表類似，依據領域類別的知識編碼儲存，假設該書本知識的第一層領域類別編碼為 10，第二層為 5，第三層為

20，書本編碼為 4，則其知識資料存放在『知識資料/10/5/20/BookList』檔案中的第四個區塊，書本簡介存放在『知識資料/10/5/20/Book4』資料夾下的 intro 檔案，書本議題資料存放在『知識資料/10/5/20/Book4』資料夾下的 issue 檔案，而書本的電子檔，則與報告一樣儲存在『全文/10/50/20/』資料夾下的 Book4。

使用案例

Actor	使用案例
一般使用者	檢視領域類別(修改流程) 檢視技術論文 檢視 Survey 論文 下載論文/報告全文(更改名稱) 檢視書本知識 下載書本電子檔
	利用解法關鍵字查詢報告/論文(更改名稱) 利用技術關鍵字查詢報告/論文(更改名稱) 利用書本議題查詢書本知識 利用作者查詢報告/論文(更改名稱) 利用作者查詢書本 依據期刊、Volume、Issue 查詢論文
Submitter	提交新技術報告(更改流程) 提交新 Survey 報告(更改流程) 提交新技術論文 提交新 Survey 論文 提交新書本知識
Committer	審核論文 將論文加入知識庫 審核書本知識 將書本知識加入知識庫

表 5-1 新增或修改的使用案例

表 5-1 為私人知識庫所需新增或修改的使用案例，一般使用者所需修改的使用案例僅有『檢視領域類別』，而『下載論文/報告全文』、『利用解法關鍵字找尋論文/報告』、『利用技術關鍵字找尋論文/報告』、『利用作者找尋論文/報告』只是修改使用案例名稱，其流程不變，而新增的使用案例有『檢視技術論文』、『檢視 Survey 論文』、『檢視書本知識』、『下載書本電子檔』、『利用書本議題查詢書

本知識』、『利用作者查詢書本』、『依據期刊、Volume、Issue 查詢論文』，以下將一一介紹這些新增或修改的使用案例。

- 檢視領域類別

Precondition：畫面上列出某一領域類別之所有子領域名稱後，使用者點選其中一子領域

一般使用者	系統
8. 點選領域類別的子領域類別其中一個	
	9. 讀取子領域類別編號 10. 利用子領域類別編號從目前知識節點指標中找出對應的子節點指標 11. 更新目前知識節點指標 12. 讀取領域類別名稱，並顯示出來 13. 讀取子領域類別/技術議題列表並顯示出來 14. 檢查是否有 Survey 議題，若是，則讀取 Survey 議題列表並顯示 15. 檢查是否有書本知識，若是，則讀取書本知識列表並顯示

- 檢視技術論文

Precondition：使用者已經瀏覽到解法節點，畫面上顯示其論文/報告列表。

一般使用者	系統
10. 點選列表其中一篇論文	
	11. 讀取論文編號 12. 利用論文編號從目前解法節點指標中找出該論文節點 13. 更新目前知識節點指標 14. 讀取節點中的技術編碼列表 15. 從技術關鍵字列表將該論文所運用的技術列舉出來 16. 利用論文編碼計算出摘要檔名，從摘要檔案讀取摘要 17. 顯示摘要、技術列表等資料，並顯示[下載全文]連結

- 檢視 Survey 論文

Precondition：使用者瀏覽到 Survey 議題節點，畫面上列出所有 Survey 論文/報告。

一般使用者	系統
9. 點選 Survey 論文/報告列表其中一篇論文	
	10. 讀取論文編號 11. 利用論文編號從目前知識節點指標中找出該論文節點 12. 更新目前知識節點指標 13. 利用論文編碼計算出子議題列表檔名,並從該檔案讀取子議題列表(包含名稱、其下的解法列表) 14. 利用論文編碼計算出摘要檔名，從摘要檔案讀取摘要 15. 顯示摘要、子議題列表等資料，並顯示[下載全文]連結

- 檢視書本知識

Precondition：使用者瀏覽到有書本知識的領域類別，畫面上列出該領域類別下的所有書本知識。

一般使用者	系統
1. 點選書本知識列表其中一本書	
	2. 讀取書本知識編號 3. 利用書本知識編號從目前知識節點指標中找出該書本知識節點 4. 更新目前知識節點指標 5. 利用編號計算出書本議題檔名，並從該檔案讀取書本議題 6. 利用編號計算出摘要檔名，從摘要檔案讀取摘要 7. 顯示摘要、議題列表、目錄等資料 8. 若有書本電子檔，則顯示[下載電子檔]連結

- 下載書本電子檔

Precondition：使用者已經瀏覽到書本知識節點，該書本有電子檔。

一般使用者	系統
5. 點選[下載電子檔]	
	6. 從目前知識節點取得書本編碼 7. 利用編碼計算電子檔檔名，讀取該檔案的資料 8. 顯示該電子檔

- 利用書本議題查詢書本知識

Precondition：使用者瀏覽到『領域類別』節點，畫面上出現『利用書

本議題查詢』連結

一般使用者	系統
16. 點選『利用書本議題查詢』	
	17. 依據目前知識節點指標，列出所有解法(議題節點取出該議題下的解法，解法、技術報告節點則直接取出其解法，Survey 報告則取出該報告所列舉的解法)
18. 選擇一解法	
	19. 讀取解法名稱 20. 列出目前知識節點的第一層領域類別、第二層領域類別…一直到最該知識節點的父領域類別
21. 選擇其中一個領域	
	22. 從該領域類別編碼，找出該領域類別下的子領域類別列表 23. 列出子領域類別列表
24. 重複步驟 6~8，直到使用者點選『設定領域範圍』按鈕	
	25. 讀取領域範圍編碼 26. 依據解法名稱，至解法關鍵字列表中找出該解法對應的逆向查詢索引 27. 根據逆向查詢索引找出有用到該解法的報告編碼 28. 比較報告編碼是否落於領域範圍之中，若是則依據該編碼取得報告節點，並透過報告節點取得該報告的名稱、作者 29. 將所有報告依據領域類別、議題排序 30. 依領域及議題順序，顯示其領域類別名稱、議題名稱及該議題下有用到該解法的報告之標題及作者

- 利用作者查詢書本

一般使用者	系統
7. 點選[利用作者查詢書本]	
	8. 顯示作者欄位
9. 輸入『作者』	
	10. 讀取作者名稱 11. 依據作者名稱，從作者列表中找出該作者所有的書本編碼 12. 利用編碼找出該書本節點並讀取標題、作者、年代、出版社，並顯示之

- 依據期刊、Volume、Issue 查詢論文

Precondition：畫面上有一個連結『Journals』

一般使用者	系統
11. 點選[Journals]	
	12. 依據期刊列表顯示所有的期刊
13. 點選其中一個期刊	
	14. 讀取期刊 15. 列出該期刊所有的 Volume
16. 點選其中一個 Volume	
	17. 讀取 Volume 18. 列出該 Volume 下所有的 Issue
19. 點選其中一個 Issue	
	20. 讀取 Issue 21. 依據期刊、Volume、Issue，取得其論文列表檔案 22. 從論文列表檔案讀取編碼，並利用編碼找出論文節點並取出標題、作者 23. 顯示標題及作者

Submitter 的使用案例有五個：提交新技術報告、提交新 Survey 報告、提交新技術論文、提交新 Survey 論文、提交新書本知識，由於建議新增領域、議題等流程與技術報告相同，故在此假設都不需要建議新增知識，分別說明如下：

- 提交新技術報告

Precondition：當 Submitter 瀏覽到最末端領域類別，畫面上出現『提交技術報告』連結

Submitter	系統
45. 點選[提交技術報告]	
	46. 詢問是否要建議新增領域類別
47. 選擇否	
	48. 列出該領域下的議題，並詢問是否要建議新增技術議題
49. 選擇其中一個議題	
	50. 讀取技術議題編碼 51. 列出該議題下的解法，詢問是否要新增解法
52. 選擇其中一個解法	
	53. 讀取解法編碼 54. 顯示八個『技術』欄位
55. 輸入技術關鍵字	

	56. 讀取技術關鍵字 57. 顯示標題、作者、摘要、全文等欄位
58. 輸入標題、作者、摘要，並選擇全文檔案上傳	
	59. 讀取報告的標題、作者、摘要、全文檔案 60. 依據目前時間取得報告序號 61. 在『未通過審核』資料夾下建立名稱爲報告序號的資料夾 62. 將報告全文資料寫入報告資料夾的全文檔案 63. 將報告摘要資料寫入報告資料夾的摘要檔案 64. 將報告的標題、作者寫入報告資料夾的標題作者檔案 65. 將領域類別路徑寫入報告資料夾的關鍵字檔案 66. 將技術議題名稱寫入關鍵字檔案 67. 將解法名稱及其繼承解法名稱寫入關鍵字檔案 68. 將技術名稱列表寫入關鍵字檔案 69. 詢問此報告是否爲公司內部報告
70. 若不是公司內部報告，則跳至步驟 30	
	71. 詢問可存取報告的部門，及該部門可讀取報告的層級
72. 輸入部門及該部門可讀取報告的層級	
	73. 將部門及層級資訊寫入該報告的 acl 檔案 74. Email 通知 Committer 有報告提交 75. 顯示完成提交報告

