新北市政府 101 年度自行研究報告

# 電子病歷互通平台應用中文特殊字型解決方案之研究

研究機關：新北市立聯合醫院

研究人員：林國中

研究期程：101.01.01~101.12.31

新北市政府衛生局編印

中華民國 101 年 12 月 07 日

附件一

# 新北市政府年度自行研究計畫表

填 表 人：林國中
填表日期：101.12.07
聯絡電話：2982-9111#3256

| | | | |
|---|---|---|---|
| 計　畫　名　稱 | 電子病歷互通平台應用中文特殊字型解決方案之研究 | | |
| 研究機關及人員 | 新北市立聯合醫院 林國中 | 期　　程 | 自 101 年 01 月 01 日<br>至 101 年 12 月 31 日 |
| 目　　的 | 因電子病歷推行為國家重大醫療政策，行政院衛生署所屬醫療機關與各直轄市衛生局各醫療單位，均主動積極的申請加入電子病歷互通平台之資料交換與陸續通過檢查；但因中文字型有其特殊之代表意義與特性，因此各醫療單位均有自行使用中文字型造字之字碼區間，但是卻未進行標準化與統一，因此電子病歷互通平台上線後，在中文姓名、中文地址或是中文病摘的部分將會出現越來越的無法交換中文字體，而以◇、□等代替(如：王書◇、◇德友等)，也造成交換平台上姓名或是地址資料缺漏等現象將越來越頻繁。本計畫目的主要在解決現行醫療機構各單位針對罕見字與自行造字字碼不統一，當導入全國性之電子病歷交換平台系統時，各項單據與 Web 系統中取得民眾姓名與地址時，造成諸多遺漏或是不正確問題。 | | |
| 方　　法 | 首先將研究機構內自行造字字碼(如：醫療資訊系統內碼與 Unicode 碼對照清單轉入資料庫；其次針對外部 BIG-5 碼媒體交換碼區間與 Unicode 碼對照，並進行內碼區間統一；再來將自造字內碼與外部媒體交換內碼排除後，匯入中文推行委員會所提供之圖形自造字內碼與 URI 於資料庫，如此便可以透過 URI 取得電子病歷中特殊字型；最後在報表以及 Web 應用程式中進行測試，期望研發結果能解決各媒體交換與罕見字與自造字展現問題。 | | |
| 經　　費 | 585,300 | | |

備註：

一、 研究機關及人員：包括研究機關、實際研究人員及參與工作人員。

二、 方法：如研究方法之訂定、問題之發掘、研究設計、資料之蒐集與分析、解決方案之研擬、研究報告之提出。

三、 已提報之自行研究計畫因故撤銷辦理者，應敘明原因行文通知。

四、 自行研究報告內容應力求與所屬局處業務相關。

附件二

# 新北市政府 101 年度自行研究計畫執行情形季報表

填表日期：101.11.22

| 計畫名稱 | 研究機關及人員 | 期程 | | 執行情形概述 | 備註 |
|---|---|---|---|---|---|
| | | 起 | 訖 | | |
| 電子病歷互通平台應用中文特殊字型解決方案之研究 | 衛生局<br>資訊室 林國中 | 101 年 1 月 1 日 | 101 年 3 月 31 日 | 1. 完成研究流程、步驟、驗證方式確認。<br>2. 完成 10,000 標準字型檢索確認。<br>3. 完成 4,000 字特殊字型檢索。<br>4. 購入並取得 14,000 中文字型外觀與字碼對應。 | |
| 電子病歷互通平台應用中文特殊字型解決方案之研究 | 衛生局<br>資訊室 林國中 | 101 年 4 月 1 日 | 101 年 6 月 30 日 | 1.完成中文字型展現相關研究文獻探討與整理分析。<br>2.已購入特殊字型之基本素材，進行應用分析，並釐清常用字體與特殊字體之使用區間(第 1 季購入;第 2 季應用分析)。<br>3.並完成字碼對照表之設計等重要工作，可行性驗證。<br>4.進入字型轉換程式之圖形特殊字型之共用 API 開發，目前已完成 35%。<br>5.完成期中報告。 | |
| 電子病歷互通平台應用中文特殊字型解決方案之研究 | 衛生局<br>資訊室 林國中 | 101 年 7 月 1 日 | 101 年 9 月 30 日 | 1. 完成本院自行造字區(1000 餘字)與 CNS 字碼比對。<br>2. 與中文數位化推廣委員會完成 CNS 字碼全區的使用規則，並具體完成轉檔程式必要功能之確認。<br>3. 完程標準字型入資料庫之全量範圍檢索。<br>4. 轉檔程式開發 API 兩支完成並準備與中推會購入之 API | |

| | | | | | |
|---|---|---|---|---|---|
| | | | | 進行測試。<br>5. 本院 1000 字難字、特殊字之造字模型完成，並與中推會驗證。<br>6. 本研究之字型資料收集、整理完成 99.3%，並編入研究報告內(約 20,000 字)。<br>7. 中文數位化推行委員會呼叫網際網路資源使用 API 購入流程。<br>8. 結案報告準備完成 70%。 | |
| 電子病歷互通平台應用中文特殊字型解決方案之研究 | 衛生局<br>資訊室 林國中 | 101 年 10 月 1 日 | 101 年 12 月 31 日 | 1. 完成中推會 API 測試。<br>2. 完成中推會 API 與自行開發之程式互相呼叫樣本頁面。<br>3. 將 Unicode 18000 字與 BIG-5 字碼之標楷體與新細明體進行對照，並完成對照表。<br>4. 考量後續研究以及節樽費用，在不影響研究成果狀況下，將 URI 資料庫建置費用省下，改採用素材匯入 EXCEL 檔案方式辦理，原編列預算為 NT226,000；改變方式後可控制在 NT95,000 元以內。<br>5. 本計畫進度完成 100%<br>6. 進行研究結案報告撰寫，並將成果整理。<br>7. 期末報告準備。 | |

| | |
|---|---|
| 計畫名稱 | 電子病歷互通平台應用中文特殊字型解決方案之研究 |
| 期程 | 101.01.01~101.12.31 |
| 經費 | 585,300 |
| 關鍵字 | 電子病歷、圖形字體、中文字型 |
| 緣起與目的 | 因電子病歷推行為國家重大醫療政策,行政院衛生署所屬醫療機關與各直轄市衛生局各醫療單位,均主動積極的申請加入電子病歷互通平台之資料交換與陸續通過檢查;但因中文字型有其特殊之代表意義與特性,因此各醫療單位均有自行使用中文字型造字之字碼區間,但是卻未進行標準化與統一,因此電子病歷互通平台上線後,在中文姓名、中文地址或是中文病摘的部分將會出現越來越的無法交換中文字體而以◇、□等代替(如:王書◇、◇德友等),也造成交換平台上姓名或是地址資料缺漏等現象將越來越頻繁。本研究目的主要在解決現行醫療機構各單位針對罕見字與自行造字字碼不統一,當導入全國性之電子病歷交換平台系統時,各項單據與 WEB 系統中取得民眾姓名與地址時,造成諸多遺漏或是不正確問題。 |
| 方法與過程 | 首先將研究機構內自行造字字碼(如:醫療資訊系統內碼與 Unicode 碼對照清單轉入資料庫;其次針對外部 BIG-5 碼媒體交換碼區間與 Unicode 碼對照,並進行內碼區間統一;再來將自造字內碼與外部媒體交換內碼排除後,匯入中文推行委員會所提供之圖形自造字內碼與 URI 於資料庫,如此便可以透過 URI 取得電子病歷中特殊字型;最後在報表以及 WEB 應用程式中進行測試,期望研發結果能解決各媒體交換與罕見字與自造字展現問題。 |

| | |
|---|---|
| 研究發現及建議 | 透過本研究計畫之執行,可以解決歷年來當病患至醫療院所臨櫃辦理各項醫療表單或是證明時,常常發生因為姓名或是地址中文字型無法顯現,而發生醫療糾紛的情勢。並可以提供衛生署相關醫療院所對於特殊字型、難字或是罕見字的呈現議題,讓民眾得到更多、更正確的服務,所開立的證明有更佳的公證性。除此之外,本研究計畫所提作法可以將電子病歷交換之內容,透過交換,讓需要交換的醫療院所,可以得到更正確的交換資訊及擴大電子病歷互通交換平台的效益。 |
| 備註 | |

# 摘要

**關鍵字**：電子病歷、圖形字體、中文字型

## 研究緣起

因電子病歷推行為國家重大醫療政策，行政院衛生署所屬醫療機關與各直轄市衛生局各醫療單位，均主動積極的申請加入電子病歷互通平台之資料交換與陸續通過檢查；但因中文字型有其特殊之代表意義與特性，因此，各醫療單位均有自行使用中文字型造字之字碼區間，但是卻未進行標準化與統一，因此電子病歷互通平台上線後，在中文姓名、中文地址或是中文病摘的部分將會出現越來越多無法交換的中文字體，而以◇、□等代替(如：王書◇、◇德友等)，也造成交換平台上姓名或是地址資料缺漏等現象將越來越頻繁。本計畫目的主要在解決現行醫療機構各單位針對罕見字與自行造字字碼不統一，當導入全國性之電子病歷交換平台系統時，各項單據與 Web 系統中取得民眾姓名與地址時，造成諸多遺漏或是不正確問題。

## 研究方法

首先，將研究機構內自行造字字碼(如：醫療資訊系統內碼與 Unicode 碼對照清單轉入資料庫；其次，針對外部 BIG-5 碼媒體交換碼區間與 Unicode 碼對照，並進行內碼區間統一；再來，將自造字內碼與外部媒體交換內碼排除後，匯入中文推行委員會所提供之圖形自造字內碼與 URI 於資料庫，如此便可以透過 URI 取得電子病歷中特殊字型;最後,在報表以及 Web 應用程式中進行測試，

期望研發結果能解決各媒體交換與罕見字與自造字展現之問題。

## 重要發現

透過本研究計畫之執行，可以解決歷年來當病患至醫療院所臨櫃辦理各項醫療表單或是證明時，常常發生因為姓名或是地址中文字型無法顯現，而發生醫療糾紛的情勢。並可以提供衛生署相關醫療院所對於特殊字型、難字或是罕見字的呈現議題，讓民眾得到更多、更正確的服務，所開立的證明有更佳的公證性。除此之外，本研究計畫所提作法可以將電子病歷交換之內容，透過交換，讓需要交換的醫療院所，可以得到更正確的交換資訊及擴大電子病歷互通交換平台的效益。

## 主要建議

本研究主要在解決現行各家醫院導入電子病歷互通後，因原本各家醫院特殊中文字碼編碼不同導致病人之基本資料闕漏，導致批價、申報、資料交換問題，主要建議如下：

(1) 導入電子病歷交換後，各醫院特殊造字流程急需統一

導入電子病歷交換後，中文語系國家將面臨嚴重的資料交換缺字或是錯誤問題，主要是面臨新舊系統、不同語系、不同交換碼之轉換機制在國內導入不夠完備，導致病人資料會有缺漏或是錯誤的情形，發生頻率將越來越高；各醫院應該提早因應，否則在大量導入電子病歷交換後，張冠李戴的中文字，將造成許多不必要的法律問題。

(2) 當醫院專線異常無法使用線上圖形字庫之因應策略

線上圖形字庫因為需要透過網際網路傳輸，因此，當網際網路不存在時，亦將發生圖形字型無法存取的問題，因此各醫院自行建造圖形字型資料庫，將勢在必行。同時也因為醫院有別於一般機關，使用醫學影像傳輸之需求相當大，也因此是否有多餘的頻寬讓圖形字體不斷使用，又是另一項值得探討的主題。

(3) 特殊中文字型造字前需謹慎確認

依據新公告之 CNS 字體，實際上可以應用的中文字體已高達 7 萬字以上，因此特殊字型一般而言均可以在 CNS 字體中找到，若貿然自行造字，將又面臨現行造字區間是否和其他醫院重複的問題，且一旦錯誤的字碼一經交換，將在政府電子病歷平台上留下錯誤的資訊，這些資料都是已經經過醫事人員簽章以及醫療院所簽章的重大資訊，修正不易。

(4) 盡早推行特殊字型造字流程統一新系統開發將越有利

本研究使中文推行委員會所提供的API來進行線上圖形字型的引用，在Web頁面上之開發標準需要進行內崁圖形字庫引用之 API，也因此已經開發的新興 Web 語言，無論 Asp、.net、JSP、PHP 等語言開發方式，可能均需要採取相同的做法，因此所有已開發程式都需要改寫，也因此，越早導入，對新興 Web 架構之服務越有利。