- **提交新 Survey 報告**

Precondition：當 Submitter 瀏覽到領域類別，畫面會出現『提交 Survey 報告』連結

Submitter	系統
35. 點選[提交 Survey 報告]	
	36. 詢問是否要建議新增領域類別
37. 點選否	
	38. 列出該領域類別下的 Survey 議題，並詢問是否要新增 Survey 議題
39. 選擇一個 Survey 議題	

	40. 紀錄議題編碼 41. 顯示子議題欄位
42. 輸入子議題	
	43. 讀取子議題資料 44. 顯示解法欄位
45. 輸入解法資料	
	46. 讀取解法資料 47. 顯示標題、作者、摘要、全文等欄位
48. 輸入標題、作者、摘要，並選擇全文檔案上傳	
	49. 讀取報告的標題、作者、摘要、全文檔案等資料 50. 依據目前時間取得報告序號 51. 在『未通過審核』資料夾下建立名稱爲報告序號的資料夾 52. 將報告全文資料寫入報告資料夾的全文檔案 53. 將報告摘要資料寫入報告資料夾的摘要檔案 54. 將報告的標題、作者寫入報告資料夾的標題作者檔案 55. 將領域類別路徑寫入報告資料夾的關鍵字檔案 56. 將 Survey 議題名稱寫入關鍵字檔案 57. 將子議題列表及其解法寫入關鍵字檔案 58. 詢問此報告是否爲公司內部報告
59. 若不是公司內部報告，則跳至步驟 29	
	60. 詢問可存取報告的部門，及該部門可讀取報告的層級
61. 輸入部門及該部門可讀取報告的層級	
	62. 將部門及層級資訊寫入該報告的 acl 檔案 63. Email 通知 Committer 有報告提交 64. 顯示完成提交報告

- **提交新技術論文**

Precondition：當 Submitter 瀏覽到最末端領域類別，畫面上出現『提交技術論文』連結

Submitter	系統
1. 點選[提交技術論文]	

	2. 詢問是否要建議新增領域類別
3. 選擇否	
	4. 列出該領域下的議題,並詢問是否要建議新增技術議題
5. 選擇其中一個議題	
	6. 讀取技術議題編碼 7. 列出該議題下的解法，詢問是否要新增解法
8. 選擇其中一個解法	
	9. 讀取解法編碼 10. 顯示八個『技術』欄位
11. 輸入技術關鍵字	
	12. 讀取技術關鍵字 13. 顯示標題、作者、期刊、Volume、Issue、年代、摘要、全文等欄位
14. 輸入標題、作者、期刊、Volume、Issue、年代、摘要，並選擇全文檔案上傳	
	15. 讀取論文的標題、作者、期刊、Volume、Issue、年代、摘要、全文檔案 16. 依據目前時間取得論文序號 17. 在『未通過審核』資料夾下建立名稱爲論文序號的資料夾 18. 將全文資料寫入論文資料夾的全文檔案 19. 將摘要資料寫入論文資料夾的摘要檔案 20. 將標題、作者、期刊、Volume、Issue、年代寫入論文資料夾的標題作者檔案 21. 將領域類別路徑寫入論文資料夾的關鍵字檔案 22. 將技術議題名稱寫入關鍵字檔案 23. 將解法名稱及其繼承解法名稱寫入關鍵字檔案 24. 將技術名稱列表寫入關鍵字檔案 25. Email 通知 Committer 有論文提交 26. 顯示完成提交論文

- 提交新 Survey 論文

Precondition：當 Submitter 瀏覽到領域類別，畫面會出現『提交 Survey 論文』連結

Submitter	系統

1. 點選[提交 Survey 論文]	
	2. 詢問是否要建議新增領域類別
3. 點選否	
	4. 列出該領域類別下的 Survey 議題，並詢問是否要新增 Survey 議題
5. 選擇一個 Survey 議題	
	6. 紀錄議題編碼 7. 顯示子議題欄位
8. 輸入子議題	
	9. 讀取子議題資料 10. 顯示解法欄位
11. 輸入解法資料	
	12. 讀取解法資料 13. 顯示標題、作者、期刊、Volume、Issue、年代、摘要、全文等欄位
14. 輸入標題、作者、期刊、Volume、Issue、年代、摘要，並選擇全文檔案上傳	
	15. 讀取標題、作者、期刊、Volume、Issue、年代、摘要、全文檔案等資料 16. 依據目前時間取得論文序號 17. 在『未通過審核』資料夾下建立名稱爲論文序號的資料夾 18. 將全文資料寫入論文資料夾的全文檔案 19. 將摘要資料寫入論文資料夾的摘要檔案 20. 將標題、作者期刊、Volume、Issue、年代寫入論文資料夾的標題作者檔案 21. 將領域類別路徑寫入論文資料夾的關鍵字檔案 22. 將 Survey 議題名稱寫入關鍵字檔案 23. 將子議題列表及其解法寫入關鍵字檔案 24. Email 通知 Committer 有論文提交 25. 顯示完成提交論文

- 提交書本知識

Precondition：當 Submitter 瀏覽到領域類別，畫面會出現『提交書本知識』連結

Submitter	系統

1. 點選[提交書本知識]	
	2. 詢問是否要建議新增領域類別
3. 點選否	
	4. 顯示書本議題欄位
5. 輸入書本議題	
	6. 讀取書本議題資料 7. 顯示標題、作者、出版社、年代、摘要、全文等欄位
8. 輸入標題、作者、出版社、年代、摘要，並選擇全文檔案上傳	
	9. 讀取標題、作者、出版社、年代、摘要、全文檔案等資料 10. 依據目前時間取得書本序號 11. 在『未通過審核』資料夾下建立名稱爲書本序號的資料夾 12. 將全文資料寫入書本資料夾的全文檔案 13. 將摘要資料寫入書本資料夾的摘要檔案 14. 將標題、作者、出版社、年代寫入書本資料夾的標題作者檔案 15. 將領域類別路徑寫入書本資料夾的關鍵字檔案 16. 將書本議題列表及其子議題寫入關鍵字檔案 17. Email 通知 Committer 有書本提交 18. 顯示完成提交書本

Committer 需要新增或修改的使用案例包括：『審核論文』、『將論文加入知識庫』、『審核書本知識』、『將書本知識加入知識庫』，說明如下(假設沒有建議新增知識、不需要修改路徑、接受 Submitter 提交的知識)：

- 審核論文

Precondition：如果系統有尚待審核的論文，則在 Committer 使用系統時，會出現『審核論文』

Committer	系統
43. 點選[審核論文]	
	44. 從『待審核論文』目錄中，讀取每一個論文資料夾的標題作者檔案，並顯示標題、作者
45. 點選其中一篇論文	
	46. 讀取論文序號

	47. 依據論文序號從該論文資料夾中讀取其全文資料，並顯示之 48. 顯示『接受』與『拒絕』按鈕
49. 點選接受	
	50. 從該論文的關鍵字檔案讀取關鍵字資料(包含領域類別名稱、議題名稱、解法名稱、技術列表)，並顯示領域類別路徑，詢問是否需要修改領域類別路徑
51. 點選否	
	52. 顯示論文的議題名稱，詢問是否需要修改
53. 點選否	
	54. 將論文資料夾搬移到『通過審核』資料夾 55. 在論文資料夾中加入 add.done 檔案 56. 顯示『審核完畢』

- 將論文加入知識庫

Precondition：有論文已經通過審核且其建議新增的領域類別、議題都已經加入系統，Committer 在使用系統時，會出現『加入論文』連結

Committer	系統
26. 點選[加入論文]	
	27. 從『通過審核』資料夾中，找出有 add.done 檔案的論文資料夾，並從該論文資料夾的標題作者檔案讀取論文的標題、作者
28. 點選其中一篇報告	
	29. 顯示『加入』按鈕
30. 點選加入	
	31. 從關鍵字檔案讀取領域類別路徑、議題名稱、解法名稱、技術名稱列表等資料，並依據路徑與議題名稱找到技術議題/Survey 議題節點 32. 若是技術論文，則依據解法名稱從議題節點的解法列表取得解法節點，並將解法節點設爲父節點 33. 從父節點的報告/論文列表，找出新的論文編碼 34. 建立論文節點，將標題、作者、期刊、Volume、Issue、年代、論文編碼等資料填入 35. 利用論文編碼，計算出摘要檔名，並將論文資料夾的摘要檔案更改檔名至計算出來的摘要檔名 36. 利用論文編碼，計算出全文檔名，並將論文資料夾的全文檔案更改檔名至計算出來的全文檔名

	37. 從父節點取得節點編碼，透過節點編碼計算出論文/報告列表檔名，將論文資料寫入報告/報告列表檔案 38. 依據論文所使用的解法，找出解法的逆向查詢資料，並將論文編碼加入該資料中，並以編碼順序排序後，將逆向查詢資料寫回檔案 39. 依據論文所使用的技術，找出技術的逆向查詢資料，並將論文編碼加入該資料中，並以編碼順序排序後，將逆向查詢資料寫回檔案 40. 依據論文的作者，找出該作者的逆向查詢資料，將論文編碼加入該資料後寫回檔案 41. 依據論文的期刊、Volume、Issue，找出該期刊、Volume、Issue的逆向查詢資料，將報告編碼加入該資料後寫回檔案 42. 刪除該論文資料夾 43. 將論文節點指標加入其父節點的論文/報告節點指標陣列之中，並更新父節點的論文/報告數量 44. 將父節點的資料寫入檔案 45. 顯示『已成功加入論文』

- **審核書本知識**

Precondition：如果系統有尚待審核的書本知識，則在 Committer 使用系統時，會出現『審核書本知識』

Committer	系統
1. 點選[審核書本知識]	
	2. 從『待審核書本』目錄中，讀取每一個書本資料夾的標題作者檔案，並顯示標題、作者
3. 點選其中一本書本	
	4. 讀取書本序號 5. 依據書本序號從該書本資料夾中讀取其摘要，並顯示之 6. 顯示『接受』與『拒絕』按鈕
7. 點選接受	
	8. 從該書本的關鍵字檔案讀取關鍵字資料，並顯示領域類別路徑，詢問是否需要修改領域類別路徑
9. 點選否	
	10. 將書本資料夾搬移到『通過審核』資料夾 11. 在書本資料夾中加入 add.done 檔案 12. 顯示『審核完畢』

- **將書本知識加入知識庫**

Precondition:有書本知識已經通過審核且其建議新增的領域類別已經加入系統，Committer 在使用系統時，會出現『加入書本知識』連結

Committer	系統
1. 點選[加入書本知識]	
	2. 從『通過審核』資料夾中，找出有 add.done 檔案的書本資料夾，並從該書本資料夾的標題作者檔案讀取書本的標題、作者
3. 點選其中一篇書本	
	4. 顯示『加入』按鈕
5. 點選加入	
	6. 從關鍵字檔案讀取領域類別路徑，並依據路徑找到該領域類別節點 7. 從領域類別節點的書本列表，找出新的書本編碼 8. 建立書本節點，將標題、作者、出版社、年代、書本編碼等資料填入 9. 利用書本編碼，計算出摘要檔名，並將書本資料夾的摘要檔案更改檔名至計算出來的摘要檔名 10. 利用書本編碼，計算出書本議題檔名，並將書本資料夾的書本議題檔案更改檔名至計算出來的書本議題檔名 11. 利用書本編碼，計算出全文檔名，並將書本資料夾的全文檔案更改檔名至計算出來的全文檔名 12. 從領域類別節點取得節點編碼，透過節點編碼計算出書本列表檔名，將書本資料寫入書本列表檔案 13. 依據書本所列舉的書本議題，找出議題的逆向查詢資料，並將論文編碼加入該資料中，並以編碼順序排序後，將逆向查詢資料寫回檔案 14. 依據書本的作者，找出該作者的逆向查詢資料，將書本編碼加入該資料後寫回檔案 15. 刪除該報告資料夾 16. 將書本節點指標加入領域類別節點的書本節點指標陣列之中，並更新父領域的書本數量 17. 將父領域的資料寫入檔案 18. 顯示『已成功加入書本』

系統設計

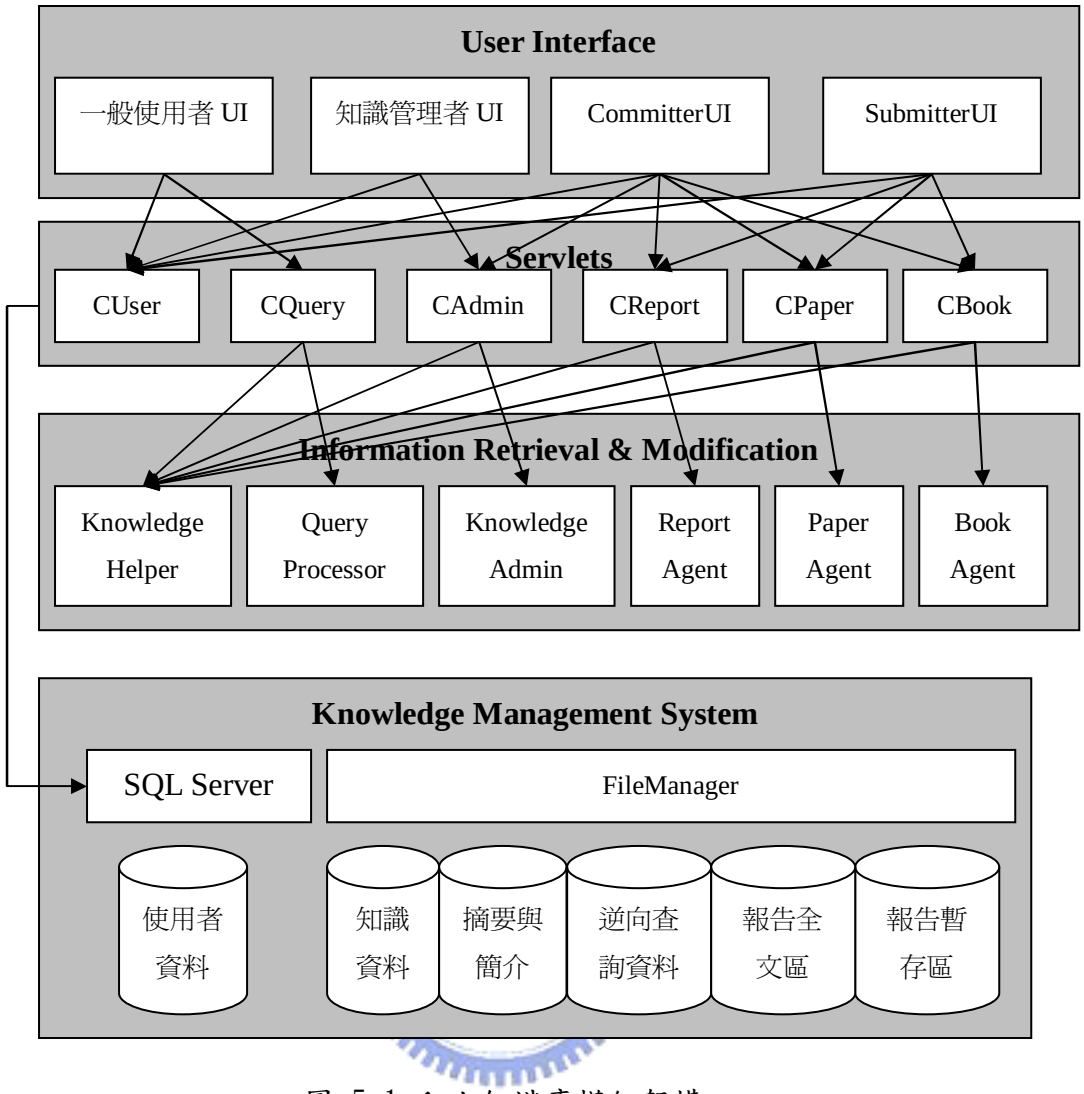


圖 5-1 私人知識庫模組架構

圖 5-1 為私人知識庫的模組架構，Servlets Layer 多了 CPaper 及 CBook 兩個類別，負責處理論文/書本知識的提交/新增，CQuery 還增加一些命令，這些命令如表 5-2 所示。

Class	Action 參數	參數意義	Actor	使用案例
CQuery	searchAuthorBook	查詢某作者的書本知識		利用作者查詢書本
	searchBkIssue	查詢運用討論某議題的書本知識		利用書本議題關鍵字查詢書本知識
	searchJournalPaper	查詢運用某解法的報告		依據期刊、Volume、Issue 查詢論文
CBook	submitBook	提交書本知識	Submitter	提交新書本知

				識
	listSubmittedBook	列出提交的書本知識	Committer	審核書本知識
	reviewBook	閱讀審核書本知識		
	rejectBook	拒絕書本知識		
	commitBook	接受書本知識		
	addBook	將書本知識加入知識庫		將書本知識加入知識庫
CPaper	submitPaper	提交論文	Submitter	提交新技術論文、提交新 Survey 論文
	listSubmittedPaper	列出提交的論文	Committer	審核論文
	reviewPaper	閱讀審核論文		
	rejectPaper	拒絕論文		
	commitPaper	接受論文		
	addPaper	將論文加入知識庫		將論文加入知識庫

表 5-2 私人知識庫 Servlet 新增的命令

第三層 Information Retrieval & Modification Layer 新增 PaperAgent、BookAgent 以處理論文及書本的提交與審核動作，另外 KnowledgeHelper 類別增加了 bkIssueTbl 欄位以儲存書本議題關鍵字列表。另外，在領域類別架構樹，增加 TechniquePaper、SurveyPaper、Book、BookIssue 四個類別，而 Category 類別增加 bookCount、bookList 以儲存書本知識數量及列表，而領域類別的硬碟儲存格式亦加上書本知識數量欄位。