# 目次

新北市政府 101 年度自行研究報告 ..... i

新北市政府年度自行研究計畫表 ..... iii

新北市政府 101 年度自行研究計畫執行情形季報表 ..... v

新北市政府 101 年度自行研究成果摘要表 ..... vii

摘要 ..... ix

目次 ..... xiii

CHAPTER ..... 1

1. 緒論 ..... 1

1.1 研究背景 ..... 1

1.2 目的說明 ..... 2

1.3 本計畫章節說明 ..... 3

2. 文獻探討 ..... 5

2.1 中文字型交換之背景 ..... 5

2.2 中文資訊交換方法 ..... 6

2.3 電子病歷交換現況 ..... 8

2.4 電子病歷交換內容與步驟 ..... 14

2.5 電子病歷中文字型轉換問題分析(與 2.1 節內容重複) ..... 17

2.6 本計畫與醫療保健之相關性 ..... 20

2.7 相關政策或法令 ..... 20

3. 研究方法 ..... 23
3.1 問題狀況與發展需求 ..... 23
3.2 醫療院所採用中文特殊字型造字方式 ..... 23
3.3 醫療院所現行採用轉碼方式 ..... 25
3.4 現行造字與轉碼做法遭遇問題 ..... 26
3.5 提出改善的架構與流程 ..... 28
4. 研究實證 ..... 29
4.1 需求分析 ..... 29
4.2 需求設計 ..... 30
4.3 實驗步驟 ..... 31
4.4 實證測試結果樣本 ..... 32
5. 研究發現 ..... 59
5.1 本研究計畫發現 ..... 59
5.2 研究發現成效 ..... 62
6. 結論與建議 ..... 65
6.1 研究成果 ..... 65
6.2 研究結論 ..... 67
6.3 研究限制 ..... 70
附錄 ..... 71
附錄A 細明體對照Unicode字碼表(一) 1-9000 ..... 71

附錄B 細明體對照Unicode字碼表(二) 9001-18000 ..... 71

附錄C 新細明體對照BIG-5 與Unicode字碼表(一) 1~9000 字 ..... 71

附錄D 新細明體對照BIG-5 與Unicode字碼表(二) 9,001-18,000 字 ..... 71

附錄E 自造字區細明體對照Unicode字碼表(6,500 特殊字) ..... 71

附錄F 本計劃程式碼以及API引用光碟 ..... 71

參考書目 ..... 72

# CHAPTER

# 1. 緒論

## 1.1 研究背景

醫療資訊注重病人資料隱私與資料正確性，而長期以來使用圖形字體國家(包含台灣、中國大陸、新加坡、日本、韓國...)等，於系統上均採用編碼方式輸入字型；在姓名與地址等相關資料中，每個圖形字因為各有其特殊之形、音、義，而代表意義也有所不同。因此，對醫院來說，有提供正確資料的義務；以求診病人來說，不希望自己姓名或是地址顯示錯誤。但以我國使用繁體字而言，其特殊字型相當多，中文字型編碼原則，一直以來受限於 BIG-5 字碼以及可以容許自行造字之數量，也因此常常面臨現行字型不足，或是有所缺漏而需要造字的現象。

以本研究之例子來說，醫療機構之資訊系統大型主機使用字碼，可用造字空間為 3,482 字，因此雖然可容納這麼多特殊造字字體，但是隨著業務量的成長，國人使用特殊字體偏多，統計 100 年 6 月止，本研究探索發現現行系統中使用之自行造字已達 3,372 字，且 100 年 7 月份開辦電子病歷互通平台系統後，需使用之政府戶役政系統統計所使用之特殊造字更甚達 18,216 字；且以新北市來說因市幅廣大，就像民族之大熔爐，且街道巷弄均有其特殊命名意義，因此這些字體均需要能更正確表達。當導入電子病歷後，民眾至醫院看診，而需要交換其他醫院的病歷時，在基本資料的部分，就會出現缺字，而這些缺字部分，可能因為交換前各家醫院之造字區間不同，且因此如何容納這些特殊字是當務

之急，其也為本研究計畫規劃特殊造字因應方法之主要目的。

## 1.2 目的說明

本計畫的目的主要在解決各醫院在電子病歷交換前因為各家醫院醫療系統主機面臨中文特殊字時，均採用自行控管之造字區間順序來自行造出所面臨的特殊中文字型；當透過電子病歷交換上線，各家醫院上傳至衛生署 EEC 平台集中後，而發生特殊中文字型編碼區間不統一，而導致的同一字碼確有多個中文字，或頁面上顯現中文之缺字現象。

本計畫設計理念，主要來自於網際網路資源 URI 之概念，首先，先將醫院內醫療系統主機會用到的 BIG-5 碼，和每個中文字型字面進行檢索並對映，並製作完成 BIG-5 碼與細明體對照表，同步調查現行在 BIG-5 碼造字區間現存的特殊中文字型，並將之歸類，並按照字碼排序。

其次，透過探索近年所公告的 Unicode 碼與的中文字型字面進行收集檢索，並使用人工方式把 Unicode 和 BIG-5 再進行對映處理，並完成新細明體之 Unicode 與 BIG-5 對照表，本表主要是作為匯入資料庫作為圖形字型檢索的驗證。

再來，針對 Unicode 與 BIG-5 碼中已經對照的內容之後屬於延展字集區中的特殊中文字型，進行比對與整理排序，並完成自造字區細明體對照 Unicode 字碼表(6,500 特殊字)，本研究將之視為常用之特殊字碼，主要目的為當後續有其他醫療院所可能因為經費，尚未建立字型資料庫時，亦可當為輕量型的解決方案對照表，而能夠快速的導入本計畫成果。

最後，把每個造字均當作為網際網路 URI 資源，有需要時可以採用醫院自

建的圖形資料表或是透過本計畫所提供的中推會中文字型資料庫引用 API，可以快速的展現正確的中文字型。未來在醫療機構內部規劃將字碼對照表匯資料庫表單中，並增加 URI 之欄位與各系統交換之對照表；期望引用電子病歷互通平台各項資料之需求發生時，如遇見特殊字體需要展現時，則由應用系統呼叫特殊之 API，並連至網際網路，透過 URI 取用中文推行委員會所建立之特殊造字伺服器上對應之字型，如此一來，可以解決造字區不足與各醫療機構中文字碼交換問題。

## 1.3 本計畫章節說明

本計畫共計 6 章，第 1 章緒論主要描述本計畫之研究背景與目的，同時說明步驟與程序；第 2 章針對本計畫所提出的方法與理論進行現行國內文獻與國外文獻的探索，包含了中文字造問題之背景與中文資訊交換的方法、以及現行文獻中所提出的解決方案，並描述本計畫與醫療保健之相關性，同時探討了電子病歷的法令與政策。

第 3 章在分析問題的狀況以及發展需求，同時將現行中文特殊字型造字的方法以及內部字碼轉換的方式進行分析；並提出改善中文字型交換的初步架構。第 4 章主要在進行本計畫所提出方案的實證，並透過有效的需求分析與需求設計步驟來設計本計畫實驗的步驟。第 5 章將本計畫發現進行獨立章節的探討，此為本計畫最大貢獻之處，包含了重要發現要項與可見成效的說明；第 6 章主要是針對研究進行結論與建議，包含了研究成果的要點說明，以及研究所得結論；並詳細說明本計畫的研究限制，以作為後續研究者參考。

# 2. 文獻探討

## 2.1 中文字型交換之背景

在台灣現有資訊應用系統的中文內碼使用上有 BIG-5 碼、CNS 國標碼、財稅碼、Unicode 碼等，以及仍在使用 CCCII 及 CNS 11643；其中 CCCII 由於主要使用者僅為圖書館界，使用層面不廣，圖書館界 [1]擬改用 Unicode，但因 CCCII 收錄的字數高達 75,684 [2]，暫時還無法停止使用。而 Windows 作業系統上線後，BIG-5 碼曾經一度作為標準；但因字碼區間太小，且加上 Windows 7 等新一代的作業系統上線後，可用字型雖高達 18,000 字以上，但是醫療主機上所使用的字型仍就停留在 BIG-5，也導致了雖然工作站特殊字型已加入，但是傳送到主機上卻無法對映，導致當他院需要病歷交換時發生缺碼的現象。

因 Unicode 字碼不斷的擴充與 CNS 11643 已為國家標準，將會持續擴編[1]，其餘早期字碼將陸續隨著資訊系統汰換，而陸續下市。且因中文 Windows 只支援 BIG-5 碼及 Unicode 2.0，限制了能使用的字數及內碼，因此政府機構新的資訊系統建置案，均使用 Unicode 已成為潮流。而戶役政系統之中文字型(4-byte EUC 碼)，因儲存了大約有 7 萬 3 千多個戶政用字 [3]，因此，常常在各公務機關，資訊媒體交換上造成困擾。

缺字問題是因電腦內碼對應文字的不完備造成無法順利瀏覽及輸入未收錄的文字，無法順利傳播及查詢內含該文字的文件。在本質上分成兩類，一是同碼不同字，另一個是同字不同碼。因而衍生出六類應用上的問題：瀏覽、輸入、

傳播、查詢、描述及造字問題等 [4]。

在現行內碼系統不協調的情況下，交換碼應運而生。交換碼主要是用來處理某領域下的內碼不協調問題。主要是將各內碼中的文字對應到交換碼中，然後再進行內碼系統的轉換。CNS11643 就是一個漢字領域的交換碼。不管是內碼或是交換碼，都有找不到文字(未收編)的時候，所以在內碼設計時皆會提供幾塊區域作為動態指定未收編字碼的區段。例如在 BIG-5 碼編碼中，“塜”並不存在，此時機關或個人使用者可以指定擴充字區的某一缺字碼給“塜”使用，例如 BIG-5：FA40。而有些機關及企業已結合 CNS11643 來規定內部缺字與內碼對應，來解決內部的造字工作。

## 2.2 中文資訊交換方法

在 2005 年政府服務平台中文資訊交換方法中，曾經提出了共通平台中文方面目前提供完整的Unicode 4.0 版全字庫TrueType字型(包括目前已收錄在Unicode 4.0 版的 7 萬多字CJK漢字及Symbol)，朝向規劃提供動態下載TrueType字型的功能，使應用系統在遇到尚未收錄在Unicode字集內的CNS11643 中文字時，可以馬上由Font Server上取得該中文字之向量字型，以便用於顯示及列印(Pai -Yu Sun, 2005)。

可惜的是醫療資訊系統在台灣發展歷史已經長達 30 年以上，許多大型醫學中心、區域醫療院所、地區醫院等(診所除外)，早期均已建置醫療資訊系統於大型主機上，當下使用的中文字碼大多採用 JIS 碼或是 BIG-5 碼等，與上述政府服務平台所提供之中文資訊交換方法相差很多，無論是針對中文字型的取得、

造字程序和作法上或是中文字庫的字型數量都不一致；也因此在新興的電子病歷平台(EEC，Electric Medical Recode Exchange Center)交換或所得的電子病歷自然有許多中文字型缺漏或是顯示不一致。

首先說明現行政府服務平台中文資訊交換方法的作法如下(Pai -Yu Sun, 2005)：

(1) 規定所使用的中文字必須以 CNS 11643 標準所收納的中文字為限，如果要使用未經收納的字，當事人必須先經申請，由主管機關暫予編碼，俟該中文字經審核收納入 CNS 11643 標準後，再行登記。

(2) 要求各政府機關將其資訊系統使用之自造字，整理後建立與 CNS 11643 對照表。

由以上的方法中，可以發現與現實中文特殊字型使用上與民眾期待落差很大，其中包含了：

(1) 中文字型的形、音、義各有其不同的代表意義，政府希望統一在 CNS11643 的標準字下，但國內民眾取名字卻是強調特殊、能夠表達意義，同時因為母語的關係，更是有取音、取義、取形等方法；要用強制法規規定人民用字，似乎不可行。

(2) 當發現有中文特殊字型造字需求時，由當事人先申請，主管機關暫時編碼，待字碼收錄後，再正式登記的作法曠時費日，且若每個機關都申請，歸戶原則也不清楚，則難免因等待而產生民眾抱怨；且主管機關常常於正式編碼後，卻未必通知承辦機關，也導致錯誤長期存在的事實。

## 2.3 電子病歷交換現況

病歷是指醫事人員於執行業務或從事醫療活動時所記載製作的各種記錄文件之總和 [5]，而電子病歷是以數位的方式記錄病患病歷中的內容，用以取代原有的紙本病歷記錄的方式(Tri-Service General Hospital, 2010) [6]。電子病歷的資料設計主要是依據臨床醫師的工作流程和所有涉及電子病歷作業參與者的業務需求而定[7]。電子病歷交換為提供以病人為中心的電子病歷，包含醫療過程中的各種健康資料，其資料來源可為本院內或不同院區的醫院資訊系統或電子病歷系統，來協助醫療專業人員之臨床診斷治療等相關活動 [8]。電子病歷發展歷程可分作五大階段：自動化病歷、電腦化病歷、電子化病歷、電子病患記錄與電子健康記錄，其中電子化病歷發展至電子病患記錄與電子健康記錄的重要特點為電子病歷可整合病患於不同醫療院所與相關單位所留下的病歷(Hsin-Ginn Hwang et al., 2009) [8]。

而電子病歷之生命週期包含(Hamdan O. Alanazi et al., 2010) [9]： (1)看診患者描述病情。(2)醫師需遵守的道德。(3)依法執行避免濫用權力行為。(4)授權獲得醫療記錄分析病情並加入病歷。(5)病人有進入自己的醫療記錄權力。(6)法律可以保護病人的權利。(7)法律規範醫療記錄與批價內容。(8)將分析及準備適用的法律。(9)分析存取的安全性要求。(10)透過資訊科技實現醫療記錄安全性需求。(11)透過資訊科技保存醫療記錄並解決問題，如圖 1。