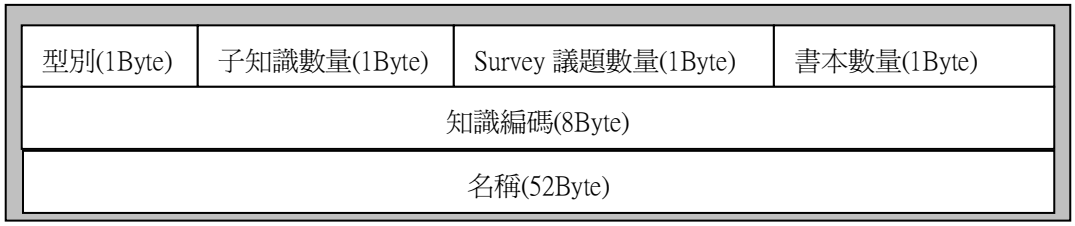


圖 5-2 領域類別節點儲存在硬碟的資料格式

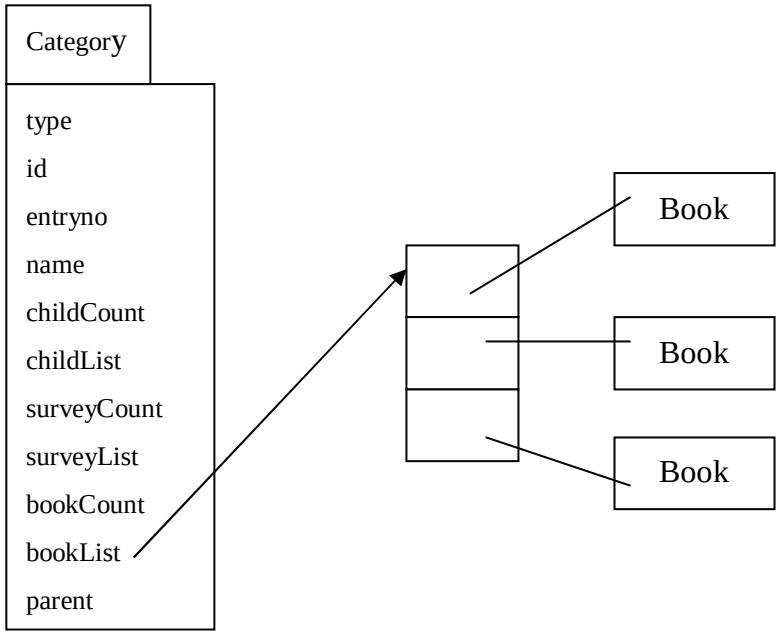


圖 5-3 領域類別節點記憶體儲存資料示意圖

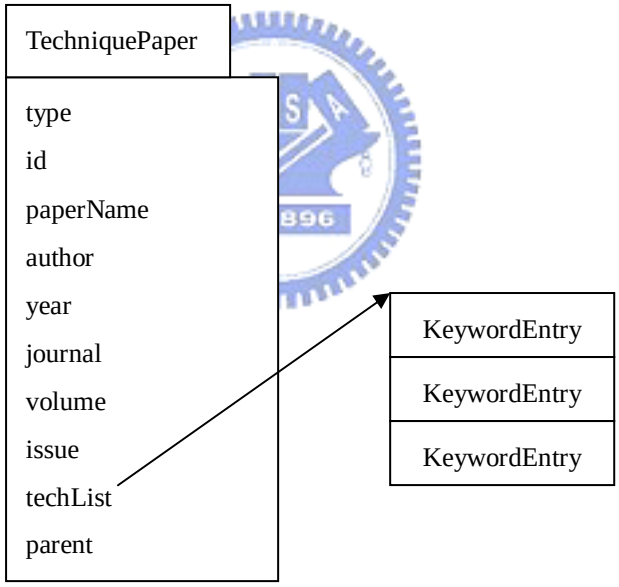


圖 5-4 技術報告節點儲存資料示意圖

圖 4-17 為技術論文的儲存在記憶體中的示意圖，type 為知識節點型態，其值為『技術論文』，id 為技術論文的知識編碼，paperName 為論文名稱，author 為作者名稱，year 代表年代，journal 為期刊代號，volume 為該篇論文位於第幾 volume，issue 為該篇論文所在的 issue，techList 為該報告所使用的技術列表，parent 為該技術議題的解法節點指標。

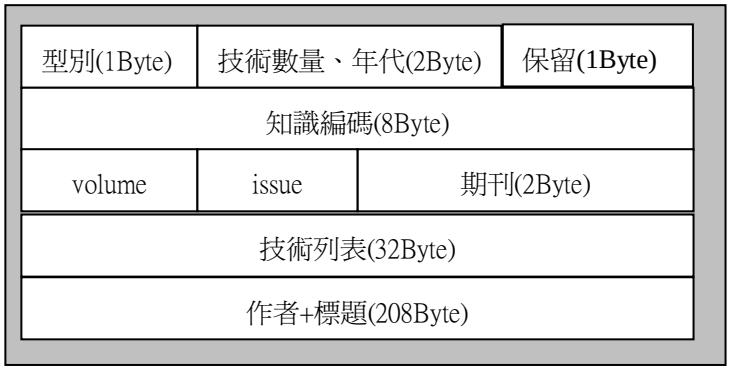


圖 5-5 技術論文儲存在硬碟中的格式

圖 4-18 為技術論文儲存在硬碟中的格式，其內容包括：

7. 型別為 1Byte
8. 技術數量以 4Bit 儲存，年代以 12Bit 儲存，共 2Byte。
9. 報告的知識編碼以 8Byte 儲存
10. volume 以 1Byte 儲存
11. issue 以 1Byte 儲存
12. 期刊代號以 2Byte 儲存
13. 技術關鍵字的編碼佔 4Byte，如第三章所述，一篇報告的技術最多 8 個，以 32Byte 儲存技術列表。
14. 標題、作者名稱長度 208Byte，首先儲存作者，作者以逗號分隔，最後加一個 0，接著儲存標題。

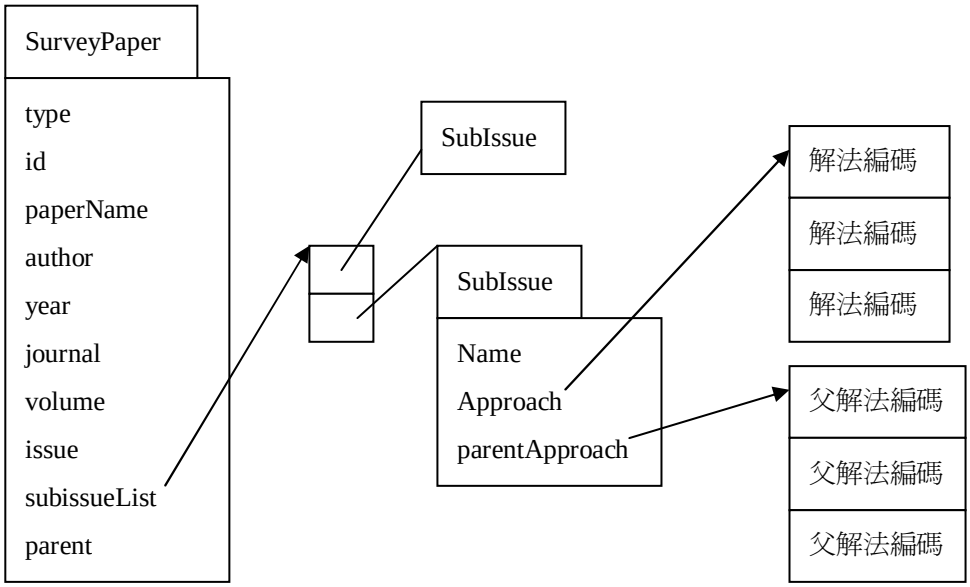


圖 5-6 Survey 論文節點儲存資料示意圖

圖 4-21 為 Survey 論文的儲存在記憶體中的示意圖，id、paperName、author、year、journal、volume、issue 等資料與技術論文節點相同，type 為知識節點型別，以 Survey 論文而言，其值為『Surve 論文』，subissueList 代表其子議題列表，為一 SubIssue 物件陣列，parent 為 Survey 報告的 Survey 議題節點指標。圖 4-22 為其儲存在硬碟中的格式。

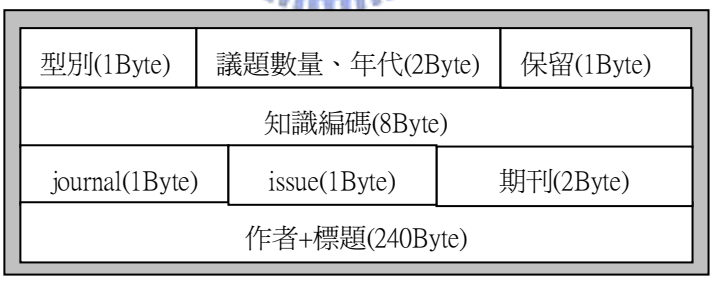


圖 5-7 Survey 報告儲存在硬碟中的格式

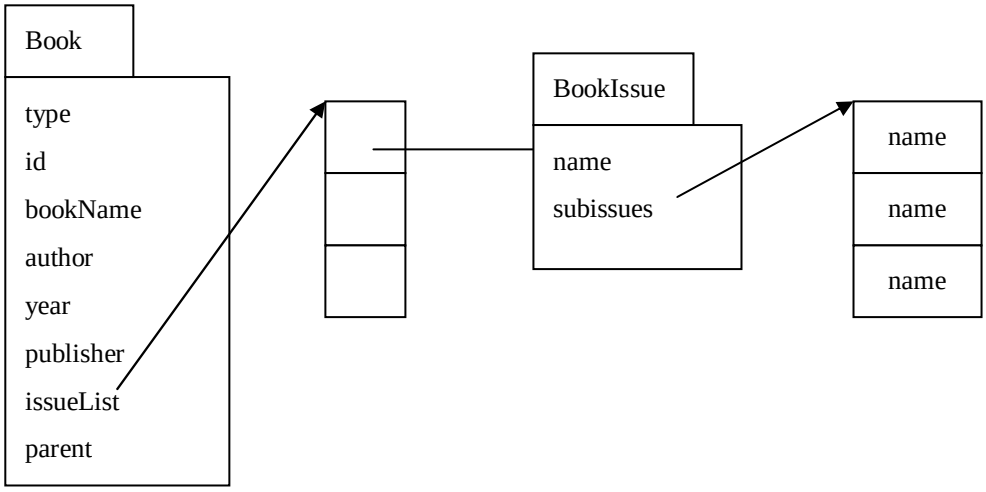


圖 5-8 書本知識節點儲存資料示意圖

圖 5-8 為書本知識的儲存在記憶體中的示意圖，id、author、year 等資料與技術論文節點相同，type 為知識節點型別，其值為『書本知識』，bookName 為書本標題，publisher 為出版社，issueList 代表其書本議題列表，為一 BookIssue 物件陣列，書本議題以章節為主，第一層為章(Chapter)，第二層為節(Section)，因此 BookIssue 物件中，name 為 chapter 的名稱，subissues 為該 chapter 之下的所有 section 名稱；parent 為 Survey 報告的 Survey 議題節點指標。圖 5-9 為其儲存在硬碟中的格式，其內容包括：

1. 型別為 1Byte
2. 年代以 2Byte 儲存。
3. 知識編碼以 8Byte 儲存
4. 出版社代號以 2Byte 儲存
5. 標題、作者名稱長度 240Byte，首先儲存作者，作者以逗號分隔，最後加一個 0，接著儲存標題。

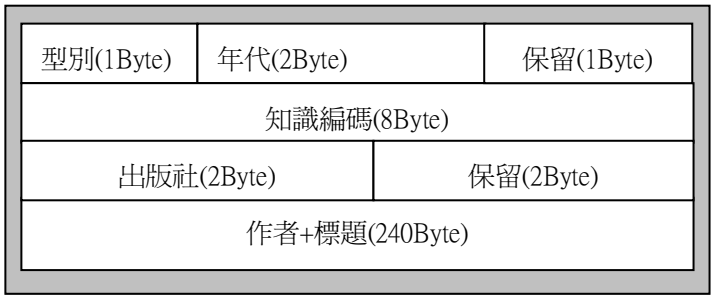


圖 5-9 書本知識儲存在硬碟中的格式

類別	方法
PaperAgent	SubmitTechniquePaper SubmitSurveyPaper listSubmittedPaper listCommittedPaper commitPaper addPaper
BookAgent	SubmitBook listSubmittedBook listCommittedBook commitBook addBook

表 5-3 PaperAgent、BookAgent 的方法列表

表 5-3 為 PaperAgent 與 BookAgent 的方法列表，這些方法，與 ReportAgent 的方法類似，故不再贅述。QueryProcessor 類別多了兩個方法：

1. SearchAuthorBook(author)：author 為要查詢的作者名稱，從該作者名稱所對應的逆向查詢資料檔案中讀取書本編碼，並透過 KnowledgeHelper 取得對應的書本節點。
2. SearchJournalPaper(journal, volume, issue)：journal 為期刊代號，從該期刊該 volume 該 issue 的逆向查詢資料檔案中讀取論文編碼，並透過 KnowledgeHelper 取得對應的論文節點。

第四層 Knowledge Management System 沒有新增類別，但是其主要類別 FileManager 多了四個方法：

1. writeBookData(Book)：將書本知識資料寫入檔案
2. updateBookInvertedFile(Book)：將書本的逆向查詢資料更新，此方法用在新增書本知識
3. updateBookInvertedFile(Book, oldid)：將報告的逆向查詢資料更新，此方法用在搬移書本知識
4. openJournalFile(journal, volume, issue)：此方法會開啟該期刊該 volume 該 issue 所對應的逆向查詢檔案，並回傳此檔案的 RandomAccessFile 物件。
5. closeJournalFile(RandomAccessFile f)：將 openJournalFile 回傳的檔案關閉。

5.2節 大型私人知識庫之設計

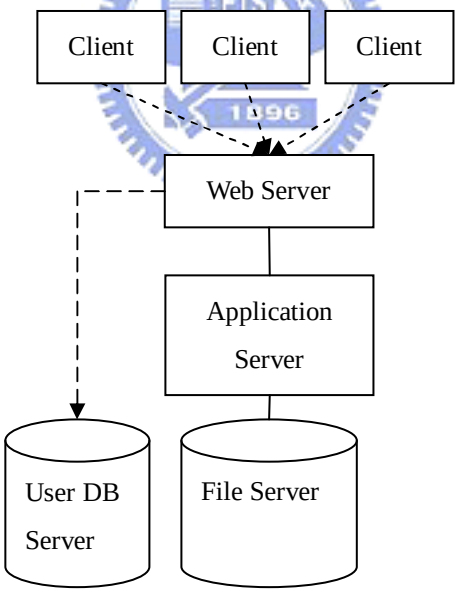


圖 5-10 小型知識庫之架構

本知識庫採取三層式架構，如圖 5-10 所示，Web Server 負責顯示使用者 UI 與使用者互動，使用者的查詢與新增知識等等動作，由 Web Server 交給 Application Server 負責處理，Application Server 負責存取 File Server 中的知識資料，File Server 可以是 Application Server 伺服器中本身的硬碟，

也可以是網路檔案系統(如 Sun Microsystem [19]的 NFS(Network File System)[20])，由 Application Server 掛載(mount)，而 User DB Server 儲存使用者資料。

當知識量越來越多時,若硬碟儲存空間不夠,可加掛硬碟,甚至增加多台 File Server 儲存知識；然而當知識庫的領域類別樹狀圖及關鍵字列表所需記憶體超過 Application Server 所能負荷的容量時,則需佈建多台 Application Server，並將知識資料分散在這些 File Server,圖 5-11 為大型私人知識庫的架構。3.3.3 節曾經探討如何切割知識庫，提到知識庫可以『領域類別』為基礎切割，在大型知識庫中，由於最上層領域類別幾乎不會更動，因此可在 Web Server 放置『第一層領域類別列表』及『領域類別與 Application Server 的對應表』。

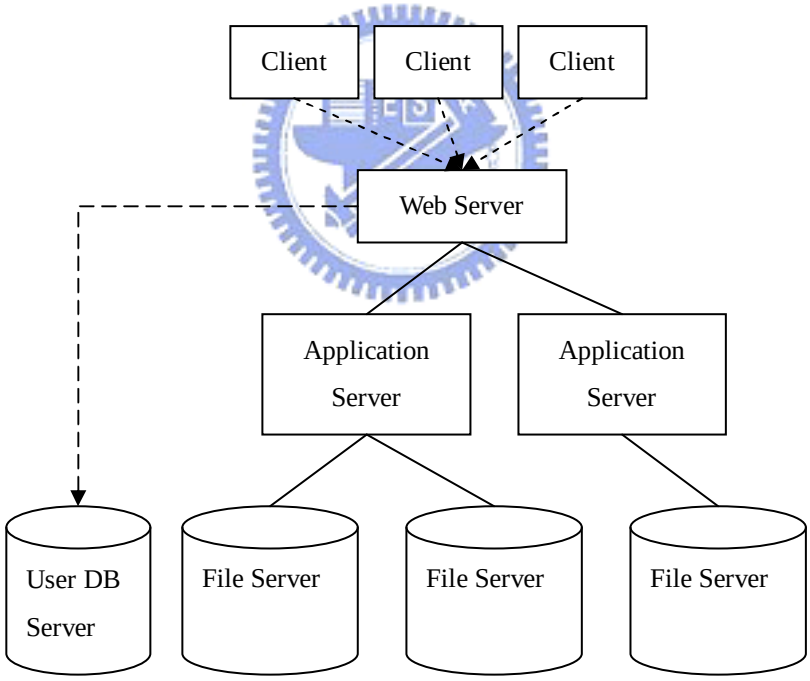


圖 5-11 大型私人知識庫的架構

圖 5-12 為大型私人知識庫的模組架構，Web Server 配置 User Interface Layer、Servlets Layer、及 Information Retrieval & Modification Layer，Application Server 配置 Information Retrieval & Modification Layer 及 Knowledge Management Layer，Web Server 與 Application Server 透過 JAVA RMI

溝通，因此 Information Retrieval & Modification Layer 的每個類別都增加 Remote 類別，如 KnowledgeHelperRemote、QueryProcessorRemote、BookAgent Remote、PaperAgentRemote、ReportAgentRemote、及 KnowledgeAdminRemote，Web Server 的 Remote 類別扮演 Stub 的腳色，當呼叫這些 Remote 類別的 Stub 物件之方法時，透過網路喚起 Application Server 對應物件的方法，並經由 JAVA RMI 機制將結果傳回 Web Server。