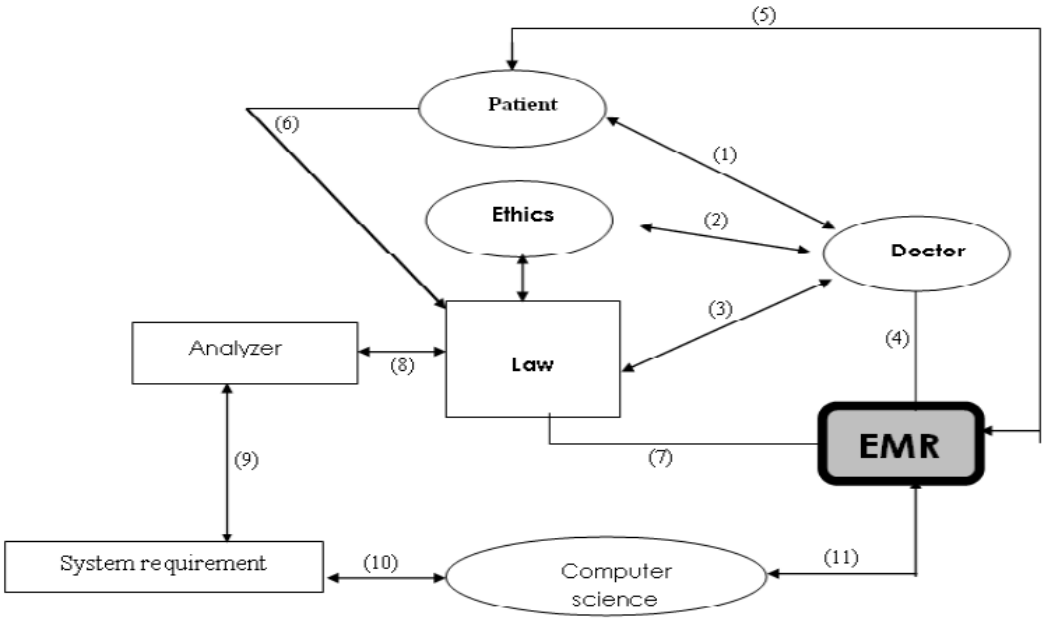



**圖 1** The structure of BPN model

為推動互通性電子病歷之重要基礎，醫學資訊能否交流與分享主要取決於語法與語義兩個方面；語法是指通訊的結構、拼寫和文法真有相同的規則，語義，用於傳達通訊的意義，指的是字典或辭典真的有一致性[10]。要達到成功的訊息交換需要包括：(1)使用可提供整合性訊息的資訊系統；(2)使用被廣泛接受的交換標準：可用於組織間整合資訊的交換；(3)區域組織網路：醫療專家和其它醫療機構可以透過區域組織網路而安全的交換醫療與非醫療的資訊[11]。

有關電子病歷的存取操作性，定義為每個診間看診的病歷，都透過 EMR 系統與 EMR 資料庫，來提供保險公司或是跨院區合作存取交換病歷。且過程涉及了四大實體議題，包含 EMR 系統、安全控制、操作 GUI 介面、需求元素等[9]，如圖 2。

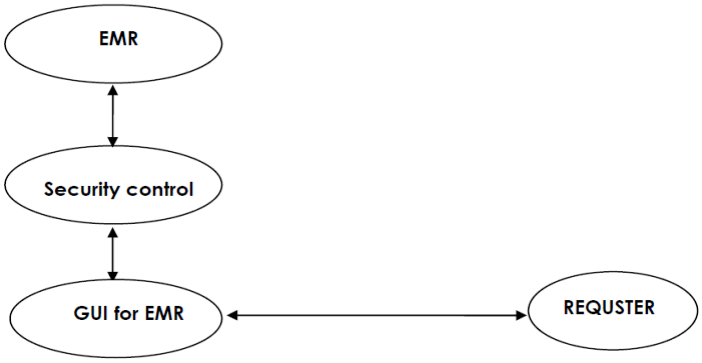



**圖 2** Operation to access the EMR

有關國內電子病歷交換的系統架構，參與交換之醫院應建置一部病歷交換主機，且應具備對交換訊息標準有編碼及解碼的功能，能夠對欲傳送之訊息加密及接收到的訊息解密；能與其他醫院的病歷交換主機及服務中心的索引伺服器互相溝通，藉以傳送及接收訊息；能依查詢的條件，由醫院資訊系統取出所需的病歷資料，適合由衛生主管機關統一建置一套共同的模組再交由各醫院自行建立擷取院內病歷的部分[12]。

現行國內醫療資料交換標準可以分成三大類，(1)訊息類，如 HL7-醫療文字、DICOM-醫療影像、X12N-財務、與 HIPAA 交易等、NCPDP-處方、與藥劑等、與 IEEE-臨床儀器、與資訊匯流排等；(2)術語類，LOINC-檢驗等、SNOMED-病理等、UMLS-醫療、NDF-RT-藥物、RxNorm-藥物、ICD-10-CM 診斷、與 CPT-帳務；(3)法規類，HIPAA-安全、與隱私權等。而在訊息的交換中，病患資料的交換有姓名和地址交換的問題，這也是本研究的需要突破的問題。

針對電子病歷重要應用理論文獻，整理如表 1；並將有關電子病歷對於資訊安全與演算法運用技術整理如表 2。而衛生署所公告之交換架構如圖 3。

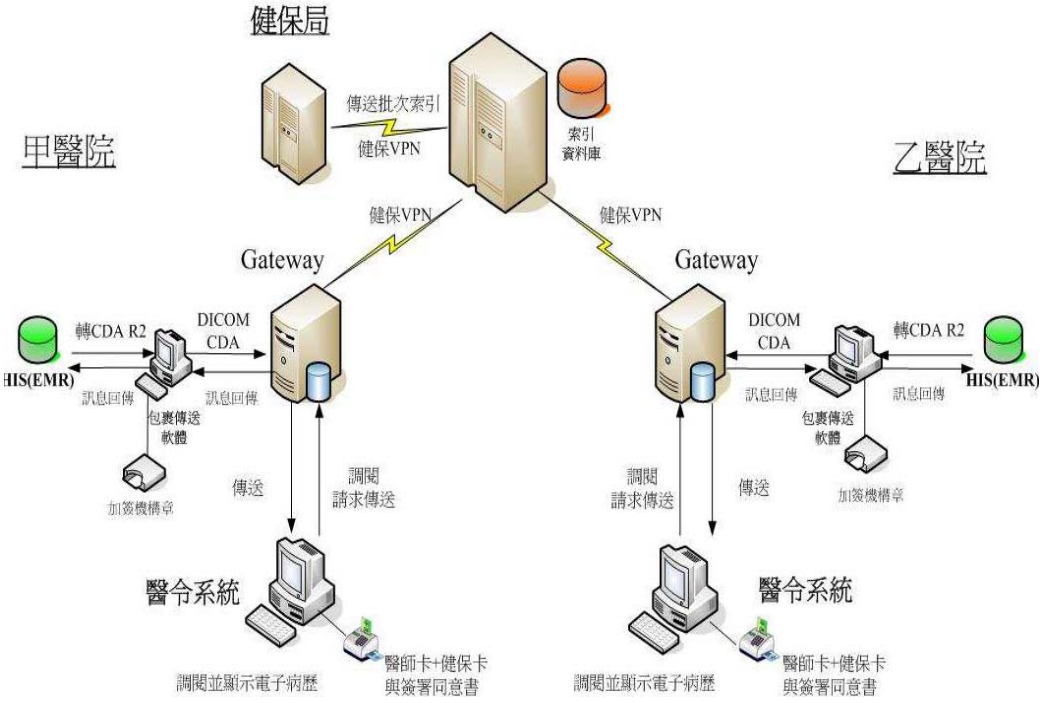



**圖 3** 衛生署公告之電子病歷現行交換介面

**表 1** 電子病歷之重要應用

| 學者 | 應用內容 | 範圍 |
|---|---|---|
| 蔡昆原等 (2009) [5] | 醫事人員於執行業務或從事醫療活動時所記載製作的各種記錄文件之總和。 | 紙本病歷 |
| 三軍總醫院 (2010) [6] | 以數位的方式記錄病患病歷中的內容，用以取代原有的紙本病歷記錄的方式。 | 電子病歷 |
| 行政院衛生署 (2012) [14] | 推動全國醫院實施醫療影像及報告、血液檢驗報告、門診用藥記錄及出院病摘之電子病歷。 | 電子病歷四大宣告單張 |
| 徐娣娥等 (2010) [15] | 電子病歷分散儲存於各醫院，交換中心提供索引集中查詢，醫院須經雙重身份認證。 | 雙重認證 |
| | 依據醫療法訂定及公告「醫療機構電子病歷製作及管理辦法」，並經二次修正，該辦法共計8條，規範電子病歷系統之管理措施等事項。 | 電子病歷製作及管理辦法 |
| 黃興進等 (2009) [8] | 五大階段：自動化病歷、電腦化病歷、電子化病歷、電子病患記錄與電子健康記錄。 | 電子病歷發展歷程 |
| | 包含醫療過程中的各種健康資料，其資料來源可為本院內或不同院區的醫院資訊系統或電子病歷系統，來協助醫療專業人員之臨床診斷治療等相關活動。 | 電子病歷交換 |
| | 成本、電腦能力、隱私與私密性、系統相容性、系統複雜性與電子病歷內容等。 | 醫師接受電子病歷的重要因素 |

| 學者 | 應用內容 | 範圍 |
| --- | --- | --- |
| | 利用web-based 技術減少各院間整合電子病歷的困難；連結各個醫療院所的異質系統，可使用HL7與臨床文件架構作為跨院電子病歷交換資訊標準。 | 交換標準 |
| In Chang (2010) [16] | (1) 病患資料可以直接傳輸至轉診之醫院，方便民眾就醫也減少醫療資源的浪費。<br>(2) 符合規定的醫療人員皆可立即取用，對於立即獲得病患及時的病況有很大的幫助。<br>(3) 電子病歷則不但清楚，也易於了解。 | 交換標準 |
| 溫信財等 (2009) [10] | 醫學資訊能否交流與分享主要取決於語法與語義兩方面。 | 交換標準 |
| 郭光明等 (2004) [11] | (1) 使用可提供整合性訊息的資訊系統。<br>(2) 使用被廣泛接受的交換標準：可用於組織間整合資訊的交換。<br>(3) 區域組織網路-醫療專家和其它醫療機構可以透過區域組織網路而安全的交換醫療與非醫療的資訊。 | 交換標準 |
| 鄭伯壎等 (2004) [13] | (1) 訊息類，如HL7-醫療文字、DICOM-醫療影像、X12N-財務等<br>(2) 術語類，LOINC-檢驗等、SNOMED-病理等、UMLS-醫療<br>(3) 法規類，HIPAA-安全、與隱私權等 | 交換標準 |
| 王亮雯等 (2009) [17] | 醫院與醫院之間的檢驗資訊需要交換時，各醫院的資訊系統無法瞭解其他醫院資訊系統的檢驗資料內容，增加了大量的處理時間及成本。 | 交換標準 |
| 陳偉等 (2010) [18] | 網際網路中，網路傳輸常發生錯誤，造成畫面破損；傳輸頻寬變動，造成多媒體資料無法傳送或頻寬浪費的情況。<br>預先備妥寬頻速率的資料，再做動態的調整；利用新的串流媒體技術採用可調適壓縮的編碼，克服寬頻變動的問題。 | 網路品質 |
| 黃衍文等 (2004) [12] | 在病歷交換中心建構索引伺服器儲存每個病患在各個醫院的病歷的位址，每次病患就診時，醫院負責傳送一個記錄到索引伺服器的資料庫。 | 交換標準 |

**表 2** 電子病歷運用資訊科技之優缺點比較

| 學者 | 優點 | 缺點 |
|---|---|---|
| Gobi and Vivekanandan, (2009) [19] | 數位信封+MD5 演算法 AES+HECC | 實施健全但過於昂貴但缺少認證機構（CA）來確保簽章是值得賴 |
| Ferreira et al., (2004) [20] | 電子病歷有保存期限<br>有效病歷的認定方式 | 電子病歷使用<br>RSA 安全性有慮 |
| Bos et al., (2004) [21] | 非對稱的公鑰基礎加密演算法(如：RSA，頁 435) | 未提及電子簽章方式 |
| Brandner, et al., (2002) [22] | 簽章在醫院公共金鑰設施<br>電子簽章與電子病歷結合<br>PKI 認證的標準化介接<br>PKI 國家或法律所規範建構 | 缺乏道德與電子病歷隱私的描述 |
| McGuire and Fisher, (2008) [23] | 透過遺傳/基因演算法提供分析、不可否認性，獨特性，並保護資料 | 需要更長遠思考<br>無明確提到電子病歷的保護方式 |
| Janbandhu and Siyal, (2001) [24] | 生物識別簽章與 PKI 整合(該研究透過 JAVA 實作虹膜識別測量+RSA+數位簽章) | 金額過於昂貴<br>沒有實際應用範例 |
| Anderson, (2000) [25] | 健康資訊政策問題<br>隱私威脅和保密來自提供病人護理服務的機構 | 分散式電子病歷是記錄好辦法<br>不明確安全要求<br>解決方案不存在 |
| O'Brien and Yasnoff, (1999) [26] | 國家衛生系統評估就業和個人資料<br>機密性<br>保密性<br>安全性<br>公平性的實踐議題 | 涵蓋加密演算化和數位隱藏<br>包含機密性、完整性、驗證<br>沒有涵蓋重要的因素不可抵賴性<br>PKI 使用不明確 |
| De Meyer and Lundgren, (1998) [27] | 記錄應該分散保存<br>安全解決方案不明確 | 好工作方式，未提及使用的演算法未提及實現安全要求的方式 |
| Epstein et al., (1998) [28] | 立法強制醫療記錄安全<br>RSA 演算法與數位簽章 | 道德議題未提及 |
| Rind et al., (1997) [29] | 透過電子標示來識別患者並跟踪傳輸的信息<br>當電子病歷記錄刪除，只是進行標記，而非真正刪除 | 未改善院際傳輸流程<br>如何保護病患資料?<br>傳輸過程是否加密?<br>外網內網傳輸病歷不同保全方式 |

## 2.4 電子病歷交換內容與步驟

現行台灣實施電子病歷交換最有利的條件為各醫院均採取衛生署公告 4 張單張之標準 XML 架構作為交換標準；大部分的作法為在醫療資訊系統中，將實施電子病歷之範圍(可能為血液檢驗類、出院病摘類、醫療影像類、門診用藥類的某些類別)的醫令資料，在完成病歷製作時，即要求醫師簽章，並產生 XML 格式檔案，暫存在院內的檔案伺服器；並透過檢查機制確認檔案格式無誤後，再加上醫療機構卡簽章完成，最後批次透過閘道器上傳至衛生署 EEC 平台。

當病人至醫療院所求診時，有需要進行外部病歷調閱時，醫師則透過網際網路來登入 EEC 平台上之電子病歷交換系統，並驗證完醫事人員卡後，在病人同意下，插入病人健保卡驗證，醫師便可在平台所顯現的某病人所有就醫資料中，挑選所需要的類別項目，調閱目前存於 EEC 平台上所有的電子病歷索引，再由各家醫院上傳提供其他細部病歷或是影像。或是，採用院內所設置的 EEC 交換平台並透過掛號資訊，先行調閱存檔，提供後續診斷病人病情上使用。

技術上來說產生電子病歷交換所需 XML 檔案程序如圖 4。簡要說明如下：醫師於醫令完成病歷製作後，需以醫事人員憑證進行簽章，再由醫療資訊系統中產生 XML 檔案，並傳送至電子病歷管理系統進行 XML 格式檢查（包含 CDA_R2 格式、LONIC 格式）；若是醫療影像類，則需要透過演算法取得 Hash 摘要，並加上醫事機構卡簽章後，將各類電子病歷互通資料傳送至 EEC 平台，以供其他醫院醫師調閱使用。

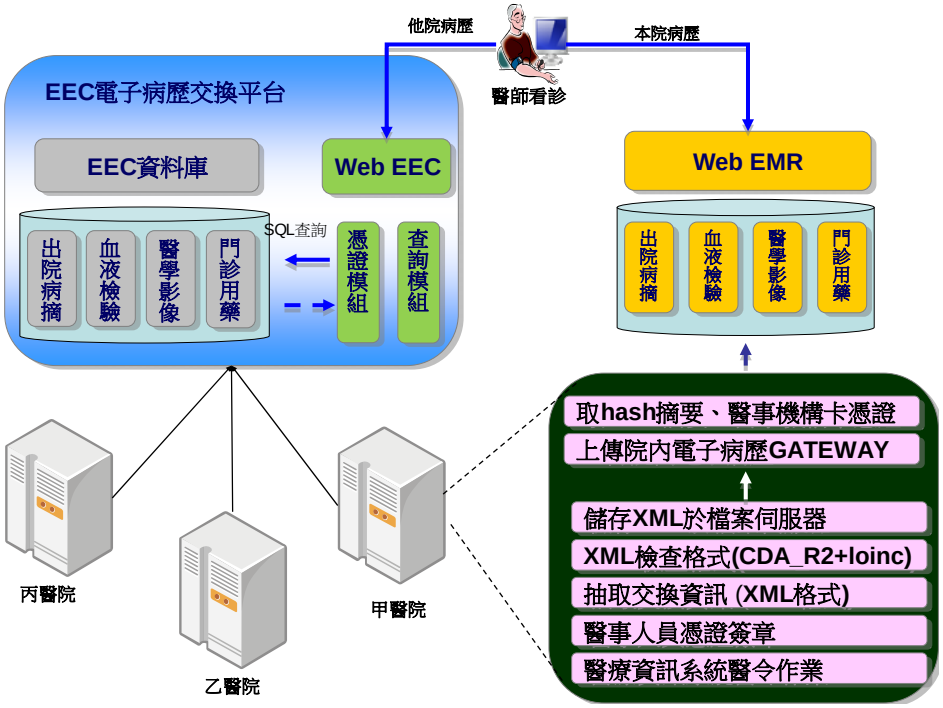



**圖 4** 產生電子病歷交換所需 XML 檔案圖

(資料來源：本研究整理)

為了有效了解台灣地區實施電子病歷後，各醫院醫師看診業務流程發生一些變化，首先說明現行各醫師調閱電子病歷相關業務之作業步驟如圖 5[14]。

(1) 步驟 1. 醫師登入系統，經過帳號密碼檢查後，以病人健保卡以及醫事人員卡查詢該病人之可交換電子病歷。

(2) 步驟 2. 如病人有可交換電子病歷記錄，醫師於 EEC 平台上輸入可調閱之醫院、病歷單張、以及查詢區間，並列印同意書後，請病人簽署。

(3) 步驟 3. 病患同意後，醫師在病人可調閱的清單中，一一尋找可能適合的病歷，並加以勾選後，按下「調閱」來瀏覽。

(4) 步驟 4. 醫師在病人可能有用的病歷清單中，若找到可用病歷，則按「下載」，則會載入於該醫院之閘道器，並主動通知病歷室列印病歷。

(5) 步驟 5. 病歷室人員接獲通知後，將下載的病歷加上醫事機構簽章，並下載至指定電腦，提供給醫師製作病歷，或是進行保存與處理。

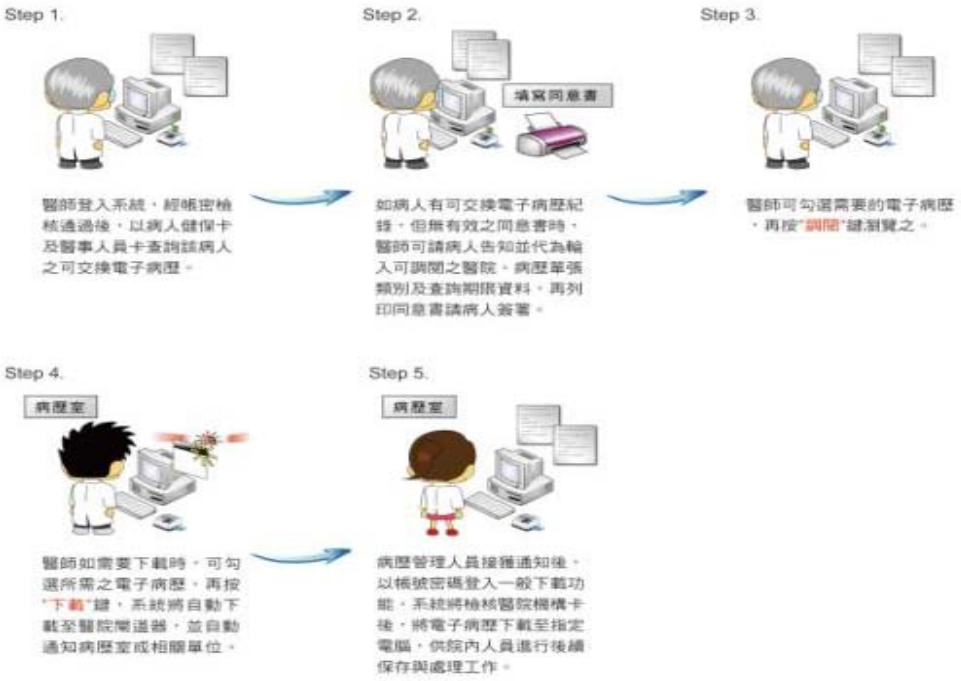



**圖 5** 現行查詢電子病歷流程

(資料來源：Department of Health, Executive Yuan, 2012)

由本節中可得知電子病歷推行的政策為政府近年來所重視，且各家醫院均把符合交換格式的 XML 檔案集中到政府 EEC 平台，目前因部分還處於專案建置期，但是已提供相關交換查詢頁面來協助醫院進行健康資訊分析，以及提供需要院外調閱電子病歷的醫師一個整合的平台與介面，可以因為尚未進行中文字型整合，導致特殊中文字型會有缺漏的問題，也因此本研究的價值在提供一個解決電子病歷交換後，對於病人個資之正確性方法。

## 2.5 電子病歷中文字型轉換問題分析

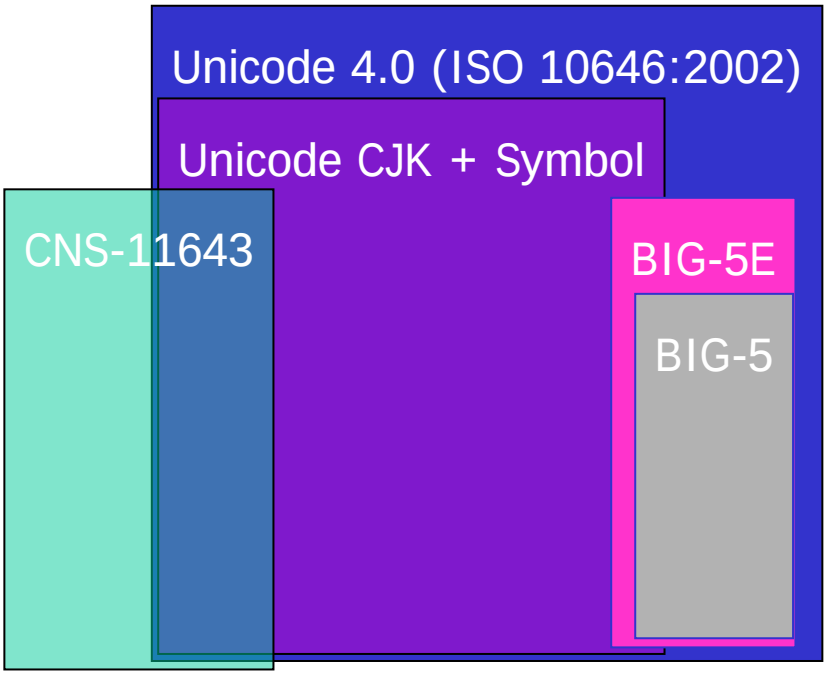



**圖 6** 各種字集的涵蓋範圍

(資料來源：Pai -Yu Sun, 2005)

在缺字問題的現行解決方法裡，如表 3，計畫中分析 5 種解決方法之優缺[4]如下：

**表 3** 現行解決方法優缺比較

| 解決方法 | 優點 | 缺點 |
|---|---|---|
| 1. 全字內碼法 | 減少缺字發生的頻率。 | 須經過蒐集.討論及長時間的審查過程。<br>缺字問題仍與時俱增。 |
| 2. 擴充字集法 | 完全結合字集及字型概念，在作業系統中得到完全的支援，可被眾多應用程式使用。 | 受限於缺字內碼數有限的影響，並無法將所有的字放入系統中。 |
| 3. 字型資源法 | 每項資源都有唯一的網路地址可供識別及指示資源所在。 | 因未將字型坎入系統中，致其他應用程式無法使用。 |
| 4. 部件組字法 | 不需立即造字，即可馬上描述缺字長相，可描述無限多個缺字。 | 在許多內部系統應用程式中無法順利瀏覽。 |
| 5. 統一造字法 | 可管理缺字，易於專家造字，使得造字可以完整。 | 系統的建置時程不一，有些比統一造字法還提 |

| 解決方法 | 優點 | 缺點 |
| --- | --- | --- |
| | | 早推出，整合不易。<br>統一造字之時程過長，時效性常常為人所詬病。 |

因各個編碼法內所含文字也各有差異，所以造成編碼系統的不協調。例如：“發”的編碼是 BIG-5：B56F 和 Unicode：767C；”塋”的編碼是 Unicode：5803 但在 BIG-5 中則沒收錄。也因此中文字資訊化常常遇到的問題非常多，但經過整理後大致上歸類 5 種主要原因 [30]：

(1) 缺字困擾普遍存在。現今任何一個系統都不能表達全部漢字，都有缺字問題。

(2) 排序混亂。現有的各漢字編碼方案均沒有認真處理好排序問題。

(3) 多種中文字碼並存，造成傳輸、交流的障礙。

(4) 輸入法的規範、優選仍無成效。高效率、易學用、與漢字基礎教育互相協調的輸入法久喚不出。

(5) 字量龐大使字庫的設計、生成、存貯、調度都有困難。

而為了擬定一個解決現行發現之各種中文碼轉換問題，經過文獻探索與和中文推行委員會訪談後，發現醫療院所進行電子病歷交換後，大略會發生的中文字型問題原因，歸納整理如下說明：

(1) 去回轉碼錯誤(Round-Trip Conversion Error)：將電子病歷 XML 檔案上傳後，存在 EEC 平台；當其他院所有必要時，則透過 EEC 下載在有需求

醫院之 Gateway 伺服器，再透過電子病歷系統讀取。但是這樣的轉碼過程就會發生原醫院有的中文特殊造字字碼，使用醫院並沒有同樣的該段中文特殊字碼區間，因而發生去回轉碼問題。