將知識資料分散在 Application Server 之後，使用者在查詢資料時，可分為兩種情況：正向查詢、反向查詢，正向查詢只需要查單一知識的資料，Web Server 可透過『領域類別與 Application Server 的對應表』可得知該向哪個 Application Server 查詢，便呼叫該 Application Server 的 Remote Stub 物件以取得知識資料。

反向查詢(如利用解法名稱、技術、作者等查詢知識)，Web Server 則需要呼叫指定範圍內 Application Server 的 Remote Stub 物件，並整合 Application Server 傳回的結果。舉 QueryProcessor 來說明，當使用者要查詢運用某種解法的知識時，Web Server 的 QueryProcessor 根據使用者指定的領域範圍，可得知需要向哪些 Application Server 查詢資料，並且透過這些 Application Server 的 QueryProcessorRemote Stub 物件呼叫其 SearchKeyword 方法，經由 JAVA RMI 喚起 Application Server 對應的 QueryProcessorRemote 物件的 SearchKeyword 方法，QueryProcessor 物件組合結果並傳遞給 User Interface Layer 顯示。

由於知識資料分散在一至多個 File Server，因此 Application Server 需要儲存『領域類別與 File Server 的對應表格』，當 Application Server 需存取 File Server 知識時，透過 FileManager 存取，FileManager 透過該表格從對應的 File Server 中存取知識資料。由於 Application Server 以網路檔案系統的

方式掛載File Server，因此對於FileManager而言，皆為透過檔案系統方式存取資料，因此『領域類別與File Server的對應表格』只需要紀錄檔案路徑即可。

這種設計僅需在Web Server增加『領域類別與Application Server的對應表格』及『第一層領域類別列表』，因此增加多台Web Server相當容易。

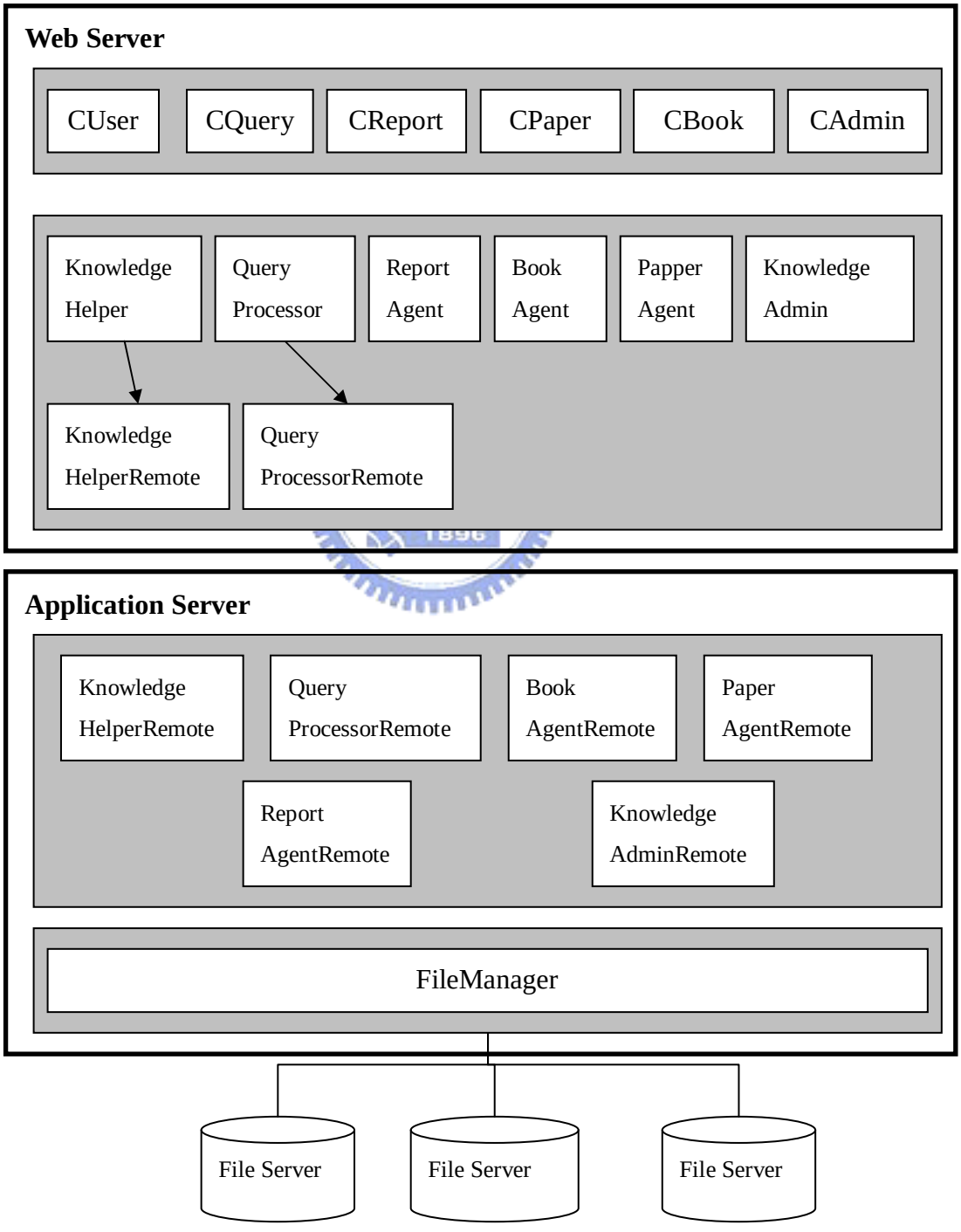


圖 5-12 大型私人知識庫模組架構

5.3節 期刊資料庫之設計

期刊資料庫的知識量通常較為龐大，需要採取類似大型私人知識庫的架構，期刊資料庫與大型私人知識庫的最大差異在於其提交知識的流程，私人知識庫提交知識時，只要 Committer 負責審核即可，然而期刊論文需要多位 Reviewer 審核其論文，審核通過後，需要 Editor 負責將論文套上期刊的樣版(如加上期刊標誌、名稱、Volume、Issue、頁次等等)，然後在交由 Committer 負責將編輯好的論文上傳。此部分需要編輯系統，而編輯系統已經相當完善，且期刊資料庫的查詢系統也相當成熟，因此期刊資料庫的知識庫系統可分為兩套:『現有查詢系統』與『問題-解法系統』。使用者在查詢時，可先選擇要使用哪套系統，當選擇『現有查詢系統』時，則使用期刊資料庫原本的系統，當使用『問題-解法系統』時，則使用本論文所實作的系統。

由上述可知，期刊資料庫與大型私人知識庫的差異在於：

3. 期刊資料庫沒有書本知識及報告
4. 不需實作期刊資料庫現有的查詢方法，如透過作者名稱查詢論文
5. 需要透過某種機制與現有系統整合

第一點差異使期刊資料庫不需要 CBook、CReport、BookAgent、ReportAgent 等類別及相關的使用者 UI；第二點則需在 CQuery 類別不需處理這些查詢方法；第三點，由於期刊資料庫現有系統已經有全文檔案、摘要檔案，因此不需在『問題-解法』系統存放這些檔案，目前期刊資料庫的論文資料，均可透過 DOI 機制[21]查詢，因此在『問題-解法』系統中，需要儲存論文的 DOI 編碼，當使用者需要下載論文時,『問題-解法』系統便透過 DOI 編碼，向『現有系統』取得論文，因此在 Information Retrieval & Modification Layer 需增加 DOIAgent 類別，負責透過 DOI 機制取得論文全文。

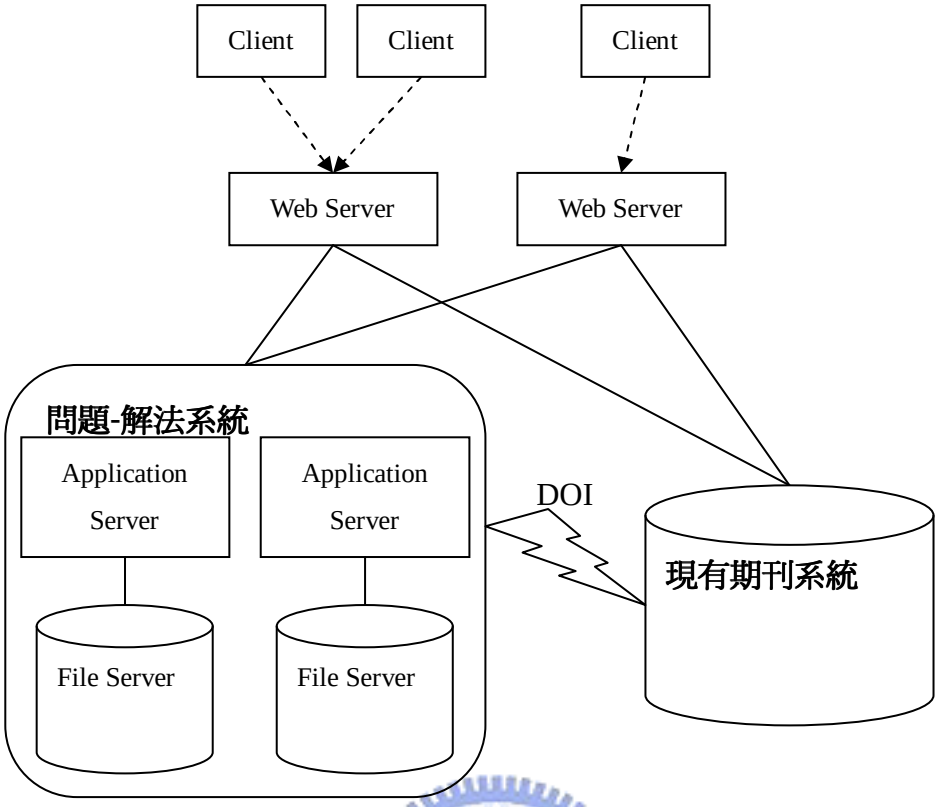


圖 5-13 期刊資料庫系統架構

圖 3-20 為期刊資料的系統架構，使用者在進入 Web Server 時，可選擇使用『現有期刊系統』或『問題-解法系統』，當使用者要下載論文全文時，『問題-解法』系統便透過 DOI 機制向現有期刊系統取得論文全文。