(2) 字碼由來源字集(Source Char-Set)轉碼至目標字集(Target Char-Set)後無法再轉回原來源字集：當從 EEC 下載後的電子病歷，如果已經展現於本地電子病歷管理系統後，且經過醫師閱讀存檔後，則會轉回本地之對映字碼，若要在回溯回原字型，則會有對映原字型上的困難。

(3) 因為轉碼過程中遺失掉某些資訊，而使字碼無法由目標字集再轉回原來源字集：此問題常常發生於去回轉碼錯誤後，因目標字集發生了使用並不存在的區間，導致資料已經遺失，當要使用時就發生了缺字的議題。

(4) 不當置換字元問題：通常是因為目標字集缺字，使的轉碼後發生來源字集的某個字碼被對應到目標字集的「置換字元」(Replacement character)：此問題為張冠李戴的嚴重問題，也就是來源字集和目標字集均有該段特殊造字區碼，但是同一碼卻造不同中文特殊字型，結果並未察覺；當印出收據、或是健康資料給民眾時，產生錯誤而拒絕繳費或是法律效力的重大問題。

(5) 忽略資訊問題：當醫院採取轉碼程式針對外部來的資訊進行轉換時，有些轉碼程式當遇到目標字集缺字時，使直接把缺字的字碼忽略掉(刪掉)，這樣就會在展現的頁面上出現缺字或是以而以◇、□等代替(如：王書◇、◇德友等)。

## 2.6 本計畫與醫療保健之相關性

透過本研究計畫之執行，可以讓電子病歷互通平台運用之相關範圍更加正確，同時可以解決歷年來當病患至醫療院所臨櫃辦理各項醫療表單或是證明時，常常發生因為姓名或是地址中文字型無法顯現，而發生醫療糾紛的情勢。透過本研究計畫，也可以提供衛生署相關醫療院所對於特殊字型、難字或是罕見字的呈現議題，讓民眾得到更多、更正確的服務，所開立的證明有更佳的公證性。

除此之外，本研究計畫所提作法可以將電子病歷交換之內容，透過交換，讓需要交換的醫療院所，可以得到更正確的交換資訊。也可以擴大電子病歷互通交換平台的效益。另，綜觀國內資訊系統趨勢，從人一出生即開始建立的戶役政字型將是國內各機構中文字體媒體交換的標準來源；也因此本計畫所提的方法，也將可以提供國內各醫療機關有資料交換需求的單位一個有效的方法，也讓醫療資訊系統的應用更加廣泛。.

## 2.7 相關政策或法令

電子病歷之發展為近年來重大國家醫療政策，行政院衛生署所屬醫療機關與各直轄市衛生局各醫療單位，均主動積極的申請加入電子病歷互通平台之資料交換與陸續通過檢查。病人就醫時，看診醫師可以透過病人的『健保 IC 卡』、『同意書』及『看診醫師的醫事人員卡』等，至電子病歷交換平台取得病人在其他已實施此作業之醫院的檢查資料。

本計畫之研究對象醫療機構，以推動「全人健康照護」為目標 [31]，透過

電子病歷互通的實施並保障個人健康資訊隱私條件下，提昇醫療資源運用效能、服務品質及病人安全，達成全民健康資訊ｅ化流。

另基於個人資料保護法 [32]，行政院衛生署亦給了電子病歷法規面及安全面嚴謹的規範 [33]，於是實施電子病歷的醫療院所需建構更周延的資安技術來防止非原記錄者修改及刪改資料，其管控比紙張更為嚴謹，更重要是每個曾經調閱或查詢病人病歷的人都會被記錄，對病人的健康照護記錄隱私保護更為完備。鑑於此，電子病歷交換將越來越頻繁，將是未來民眾就醫的趨勢；同時透過電子病歷的推行，將可以更有效的運用醫學資源。

# 3. 研究方法

## 3.1 問題狀況與發展需求

本研究旨在規劃研究計畫對象醫療機構(以下簡稱對象醫療機構)，因應電子病歷互通交換平台系統上線後，因各家醫院針對難字、罕見字、自造字所使用之造字內碼在民眾姓名與地址呈現時，且因各醫療單位自造字編碼均不同，且尚未統一，在使用衛生署電子病歷互通交換平台系統(使用 Unicode 3.0)之各醫療院所之 Gateway 上之衍生應用系統呈現之缺字、替代字造成法律效力上重要議題之解決方案。

## 3.2 醫療院所採用中文特殊字型造字方式

首先說明醫院內處理中文特殊字型的流程，以作為本計畫預解決問題的方法之設計依據，並將步驟簡易說明如下：

### 3.2.1 掛號櫃台發生中文特殊字型需求

當民眾到醫院看診時，會先確認是初診或是複診；若是初診病患，則需要進行基本資料的填寫，當櫃台輸入人員發現病人的姓名或是地址等資訊有無法輸入的中文特殊字型時，首先需要向病人提出身分證件確認，確認無誤後，則會使用手寫辨識法等方式查其注音輸入法之拼法，找到該字後再輸入於系統中；若找不到則先向病歷管理單位提出造字申請，並留下記錄，等待造字人員完成字型後，通知補入於病人看診之基本資料中。

### 3.2.2 造字人員確認並造字

收到造字申請單後，負責造字人員會先視其字型，並檢查全字庫字型，是

否已經收錄該字型，且是否編碼於標準字碼內，若已收錄則通知需求單位使用其對映之 BIG-5 碼來進行處理，並停止造字；若未收錄或是並未於標準字碼區，則可以視其外觀，多找相近字型，準備在下一步驟組成該特殊字型。

### 3.2.3 下載醫療資訊系統之造字區並新增造字

經過 3.2.2 後，造字人員在造中文特殊字型時，需將醫療資訊系統中所使用之造字區間下載於造字電腦端，並按照順序點選所需新增的造字碼後，組合或是新創中文特殊字型，並設定其所對映的內碼輸入法、注音輸入法、新注音輸入法....等等，並加以存檔。

### 3.2.4 將新增造字區上傳至醫療系統主機

造完字型後，需將已完成造字之中文特殊字區間重新上傳至醫療資訊系統主機，並更新造字檔；如此一來，當有新的 Client 端連入，並登入主機時，則會自動下載新字型於 Local 端，以備不時之需。

### 3.2.5 Client 端重新讀取造字區後展現中文特殊字

完成上述造字程序後，Client 端需要重新更新並下載字型，如此並可以透過通知單上的注音輸入法、新注音輸入法、內碼輸入法...等來選擇已經造字完成的新中文特殊字型。

經由以上的分析，可以知道現行的中文特殊字型造字方式，其最大目的僅在於緊急需要時，提供一個造字方法來滿足本地醫院之醫療業務所需，對於電子病歷交換或是其他資訊交換的議題，則不見相關的因應方案等重大議題。

## 3.3 醫療院所現行採用轉碼方式

除了造字的流程問題外，還有一項重要的工作是，在當外部大量資料提供與對象醫院時(如農漁民健康檢查名冊、團體健康檢查名冊...)，則需要先進行轉碼後，再行匯入院內的資訊系統，並取得相關病歷號碼。此於因為有外部的資料轉入，來源資料不管是來自於農會或是其他醫院，亦有原本屬與其他醫院因中文特殊字型之自造區域的問題，因此也需要進行處理。在本節中先分析現行各醫院中文資訊交換的流程。

醫療醫院資訊系統之目前字碼轉換作業，主要以原 HIS 醫療系統字碼(如：整合醫療資訊系統內碼與大五碼(BIG-5)轉換作為資料交換方式，字碼轉換分為兩種，通常主要在於整合醫療資訊系統內碼與 BIG-5 字碼對照表字數的多寡不同。其原理為原 HIS 系統與開放系統(如：本院後續配合之電子病歷互通平台之新系統)資料交換以及 BIG-5 碼與醫療資訊系統內碼互轉，以程式進行轉換作業後儲存於資料庫。

在本流程中，重要的是目前HIS系統中使用字碼字數，很多醫院都還是使用 BIG-5 碼來進行中文字碼的處理，其中文字碼約 8,756 字；但是周遭相關單位轉入後，可能用到的字碼數卻有 12,733 字以上，也因此缺字機率相當高。

**表 4** 目前HIS系統中使用字碼字數表

| 轉碼程式名稱 | 現行內碼與 BIG-5 對照表字數 |
|---|---|
| 醫療資訊系統內碼(HIS) | 8,756 |
| 周邊資訊系統內碼 | 12,733 |

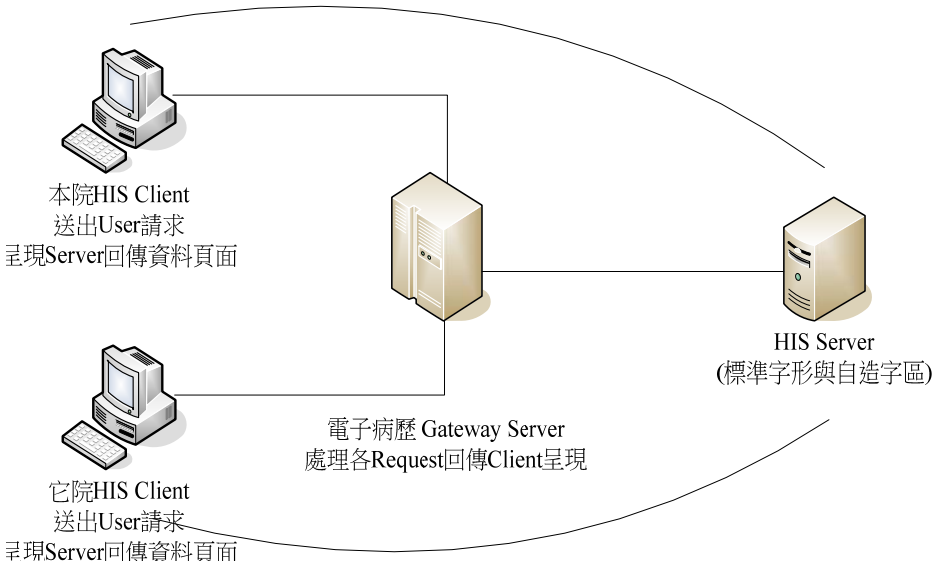



**圖 7** 現行中文字型使用示意圖

也因此現行本案之醫療機構應用系統使用中文字型的流程由字型伺服器提供現行 Unicode 3.0 標準字型與本案之醫療機構自造字，如使用者登入 AP Server 後，通過權限驗證後，如果有自造字型需求，則透過 AP Server 至原 HIS 系統造字區中取得，因此在電子病歷互通平台使用戶役政字型如未於自造字區，則抓不到字型的問題。

## 3.4 現行造字與轉碼做法遭遇問題

為了有效解決中文特殊字型資訊交換的問題，在本計畫中，由上章節所討論的現行中文特殊字型造字流程與外部檔案轉碼程序來說明遭遇的問題：

### 3.4.1 現行造字程序的問題

中文字型有其特殊之代表意義與特性，因此各醫療單位均有使用中文字型造字之字碼區間，但是卻未進行標準化與統一，因此電子病歷互通平台上線後，在中文姓名、中文地址或是部份中文病摘會出現越來越的無法交換中文字體，

而以◇、□等代替(如：王書◇、◇德友..等)，也造成交換平台上姓名或是地址資料缺漏等現象將越來越頻繁。因此當前主要課題為解決現行醫療機構各單位針對罕見字與自行造字字碼不統一，當導入全國性之電子病歷交換平台系統時，取得民眾姓名與地址時，造成諸多遺漏或是不正確問題(如圖 8)。

中文特殊字型

300 筆 資料, 顯示第 141到 160筆 .[第一頁/前一頁] 4, 5, 6, 7,

| NPIDS | 原名稱 | CNS |
|---|---|---|
| 010000057617 | 儲◇忻 | 姓名-2-d6b74/ |
| 010000057633 | 林作梅 | 門牌-1-33a28/ |
| 010000059700 | 王書◇ | 姓名-3-d2d28/ |
| 010000059741 | 林凰麟 | 姓名-2-34567/ |
| 010000058663 | 鴻瑾 | 姓名-1-f614b/ |
| 010000058697 | 陳能猷 | 門牌-1-33a28/ |
| 010000058144 | 潘文財 | 門牌-1-33a28/ |
| 010000058172 | 路祖豪 | 門牌-1-33a28/ |
| 010000060563 | 程際治 | 門牌-1-33a28/ |
| 010000060586 | 韓聘三 | 門牌-1-33a28/ |
| 010000060840 | 瞿荊洲 | 姓名-2-33e57/門牌-1-33a28/ |
| 010000060874 | 蔡樹木 | 門牌-1-33a28/ |
| 010000060881 | 胡獻昂 | 門牌-1-33a28/ |
| 010000060945 | 任潤 | 姓名-3-f2268/ |
| 010000061074 | 邱冠儒 | 門牌-1-33a28/ |
| 010000064092 | 郭紅柿 | 姓名-3-42669/ |
| 010000067444 | 林游旭理 | 門牌-1-33a28/ |
| 010000061294 | ◇德友 | 姓名-1-c6174/ |
| 010000061396 | 江覲進 | 姓名-2-32f29/ |
| 010000064250 | 姜戴英 | 門牌-1-33a28/ |

**圖 8** 現行遭遇的中文特殊字型呈現問題

### 3.4.2 現行轉碼流程的問題

在現行轉碼的程序中，也因為字碼區間不同，因而有一些現象已經發生在現行的外部資料匯入的過程來分析，包含三大重要的問題：

(1) 遺失問題：字碼由外部來源轉換至本院字碼時，因為外部字碼可能使用全字庫(約 8 萬字)或是 Unicode (12,733 字以上)多於本地字碼(8,756 字)，導致轉換完成後，本院所需中文資料之特殊字型無法顯示或是亂碼的重大問題。

(2) 錯置問題：當外部來源資料使用轉碼程式後，發生特殊造字區之字集的某個字碼被對應到目標字集的另一個字碼，此問題為張冠李戴的嚴重問題，也就是來源字集和目標字集均有該段特殊造字區碼，但是同一碼卻造不同中文特殊字型，結果並未察覺；當印出收據、或是健康資料給民眾時，產生錯誤而拒絕繳費或是法律效力的議題。

(3) 方塊問題：當醫院採取轉碼程式針對外部來的資訊進行轉換時，有些轉碼程式當遇到目標字集缺字時，使直接把缺字的字碼以某個符號或是方塊來替代，導致顯示錯誤的問題。

## 3.5 提出改善的架構與流程

通常透過 EUC 碼交換，如對照於戶役政系統上字型資料：CVLDTL 資料表，使用 EUC 碼，轉換為 Unicode，並將姓名、地址難字部分轉為對應的圖形字型碼，存放於 msg01 欄位中，可供程式透過網路存取字型伺服器字型，如圖 9。

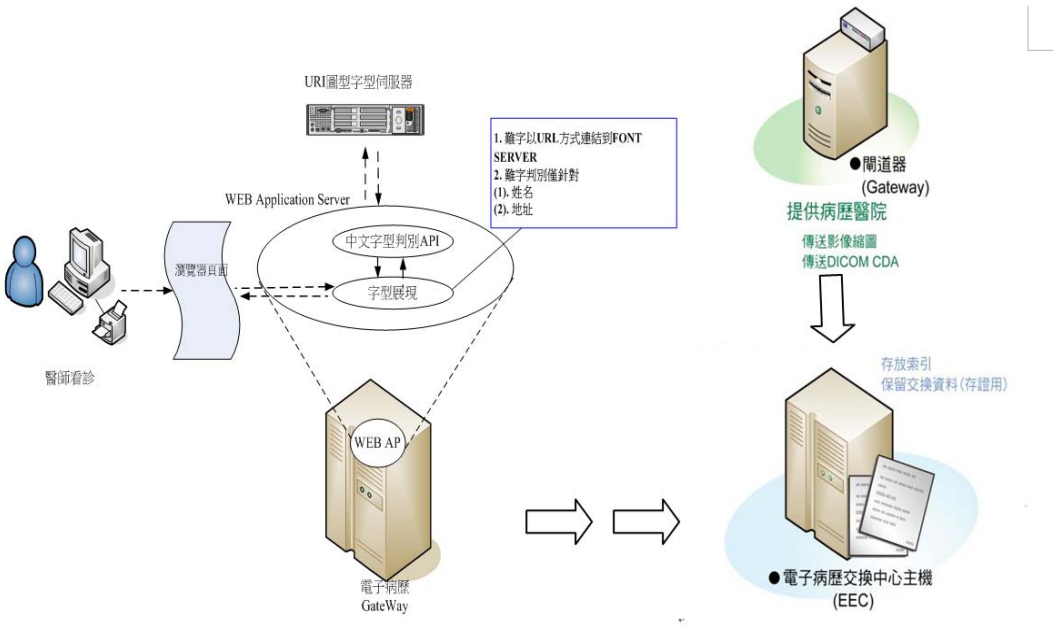



**圖 9** 本計畫取用電子病歷互通平台中特殊字型方式

# 4. 研究實證

## 4.1 需求分析

(1) 本案之醫療機構使用者頁面展現特殊字問題(含 Local 列印)

可以預見，本案之醫療機構各應用系統，在本案之醫療機構使用者登入系統後各頁面呈現，如果有戶役政造字，依現行機制，因為戶役政字碼編碼與本案之醫療機構編碼不同，將無法展現。

(2) 櫃台進入應用系統查詢資料特殊字展現問題(含 Local 列印)

因現行戶役政與本案之醫療機構造字編碼不同；且兩者均與現行 Windows 作業系統之字碼編碼不同，若保險人在家中使用電腦，其作業系統因素，將導致查詢時，會出現不正常字體。

(3) 本案之醫療機構 EIS 醫療統計報表列印問題

因 EIS 醫療統計報表強調效能與列印之正確性，印表機如遇見特殊區段編碼之中文字型，將自動至本案之醫療機構現行中文字型伺服器中取用字型後列印，而現行本案之醫療機構中文字型伺服器是採用 Unicode 3.0 編碼；且資料庫中使用 Unicode 3.0 字碼儲存，因此如果電子病歷互通平台使用戶役政字型未造於本案之醫療機構字型資料庫，將會造成特殊字無法印出問題。

(4) 資訊交換問題

因現行外部資料來源，各機關所提供的資訊，並非都按照本案之醫療機構所需格式編碼，因此可預期相關資料轉入後，會有部份字型無法正常轉入的問題，而導致字型遺失；而本案之醫療機構現行所提供的轉碼後的檔案，也無法

滿足各機關所需，因此可能將會造成其它單位使用上的問題。

## 4.2 需求設計

以上四大問題經過客觀分析，歸納出幾項問題，需要先行解決，才能處理上述的四大問題：

(1) 將 CNS 字碼轉入 Unicode 3.0 取代現行自造字型

目前戶政使用 EUC 編碼，本案之醫療機構上端舊系統使用整合醫療資訊系統內碼(JIS)編碼，「電子病歷互通平台」等新系統因使用 Unicode 3.0 編碼；而現行各種編碼中均無法完全解決中文難字與自造字的問題；但是以 CNS 所涵蓋的字型最廣也最多，且具備國家認證，建議將以 CNS 字碼為 Base 轉入本案之醫療機構現行 Unicode 3.0 編碼，以擴充現有字型庫。

(2) 現存於資料庫中本案之醫療機構自造字字型轉碼需求

在字型編碼更替之前，須將本案之醫療機構現行存於資料庫的 Unicode 3.0 自造字型須進行與 CNS 對照後進行轉碼，再存回資料庫，以確定正確字碼存於資料庫中，此項工作建議能夠由中文推行委員會(以下簡稱中推會)提供相關對照編碼表與本案之醫療機構自造字區碼進行比對，並透過相關 API 程式進行轉碼，中文推行委員會公開網址上可以取得相關 API 網址 [34]。

(3) 未來字型更新的程序與規範需求

有關 CNS 新字問題，因其為國家統一標準，可以盡量減少未來造字需求；但是如不可避免須造字時，建議先行造字型圖檔，待每月 CNS 更新後進行下載，即可展現出相關字型。字型圖檔的使用方式，為採用非 CNS 碼區字，則透過 API

進行圖形字型下載使用。同時硬體設備建議為一台字型伺服器供各系統共用，以避免同步需求，以確保圖形字檔的統一。

(4) 原 HIS 系統造字區重新討論的需求

原 HIS 系統造字區對應之 Unicode 3.0 造字區，之前規劃部份僅剩可造字範圍，上端還有大量造字需求，本計畫將重新檢討難字區配置，此部份需要原 HIS 廠商協助配合。

(5) Unicode 3.0 字碼與 CNS 字碼對照表的維護需求

現行本案之醫療機構使用的自造字區碼，與 CNS 中文碼的對照表，需要共同制定，並須提供現行所有的自造字區碼，因無論是應用程式使用、或是印表機呼叫字型都將以此為標準，因此該表的首次維護非常重要。

## 4.3 實驗步驟

為了有效執行計畫，首先將本計劃中相關實驗的程序進行繪製，以有效進行後續的計畫工作，其中包含：

步驟 1. 取得資源：取得外部資源和整理內部資源對映，對象醫院有醫療資訊系統(HIS)現行使用之 BIG-5 相關字檔；但缺乏 Unicode 常用之 18,000 字集，因此本計劃首先需先取得外部資源檔 Unicode，以整理和對象醫院之 BIG-5 碼進行對映比較。

步驟 2. 進行 Unicode 與細明體檢索對映：取得外部資源後，需要將外部字碼 Unicode 之中文字型外觀進行辨認，因此計畫中首先將 18,000 字之外部資源，進行與新細明體之整理，因檔案過大，本步驟需切為

兩個檔案進行。

步驟 3. 將 Unicode 與 BIG-5 檢索對映：本工作為本計劃最重要的工作，亦即將外部取得的 Unicode 經過與細明體整理後，必須與對象醫院醫療資訊系統使用之 BIG-5 碼進行相關對映，此對映為撰寫轉檔程式與購入中推會 API 程式之測試依據；有了此表亦可以匯入資料庫，作為輕量級的圖形字庫，若醫院經費不足時，則可以在院內自行架設圖形字型庫，以供存取 。

步驟 4. 開發轉碼程式：本步驟主要在將對映後的 Unicode 與 BIG-5 碼資料表，載入資料庫後（或是低於 65,535 字以下，亦可使用 Excel 資料表），透過 C# 程式語言來撰寫轉碼程式，此步驟可以解決外部來源資料檔轉換以及提供給外部單位參考的資料檔先進行轉檔使用，此部分採取自行開發程式比對；而在頁面所使用之中推會字型 API，因需要至網際網路上搜尋資源，則採用購入呼叫元件，來內崁於網頁上的方式。

步驟 5. 崁入中文特殊圖形字型庫呼叫 API 於測試系統：本步驟主要在提供電子病歷管理系統或是未來新興應用程式若有需要使用特殊中文字型時，透過本計劃所撰寫的標準頁面碼，即可將已對映的圖形，正確展現於客戶端。

## 4.4 實證測試結果樣本

為了提供後續參考者，很快得到所需要的特殊中文字型轉換後所得結果，

本計劃將重要的中文特殊字區間 Unicode 檢索排序，檢索結果樣本、Unicode 和大五碼(BIG-5)進行對映檢索結果樣本、Extension Unicode 和大五碼(BIG-5)進行對映結果樣本、轉碼程式撰寫樣本、標準頁面設計樣本等五項實證結果呈現於下方說明。

(1) 中文特殊字區間與 Unicode 檢索排序

本程序主要在將現行可能會用到的特殊字碼與 Unicode 進行對照，將現行 Windows XP 中可能會用到的特殊字碼與 Unicode 進行對照，在此列舉 50 筆如表 5，經過本研究檢索排序後之內容如附錄 A。

**表 5** 特殊字碼與Unicode進行對照表(樣本 50 字)

| 項目 | Unicode | 細明體 |
|---|---|---|
| 17140 | **90F3** | 郠 |
| 17141 | **90F4** | 郴 |
| 17142 | **90F5** | 郵 |
| 17143 | **90F6** | 郶 |
| 17144 | **90F7** | 鄉 |
| 17145 | **90F8** | 鄐 |
| 17146 | **90F9** | 鄖 |
| 17147 | **90FA** | 鄨 |
| 17148 | **90FB** | 鄳 |
| 17149 | **90FC** | 鄼 |
| 17150 | **90FD** | 都 |
| 17151 | **90FE** | 鄾 |
| 17152 | **90FF** | 酃 |
| 17153 | **9100** | 荎 |
| 17154 | **9101** | 酅 |
| 17155 | **9102** | 鄂 |
| 17156 | **9103** | 酇 |
| 17157 | **9104** | 酆 |
| 17158 | **9105** | 酁 |

| 項目 | Unicode | 細明體 |
|---|---|---|
| 17159 | **9106** | 鄞 |
| 17160 | **9107** | 鄠 |
| 17161 | **9108** | 鄰 |
| 17162 | **9109** | 鄉 |
| 17163 | **910A** | 鄋 |
| 17164 | **910B** | 鄚 |
| 17165 | **910C** | 鄘 |
| 17166 | **910D** | 鄗 |
| 17167 | **910E** | 鄐 |
| 17168 | **910F** | 鄜 |
| 17169 | **9110** | 鄤 |
| 17170 | **9111** | 鄢 |
| 17171 | **9112** | 鄒 |
| 17172 | **9113** | 鄔 |
| 17173 | **9114** | 鄃 |
| 17174 | **9115** | 鄑 |
| 17175 | **9116** | 鄖 |
| 17176 | **9117** | 鄗 |
| 17177 | **9118** | 鄘 |
| 17178 | **9119** | 鄙 |
| 17179 | **911A** | 鄚 |
| 17180 | **911B** | 鄴 |
| 17181 | **911C** | 鄘 |
| 17182 | **911D** | 鄝 |
| 17183 | **911E** | 鄞 |
| 17184 | **911F** | 鄟 |
| 17185 | **9120** | 鄠 |
| 17186 | **9121** | 鄡 |
| 17187 | **9122** | 鄟 |
| 17188 | **9123** | 鄟 |

(2) 將 Unicode 和大五碼(BIG-5)進行對應

再來，本研究將系統可使用 Unicode 範圍和大五碼(BIG-5)進行對應，已提供字型正確對應，在此列舉 50 筆如表 6，經過本研究檢索排序後內容如附錄 B。