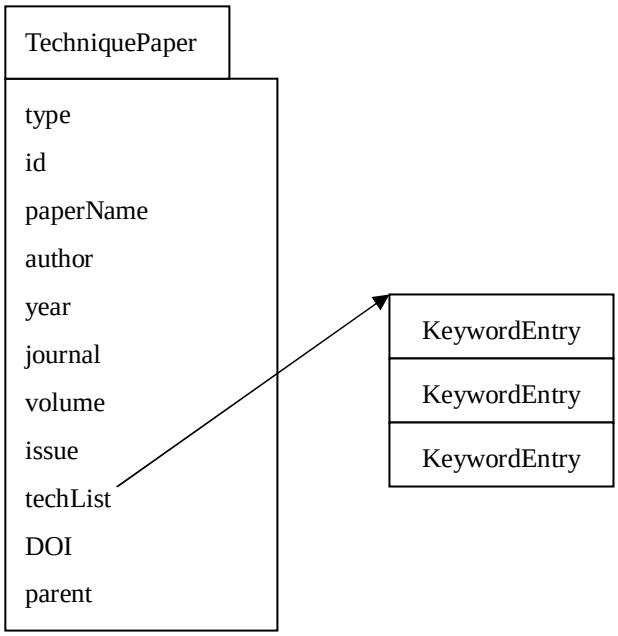


圖 5-14 技術報告節點儲存資料示意圖

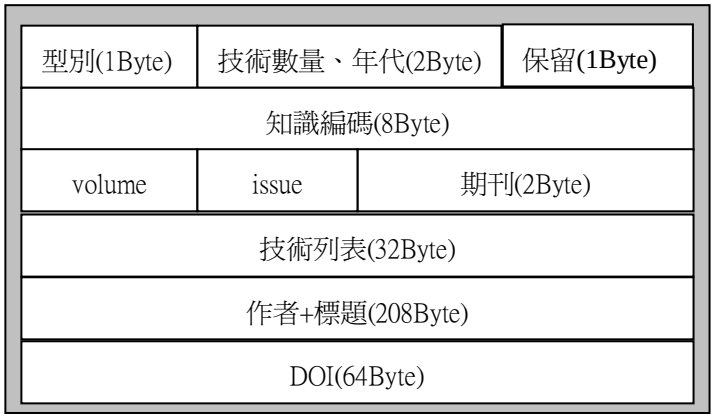


圖 5-15 技術論文儲存在硬碟中的格式

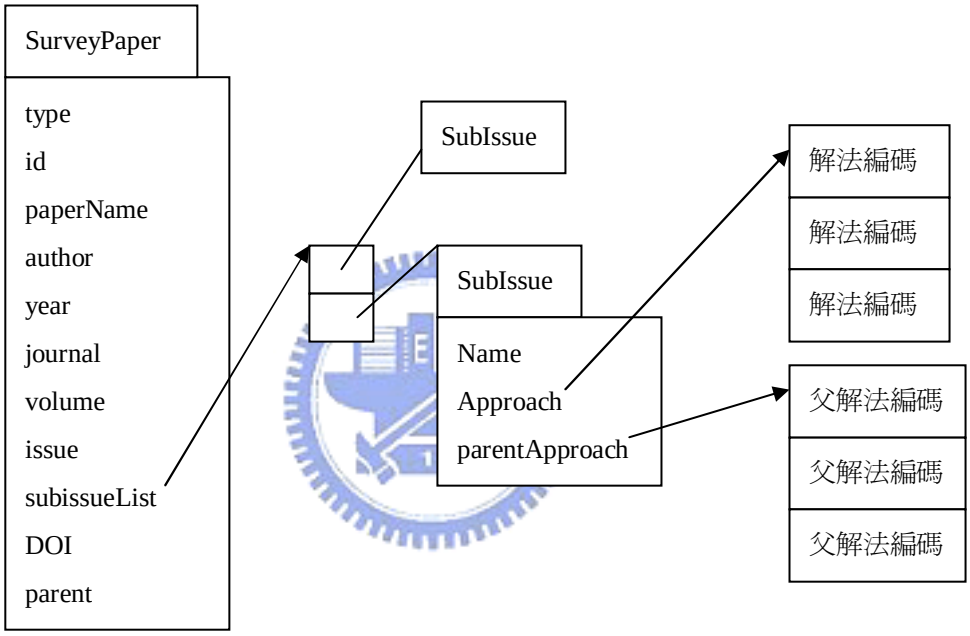


圖 5-16 Survey 論文節點儲存資料示意圖

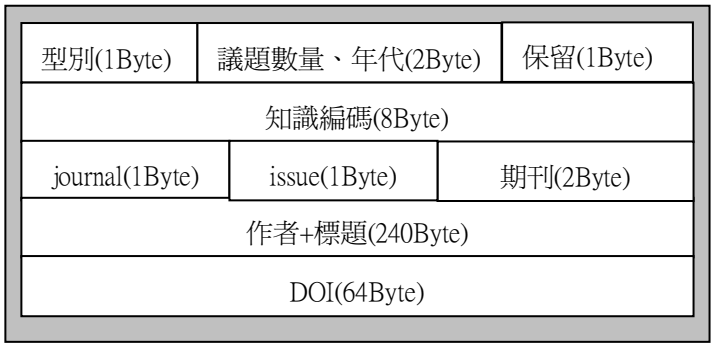


圖 5-17 Survey 報告儲存在硬碟中的格式

圖 5-14、圖 5-15、圖 5-16、圖 5-17 分別為技術論文儲存在記憶體中的示意圖、技術論文儲存在硬碟中的格式、Survey 論文儲存在記憶體中的示意圖、

Survey 論文儲存在硬碟中的格式，與私人知識庫的論文格式差異在於增加了 DOI 編碼欄位，DOI 編碼可分為兩個部分：Prefix、Suffix，Prefix(如 10.1009)中的 10 代表 DOI 的識別名稱，此部份固定為 10，1009 代表期刊出版商代碼，長度在 DOI 的規範中沒有設定上限，不過其字元都必需為數字；Suffix(如 TSE.2005.131)則由出版商自行編碼，例如 IEEE 的登記編碼為 1109，IEEE 對 Braberman 等人的”A Scenario-Matching Approach to the Description and Model Checking of Real-Time Properties”所給予的字尾編碼為 TSE.2005.131，因此此篇論文的 DOI 編碼為 10.1109/ TSE.2005.131。IEEE 以 TSE 代表 Transactions on Software Engineering，2005 代表其發表年代為 2005 年，131 是該篇論文的序號。DOI 編碼在單一期刊資料庫中，其 Prefix 固定，故不需儲存 Prefix，只要儲存 Suffix 即可，因此以 64Byte 儲存論文 DOI 編碼。

圖 5-18 為期刊資料庫的模組架構，在 Application Server 中增加了 DOIAgent，當使用者要取得某篇論文時，Web Server 由 KnowledgHelper 喚起 Application Server 的 KnowledgeHelperRemote 物件，然後 KnowledgeHelper Remote 物件透過 DOIAgent 向『現有期刊系統』取得論文全文。

除了『問題-解法』系統需要修改之外，現有系統的編輯系統也需要做下列調整：

1. 關鍵字需採用『問題-解法』知識庫的關鍵字架構，讓論文作者提交論文時，使用這些關鍵字來定義論文關鍵字
2. Committer 加入論文到系統時，現有系統需將論文作者、標題、期刊、Volume、Issue、DOI 編碼、及其關鍵字傳送到『問題-解法』系統