表 **6** 特殊字碼與Unicode進行對照表(樣本 50 字)

| 項目 | Unicode | 細明體 | BIG-5 | CMEX 標楷體 |
|---|---|---|---|---|
| 17140 | **90F3** | 郒 | **D7ED** | 郒 |
| 17141 | **90F4** | 郴 | **D7EB** | 郴 |
| 17142 | **90F5** | 郵 | **B66C** | 郵 |
| 17143 | **90F6** | 郶 | | 郶 |
| 17144 | **90F7** | 鄉 | **97E8** | 鄉 |
| 17145 | **90F8** | 鄚 | | 鄚 |
| 17146 | **90F9** | 鄖 | **DC56** | 鄖 |
| 17147 | **90FA** | 鄨 | **EBD4** | 鄨 |
| 17148 | **90FB** | 鄡 | **DC57** | 鄡 |
| 17149 | **90FC** | 鄟 | **DC54** | 鄟 |
| 17150 | **90FD** | 都 | **B3A3** | 都 |
| 17151 | **90FE** | 鄅 | **B66E** | 鄅 |
| 17152 | **90FF** | 鄙 | **DC53** | 鄙 |
| 17153 | **9100** | 鄐 | **DC59** | 鄐 |
| 17154 | **9101** | 鄘 | **DC58** | 鄘 |
| 17155 | **9102** | 鄂 | **B66B** | 鄂 |
| 17156 | **9103** | 鄃 | **DC5C** | 鄃 |
| 17157 | **9104** | 鄄 | **DC52** | 鄄 |
| 17158 | **9105** | 鄅 | **DC5B** | 鄅 |
| 17159 | **9106** | 鄆 | **DC50** | 鄆 |
| 17160 | **9107** | 鄋 | **DC5A** | 鄋 |
| 17161 | **9108** | 鄍 | **DC55** | 鄍 |
| 17162 | **9109** | 鄉 | **B66D** | 鄉 |
| 17163 | **910A** | 鄋 | **9A55** | 鄋 |
| 17164 | **910B** | 鄚 | **E0AA** | 鄚 |
| 17165 | **910C** | 鄘 | **9C41** | 鄘 |
| 17166 | **910D** | 鄖 | **E0A5** | 鄖 |
| 17167 | **910E** | 鄔 | **E0AB** | 鄔 |
| 17168 | **910F** | 鄑 | **E0A6** | 鄑 |
| 17169 | **9110** | 鄐 | **E0A4** | 鄐 |
| 17170 | **9111** | 鄑 | **E0A7** | 鄑 |
| 17171 | **9112** | 鄒 | **B951** | 鄒 |
| 17172 | **9113** | 鄗 | **9C42** | 鄗 |
| 17173 | **9114** | 鄔 | **E0A9** | 鄔 |
| 17174 | **9115** | 鄉 | **9C43** | 鄉 |
| 17175 | **9116** | 鄖 | **E0A8** | 鄖 |

| 項目 | Unicode | 細明體 | BIG-5 | CMEX 標楷體 |
|---|---|---|---|---|
| 17176 | **9117** | 鄗 | **B952** | 鄗 |
| 17177 | **9118** | 鄘 | **BBC1** | 鄘 |
| 17178 | **9119** | 鄙 | **BBC0** | 鄙 |
| 17179 | **911A** | 鄚 | **E46E** | 鄚 |
| 17180 | **911B** | 鄴 | **E471** | 鄴 |
| 17181 | **911C** | 鄘 | **E469** | 鄘 |
| 17182 | **911D** | 鄝 | **E46D** | 鄝 |
| 17183 | **911E** | 鄞 | **BBC2** | 鄞 |
| 17184 | **911F** | 鄟 | **E46C** | 鄟 |
| 17185 | **9120** | 鄠 | **E46A** | 鄠 |
| 17186 | **9121** | 鄡 | **E470** | 鄡 |
| 17187 | **9122** | 鄢 | **E46B** | 鄢 |
| 17188 | **9123** | 鄤 | **E468** | 鄤 |

(3) 將 Extension Unicode 和細明體進行對應

本研究將系統可使用 Extension Unicode 範圍和大五碼(BIG-5)進行對應，已提供字型正確對應，在此列舉 50 筆如表 7，經過本研究檢索排序後之內容如附錄 C。

**表 7** 特殊字碼Extension Unicode與細明體對照表(樣本 50 字)

| 項目 | Unicode | 細明體 |
|---|---|---|
| 6520 | **4D77** | 揿 |
| 6521 | **4D78** | 氂 |
| 6522 | **4D79** | 氍 |
| 6523 | **4D7A** | 塴 |
| 6524 | **4D7B** | 鼏 |
| 6525 | **4D7C** | 灨 |
| 6526 | **4D7D** | 爇 |
| 6527 | **4D7E** | 鼓 |
| 6528 | **4D7F** | 鼗 |
| 6529 | **4D80** | 鼙 |
| 6530 | **4D81** | 鼛 |
| 6531 | **4D82** | 齖 |
| 6532 | **4D83** | 齤 |

| 項目 | Unicode | 細明體 |
|---|---|---|
| 6533 | **4D84** | 鼗 |
| 6534 | **4D85** | 鼙 |
| 6535 | **4D86** | 鼛 |
| 6536 | **4D87** | 鼢 |
| 6537 | **4D88** | 鼤 |
| 6538 | **4D89** | 鼷 |
| 6539 | **4D8A** | 齀 |
| 6540 | **4D8B** | 齁 |
| 6541 | **4D8C** | 齙 |
| 6542 | **4D8D** | 齚 |
| 6543 | **4D8E** | 齛 |
| 6544 | **4D8F** | 齴 |
| 6545 | **4D90** | 齾 |
| 6546 | **4D91** | 龑 |
| 6547 | **4D92** | 龒 |
| 6548 | **4D93** | 龖 |
| 6549 | **4D94** | 龗 |
| 6550 | **4D95** | 龞 |
| 6551 | **4D96** | 龡 |
| 6552 | **4D97** | 龢 |
| 6553 | **4D98** | 龣 |
| 6554 | **4D99** | 龤 |
| 6555 | **4D9A** | 龥 |
| 6556 | **4D9B** | 龦 |
| 6557 | **4D9C** | 龧 |
| 6558 | **4D9D** | 龨 |
| 6559 | **4D9E** | 龩 |
| 6560 | **4D9F** | 龪 |
| 6561 | **4DA0** | 龫 |
| 6562 | **4DA1** | 龬 |
| 6563 | **4DA2** | 龭 |
| 6564 | **4DA3** | 龮 |
| 6565 | **4DA4** | 龯 |
| 6566 | **4DA5** | 龰 |
| 6567 | **4DA6** | 龱 |
| 6568 | **4DA7** | 龲 |

(4) 開發字型轉換 API

為了有效將為來各醫院之自造字型與中推會公告之區碼進行對應，本研究與開發相關部分 API 轉換程式，如下：

a. 判斷各種字型編碼現行區間最大值與最小值程式碼。

```
#define LINE_MAX_CHARS    256
#define MAX_PHONE_NUM      6
#define MAX_B5P_INDEX     (126*190)
#define MAX_TAX_INDEX      37000
#define MAX_CNS_INDEX     ((ULONG)16*8836)
#define MAX_UCS_INDEX     (3*65536)
#define MAX_DCI_INDEX     (3*8836)
#define MAX_IBM_INDEX     ((ULONG)190*189)
#define MAX_B5G_INDEX     (126*157)
// added on 2012.11.26
#define MAX_JIS_INDEX     (4*8836)
#define MAX_GAIJI_NDX      6217
#define MAXDB_NO           28053
#define B5P_NDX_FILE       "B5P_NDX.DB"
#define TAX_NDX_FILE       "TAX_NDX.DB"
#define CNS_NDX_FILE       "CNS_NDX.DB"
#define UCS_NDX_FILE       "UCS_NDX.DB"
#define DCI_NDX_FILE       "DCI_NDX.DB"
#define IBM_NDX_FILE       "IBM_NDX.DB"
#define B5G_NDX_FILE       "B5G_NDX.DB"
// added on 2005.8.12 by james
#define JIS_NDX_FILE       "JIS_NDX.DB"
#define ALL_DB_FILE        "ALL_INF.DB"
#define B5P                 0
#define TAX                 1
#define CNS                 2
#define UCS                 3
#define DCI                 4
#define IBM                 5
#define B5G                 6
// added on    2012.11.26
#define JIS                 9
#define ALL                 255
#define NON_DEFINED_NDX 0xFFFF
#define NON_MAP_CODE       0x0000
#define Between(code, a, b)        ((code >= a) && (code <= b))
#define InCJK(c)                   (Between(c,0x0020,0x33FF) || Between(c,0x4E00,0xFFFF))
// added on 2005.8.12 by james
#define JIS1(high, low)            (Between(high, 0x21, 0x7E) && Between(low, 0x21, 0x7E))
#define JIS2(high, low)            (Between(high, 0xA1, 0xFE) && Between(low, 0xA1, 0xFE))
#define JIS3(high, low)            (Between(high, 0x21, 0x7E) && Between(low, 0xA1, 0xFE))
```

```
#define JIS4(high, low)            (Between(high, 0xA1, 0xFE) && Between(low, 0x21, 0x7E))

typedef struct {
   USHORT Big5E;
   USHORT Big5p;
   CHAR     Radical;
   CHAR     Stroke;
   USHORT Jis;
   ULONG    CNSC;
   ULONG    UCS4;
   USHORT NewDCI;
   USHORT IBMC;
   CHAR     cj_code[6];
   CHAR     phone_num;
   CHAR     ph_code[MAX_PHONE_NUM][5];
} DB_BIN_S;

typedef struct {
   CHAR    Big5p_str[5];
   CHAR    Radical_str[4];
   CHAR    Stroke_str[4];
   CHAR    Jis_str[5];
   CHAR    CNS_str[6];
   CHAR    UCS4_str[6];
   CHAR    NewDCI_str[6];
   CHAR    IBM_str[5];
   CHAR    cj_str[11];
   CHAR    phone_num;
   CHAR    ph_str[MAX_PHONE_NUM][9];
} DB_STR_S;

#endif
```

b. 特殊字型測試樣本檔

```
import www.dynacw.com;
public class Sample {
      public static void main(String[] args) throws Exception{
            // TODO : Convert Class Sample.
            Convert conv = new Convert();
            /*
              * TODO Conver to File Function.
              * Code Type
                  B5P : BIG5P
                  CNS : CNS
                  UCS : Unicode
                  DCI : NewDCI
                  IBM : IBM
                  UT8 : UTF8
                  JIS : JIS
                  EBCDIC : EBCDIC
```

```
                Parameters:
                    1. SrcFile         : FileName of Source File.
                    2. DestFile        : FileName of Target File.
                    3. src_t Code    : Type of Source Type.
                    4. dest_t Code : Type of Target Type.
            */
            conv.Convert2File("sampleFile/sample_b5p.txt",    "sampleFile/b5p2jis.txt",    conv.B5P,
conv.JIS);
            /*
              * TODO Conver to String Function.
              * Code Type
                    B5P : BIG5P
                    CNS : CNS
                    UCS : Unicode
                    DCI : NewDCI
                    IBM : IBM
                    UT8 : UTF8
                    JIS : JIS
                    EBCDIC : EBCDIC

                Parameters:
                    1.Src       : String Source Byte Array.
                    2.src_t    : Code Type of Source Type.
                    3.dest_t : Code Type of Target Type.
              */
            byte[] Src = new byte[2];
            byte[] Dest;
            int Return_Len = 0;
            Src[0] = (byte)0xB5;
            Src[1] = (byte)0xD8;

            /*
              * ConvertString -> Return String Buffer Length.
              * The return value is smaller than or equal zero, make sure your input type encoding.
              */
            if(conv.ConvertString(Src, conv.B5P, conv.JIS) != 0){
                Dest = new byte[Return_Len];
                // Get String Buffer Info.
                Dest = conv.get_ResuleStringBuff();

                System.out.println(Return_Len);          // Print String Buffer Length.
                System.out.println(Dest[0]);
                System.out.println(Dest[1]);
            }
        }
}
```