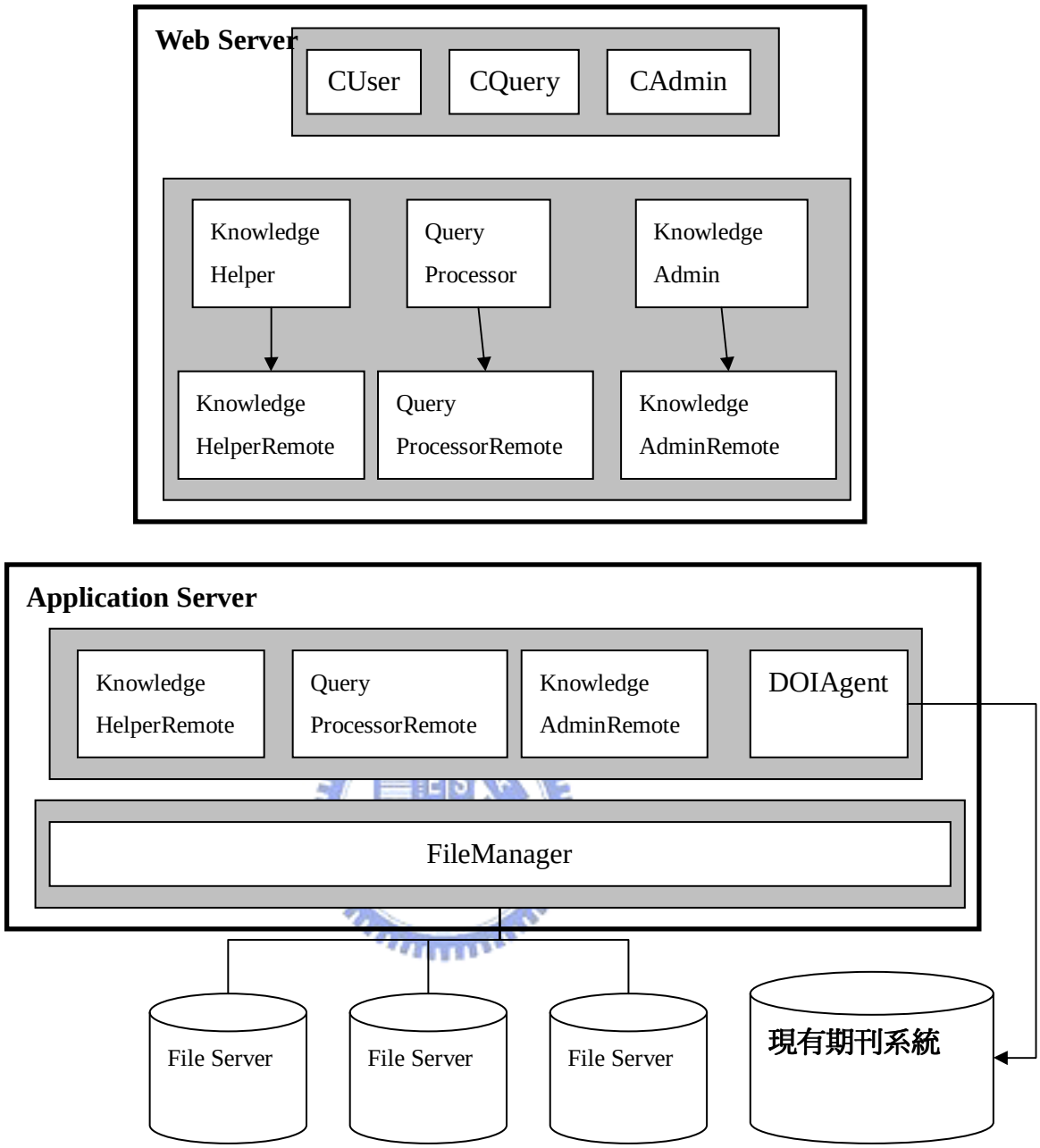


圖 5-18 期刊資料庫模組架構

5.4節 全文資料庫之設計

全文資料庫與期刊資料庫的差異在於知識的增加與修改，由廠商負責更新，因此知識管理者需要負責處理更新及新增知識的部分，在此假設全文資料庫的上游廠商採用『問題-解法』知識分類法。

由於雙方知識庫的分類不盡相同，因此需要做轉換的機制。本系統利用 XML 來描述分類如何轉換，如圖 5-19 所示，該 XML 檔案的根節點為 mappings，mappings 標籤需包含一至多個 mapping 以表示多個轉換關係，mapping 標籤需包含 from 與 to 標籤，而 from 與 to 需包含一至多個 category 標籤，以此範例來說，則是從 Software 領域轉換到 Computer/Software 領域。

```
<mappings>
<mapping>
    <from>
        <category>Software</category>
    </from>
    <to>
        <category>Computer</category>
        <category>Software</category>
    </to>
<mapping>
<mapping>…</mapping>
</mappings>
```

圖 5-19 XML 轉換格式範例

第6章 結論

本實驗室學長謝祖望提出『問題-解法』知識分類法，將軟體知識依「領域類別」(Category)、「議題」(Issue)、「解法方法」(Approach)、「技術」(Technique)四類關鍵字來作分類，並實作了一雛型（Prototype）知識庫，然而研究並未針對 Survey 性質的論文/報告納入，且其雛型系統有下列缺點：

1. 缺少逆向查詢功能
2. 使用者分類不清
3. 使用單一伺服器，無法處理大量知識資料以及大量查詢
4. 將索引資料全部載入至記憶體，無法適用大型量知識庫
5. 缺少知識庫常用的查詢方法

該研究與其雛型系統有上述缺點，因此無法實用；本研究針對其缺點，提出下列解決方法：

1. 分析 Survey 類型論文/報告的性質，發現其關鍵字可分為『領域類別』、『Survey 議題』。
2. 分析技術報告庫、私人知識庫、期刊資料庫、全文資料庫等不同類型知識庫的使用方式及其使用者的行為模式，提出符合該使用模式的系統架構及使用流程。
3. 提出『問題-解法』軟體知識分類法關鍵字的逆向查詢機制，此機制可應用於分散式知識庫作查詢。
4. 針對『問題-解法』關鍵字架構進行分析，將常用索引載入至記憶體，當索引數量無法在單一電腦儲存時，則透過分散式架構處理。
5. 針對私人知識庫、技術報告庫，提出系統如何實作現有軟體知識庫常用查詢方法，針對期刊資料庫、全文資料庫，提出如何將『問題-解法系

統』與現有系統進行整合

由於知識庫系統架構龐大複雜，限於時間與研究人力的關係，此知識分類方法在實作上仍有許多可再加強的地方：

1. 建構標準化的知識描述交換格式

當期刊資料庫採用『問題-解法』系統時，可透過以 XML（Extensible Markup Language）為基礎的規格，讓其他訂閱期刊資料庫的私人知識庫及全文資料庫，透過此 XML 檔案更新知識，並藉由 DOI 機制取得論文全文。

2. 現有資料庫採用『問題-解法』知識庫的輔助轉換機制

由於現有資料庫沒有採用『問題-解法』關鍵字，但是當使用者想要透過『問題-解法』關鍵字查詢舊有論文/報告時，若沒有針對舊有論文輸入其領域類別、議題等關鍵字，則無法查詢，但是純人工輸入關鍵字耗時耗力，可分析舊有論文的關鍵字，得知該篇論文大約在哪些領域之下，並可藉由關鍵字的特性，分析出該論文的解法、技術關鍵字有哪些，然後在藉由人工審核及修改以達轉換的目的。

參考文獻

[1] ACM Digital Library, http://portal.acm.org/dl.cfm

[2] IEEE Xplore, http://ieeexplore.ieee.org/Xplore/conhome.jsp

[3] JSTOR - The Scholarly Journal Archive, http://www.jstor.org/

[4] Project MUSE, http://muse.jhu.edu/about/index.html

[5] Springer Link, http://www.springerlink.com/

[6] 謝祖望, “以『問題-解法』為基礎的軟體知識分類法”, 中華民國九十三年六月

[7] NASA Technical Reports Server, http://ntrs.nasa.gov/

[8] SCE Technical Reports, http://reports-archive.adm.cs.cmu.edu/

[9] SCS Technical Report Collection, http://reports-archive.adm.cs.cmu.edu/

[10] George A. Miller, “The Magical Number Seven, Plus or Minus Two: Some Limits on Our Capacity for Processing Information”, The Psychological Review, 1956, vol. 63, pp. 81-97

[11] Fabio Kon, ”Distributed File Systems Past, Present, and Future - A Distributed File System for 2006”,March 6, 1996

[12] Simonetta Balsamo, Antinisca Di Marco, Paola Inverardi, Marta Simeoni, “Model-based Performance Prediction in Software Development: A Survey”, IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, Vol. 30, No. 5, MAY 2004

[13] Minoru Etoh and Takeshi Yoshimura, “Wireless Video Applications In 3G and Beyond”, IEEE Wireless Communications, August 2005

[14] Extensible Markup Language, http://www.w3.org/XML/

[15] Apache Tomcat，http://tomcat.apache.org/

[16] MySQL，http://mysql.com

[17] Z. Aral, J. Bloom, T. Doeppner, I. Gertner, A. Langerman, &

G.Scharffer, ”Variable Weight Processes with Flexible Shared Resources,” USENIX Association Conference Proceedings, pp.405-412, January 1989.

[18] W. T. Marshall,”A Unified Approach to the Evaluation of a Class of ‘Working Set Like’ Replacement Algorithms,” PhD Thesis, Department of Computer Engineering, Case Western Reserve University, Cleveland, OH, May 1979.

[19] Sun Microsystem，http://www.sun.com/

[20] Network File System (protocol)，
http://en.wikipedia.org/wiki/Network_File_System_%28protocol%29

[21] Digital Object Identifier，http://www.doi.org