C. 自行開發的架接 API 程式部分程式碼

```
#include <stdlib.h>
#include <stdio.h>
//#include <io.h>
#include <string.h>
#include <fcntl.h>
#include <sys/stat.h>
//#include <dos.h>
#include "def.h"
void       Get_Hex_Str(USHORT val, CHAR *str)
{
   if (val <= 0x000F) {
      sprintf(str, "000%1X", val);
   }
   else if (val <= 0x00FF) {
      sprintf(str, "00%2X", val);
   }
   else if (val <= 0x0FFF) {
      sprintf(str, "0%3X", val);
   }
   else {
      sprintf(str, "%4X", val);
   }
}
void       Get_Dec_Str(USHORT val, CHAR *str)
{
   if (val <= 9) {
      sprintf(str, "000%1u", val);
   }
   else if (val <= 99) {
      sprintf(str, "00%2u", val);
   }
   else if (val <= 999) {
      sprintf(str, "0%3u", val);
   }
   else {
      sprintf(str, "%u", val);
   }
}
CHAR       InStandard_Big5(USHORT code)
{
   BYTE    high, low;

   high = (code & 0xFF00) >> 8;
   low = code & 0xFF;
   if (Between(low, 0x40, 0x7E) || Between(low, 0xA1, 0xFE)) {
      if (Between(high, 0xA3, 0xA3) && Between(low,    0xC0, 0xFE)) return(FALSE);
      if (Between(high, 0xA1, 0xC5) || Between(high, 0xC9, 0xF9)) return(TRUE);
      else if (Between(high, 0xC6, 0xC6) && Between(low, 0x40, 0x7E)) return(TRUE);
      return(FALSE);
```

```
    }
    return(FALSE);
}

/* ===== Compute/Get Big5E Information ===== */
SHORT      GetB5G_Index(USHORT code, USHORT *index)
{
    USHORT              high, low;

    high = (code & 0xFF00) >> 8;
    low = code & 0xFF;

    *index = 0;
    if (!(Between(low, 0x40, 0x7E) || Between(low, 0xA1, 0xFE))) return(ILLEGAL_CODE);
    if (!(Between(high, 0x81, 0xFE))) return(ILLEGAL_CODE);

    if (Between(low, 0xA1, 0xFE)) {
        low -= 0x22;
    }
    *index = (high - 0x81) * 157 + (low - 0x40);
    return(OK);
}
SHORT      GetB5G_Code(USHORT index, USHORT *code, CHAR *str)
{
    USHORT              high, low;

    strcpy(str, "");
    *code = 0;
    if (index >= MAX_B5G_INDEX) return(ILLEGAL_CODE);

    high = index / 157;
    low = index % 157;

    if (low > 0x3E) low += 0x22;

    *code = (high + 0x81) * 256 + (low + 0x40);
    sprintf(str, "%X", *code);
    return(OK);
}

/* ===== Compute/Get Big5P Information ===== */
SHORT      GetB5P_Index(USHORT code, USHORT *index)
{
    USHORT              high, low;

    high = (code & 0xFF00) >> 8;
    low = code & 0xFF;

    *index = 0;
    if (!(Between(low, 0x40, 0x7E) || Between(low, 0x80, 0xFE))) return(ILLEGAL_CODE);
    if (!(Between(high, 0x81, 0xFE))) return(ILLEGAL_CODE);
```

```
    if (Between(low, 0x80, 0xFE)) {
        low --;
    }
    *index = (high - 0x81) * 190 + (low - 0x40);
    return(OK);
}
SHORT       GetB5P_Code(USHORT index, USHORT *code, CHAR *str)
{
    USHORT              high, low;

    strcpy(str, "");
    *code = 0;
    if (index >= MAX_B5P_INDEX) return(ILLEGAL_CODE);

    high = index / 190;
    low = index % 190;

    if (low > 0x3E) low ++;

    *code = (high + 0x81) * 256 + (low + 0x40);
    sprintf(str, "%X", *code);
    return(OK);
}

/* ===== Compute/Get TAX Information ===== */
SHORT       GetTAX_Index(CHAR *str, USHORT *index)
{
    CHAR    c1, c2, c3, c4;
    int i;

    *index = 0;

    for (i=0;i<strlen(str);i++)
        str[i]=toupper(str[i]);

    if (strlen(str) != 4) return(ILLEGAL_CODE);
    c1 = str[0];
    c2 = str[1];
    c3 = str[2];
    c4 = str[3];
    if (!(Between(c1, '0', '9') || Between(c1, 'A', 'Z'))) return(ILLEGAL_CODE);
    if (!(Between(c2, '0', '9'))) return(ILLEGAL_CODE);
    if (!(Between(c3, '0', '9'))) return(ILLEGAL_CODE);
    if (!(Between(c4, '0', '9'))) return(ILLEGAL_CODE);

    if (Between(c1, '0', '9')) *index = c1 - '0';
    else *index = c1 - 'A' + 10;

    *index = (*index)* 1000 + (c2 - '0') * 100 + (c3 - '0') * 10 + (c4 - '0');
    return(OK);
}
SHORT       GetTAX_Code(USHORT index, CHAR *str)
```

```
{
   CHAR    c1, c2, c3, c4;
   strcpy(str, "");
   if (index >= MAX_TAX_INDEX) return(ILLEGAL_CODE);

   c1 = index / 1000;
   index %= 1000;
   c2 = index / 100;
   index %= 100;
   c3 = index / 10;
   index %= 10;
   c4 = index;

   if (c1 >= 10) c1 = c1 - 10 + 'A';
   else c1 = c1 + '0';
   c2 += '0';
   c3 += '0';
   c4 += '0';
   sprintf(str, "%c%c%c%c", c1, c2, c3, c4);
   return(OK);
}

/* ===== Compute/Get CNS Information ===== */
SHORT     GetCNS_Index(CHAR *str, ULONG *index)
{
   CHAR                plane, code_str[5];
   USHORT               code, high, low;

   *index = 0;
   int i;

   for (i=0;i<strlen(str);i++)
      str[i]=toupper(str[i]);

   if (strlen(str) != 5) return(ILLEGAL_CODE);
   plane = str[0];
   if (!(Between(plane, '1', '9') || Between(plane, 'A', 'Z'))) return(ILLEGAL_CODE);
   memcpy(code_str, &str[1], 4);
   code_str[4] = 0;
   sscanf(code_str, "%2X%2X", &high , &low);
   if (!(Between(high, 0x21, 0x7E) && Between(low, 0x21, 0x7E))) return(ILLEGAL_CODE);

   if (Between(plane, '1', '9')) plane = plane - '0';
   else plane = plane - 'A' + 10;
   plane --;
   *index = (ULONG)plane * 8836 + (high - 0x21) * 94 + (low - 0x21);
   return(OK);
}
SHORT     GetCNS_Code(ULONG index, USHORT *cpl, USHORT *code, CHAR *str)
{
   USHORT               plane, high, low;
```

```
    strcpy(str, "");
    if (index >= MAX_CNS_INDEX) return(ILLEGAL_CODE);

    plane = index / 8836;
    index %= 8836;
    high = (index / 94) + 0x21;
    low =    (index % 94) + 0x21;

    sprintf(str, "%1X%2X%2X", plane+1, high, low);
    *cpl = plane + 1;
    *code = high * 256 + low;
    return(OK);
}

/* ===== Compute/Get UCS Information ===== */
SHORT      GetUCS4_Index(CHAR *str, ULONG *index)
{
    CHAR                ucs2_str[10];
    USHORT              high_word, low_word, len;

    *index = 0;
    len = strlen(str);
    if (len < 4) return(ILLEGAL_CODE);

    high_word = 0x0000;
    low_word = 0x0000;
    memcpy(ucs2_str, &str[len-4], 4);
    ucs2_str[4] = 0;
    sscanf(ucs2_str, "%x", &low_word);
    memcpy(ucs2_str, &str[0], len-4);
    ucs2_str[len-4] = 0;
    sscanf(ucs2_str, "%x", &high_word);

    *index = (ULONG)(high_word) << 16;
    *index += low_word;
    return(OK);
}
SHORT      GetUCS4_Code(ULONG index, CHAR *str)
{
    USHORT              high_word, low_word;
    CHAR                ucs2_str[5];

    strcpy(str, "");
    if (index >= MAX_UCS_INDEX) return(ILLEGAL_CODE);
    high_word = index >> 16;
    low_word = index & 0xFFFF;
    Get_Hex_Str(low_word, ucs2_str);
    sprintf(str, "%1X%s", high_word, ucs2_str);
    return(OK);
}

/* ===== Compute/Get DCI Information ===== */
```

```
SHORT     GetDCI_Index(CHAR *str, USHORT *index)
{
   CHAR                plane, ndx_str[5];
   USHORT              ndx, high, low;

   *index = 0;
   int i;

   for (i=0;i<strlen(str);i++)
      str[i]=toupper(str[i]);

   if (strlen(str) != 5) return(ILLEGAL_CODE);
   plane = str[0] - '0';
   if (!(Between(plane, 0, 2))) return(ILLEGAL_CODE);
   memcpy(ndx_str, &str[1], 4);
   ndx_str[4] = 0;
   sscanf(ndx_str, "%u", &ndx);
   if (ndx >= 8836) return(ILLEGAL_CODE);

   *index = plane * 8836 + ndx;
   return(OK);
}
SHORT     GetDCI_Code(USHORT index, USHORT *code, CHAR *str)
{
   USHORT              plane, high, low;
   CHAR                tmp_str[5];

   strcpy(str, "");
   *code = 0;
   if (index >= MAX_DCI_INDEX) return(ILLEGAL_CODE);

   plane = index / 8836;
   index %= 8836;
   Get_Dec_Str(index, tmp_str);
   sprintf(str, "%1u%s", plane, tmp_str);
   high = (index / 94) + 0x21;
   low =    (index % 94) + 0x21;
   *code = high *256 + low;
   if (plane == 0) *code += 0x8080;
   else if (plane == 1) *code += 0x8000;
   return(OK);
}

/* ===== Compute/Get IBM Information ===== */
SHORT     GetIBM_Index(USHORT code, USHORT *index)
{
   USHORT              high, low;

   high = (code & 0xFF00) >> 8;
   low = code & 0xFF;

   *index = 0;
```

```
    if (!(Between(low, 0x40, 0x7F) || Between(low, 0x81, 0xFD))) return(ILLEGAL_CODE);
    if (!(Between(high, 0x40, 0xFD))) return(ILLEGAL_CODE);

    if (Between(low, 0x81, 0xFD)) {
        low --;
    }
    *index = (high - 0x40) * 189 + (low - 0x40);
    return(OK);
}
SHORT      GetIBM_Code(USHORT index, USHORT *code, CHAR *str)
{
    USHORT              high, low;

    strcpy(str, "");
    *code = 0;
    if (index >= MAX_IBM_INDEX) return(ILLEGAL_CODE);

    high = index / 189;
    low = index % 189;

    if (low > 0x3F) low ++;

    *code = (high + 0x40) * 256 + (low + 0x40);
    sprintf(str, "%X", *code);
    return(OK);
}
```

d. 自動判別其字碼區間

```
#include <stdio.h>
//#include <io.h>
#include <fcntl.h>
#include <string.h>
#include <sys/stat.h>
//#include <alloc.h>
//#include <dos.h>

#include "def.h"
#include "convert.h"

#define SOURCE_NDX          1
#define TARGET_NDX          2
#define SRC_CODE_NDX        3
#define DEST_CODE_NDX       4
#define MAX_ARGC            5

int all_db_max_count;
BYTE DB1[MAX_UCS_INDEX*2+1];
DB_BIN_S ALLDB[MAXDB_NO+1];

USHORT    GetCodeType(CHAR *str)
{
```

```
    int i;

    // capitalize str
    for (i=0;i<strlen(str);i++)
        str[i]=toupper(str[i]);

    if (strcmp(str, "B5P") == 0) {
        return(B5P);
    }
    else if (strcmp(str, "TAX") == 0) {
        return(TAX);
    }
    else if (strcmp(str, "CNS") == 0) {
        return(CNS);
    }
    else if (strcmp(str, "UCS") == 0) {
        return(UCS);
    }
    else if (strcmp(str, "DCI") == 0) {
        return(DCI);
    }
    else if (strcmp(str, "IBM") == 0) {
        return(IBM);
    }
    else if (strcmp(str, "B5G") == 0) {
        return(B5G);
    }
    else if (strcmp(str, "JIS") == 0) {
        return(JIS);
    }
    return(B5P);
}

main(argc, argv)
USHORT    argc;
CHAR      *argv[];
{
    FILE              *fp,*tp;
    int               status;
    USHORT            src_type, dest_type, alpha_type=0;
    USHORT            ret_val;

    USHORT            argc_used;

    printf("DynaComware Code Convert Utility, Ver. 1.00\n");
    if (argc < MAX_ARGC) {  // MAX_ARGC=5
        printf(" Arguments Error\n\n");
        printf("Format : %s arg1 arg2 arg3 arg4 [/ALPHA d]\n", argv[0]);
        printf("       where arg1 : Filename of Source File\n");
        printf("             arg2 : Filename of Target File\n");
        printf("             arg3 : Code Type of Source File\n");
        printf("             arg4 : Code Type of Target File\n");
```

```
        printf("       Code Type : B5P : BIG5P\n");
        printf("                 : JIS : JIS\n");
        printf("                 : CNS : CNS\n");
        printf("                 : UCS : Unicode\n");
        printf("                 : DCI : NewDCI\n");
        printf("                 : IBM : IBM\n");
        printf(" Option :\n");
        printf(" /ALPHA    d : d is type for convertion when encounter alphabet\n");
        printf("              0 : No Action\n");
        printf("              1 : Full to Half\n");
        printf("              2 : Half to Full\n");
        printf("\n\n");
        exit(0);
    }
/* ignore alpha_type, remarked by kywu on 2005/8/10
    argc_used = MAX_ARGC;              // MAX_ARGC = 5
    alpha_type = NO_CONVERT;  // NO_CONVERT = 0
    while (argc_used < argc) {
        if (stricmp(argv[argc_used], "/ALPHA d") == 0) {
            sscanf(argv[argc_used+1], "%u", &alpha_type);
            if (alpha_type > MAX_ALPHA) {          // MAX_ALPHA = 2
                alpha_type = NO_CONVERT;
            }
            argc_used += 2;
        }
        else {
            argc_used ++;
        }
    }
*/
    src_type = GetCodeType(argv[SRC_CODE_NDX]);
    dest_type = GetCodeType(argv[DEST_CODE_NDX]);

    if (src_type == dest_type) {
        printf("Source & Target Code Type Is Same, Skip.....\n");
        exit(0);
    }
/*
    if (src_type == TAX || src_type == UCS || dest_type == TAX || dest_type == UCS) {
    }
    else {
        printf("Such Code Convertion Not Support Now, Skip.....\n");
        exit(0);
    }
*/

    status = LoadDB1(src_type);
    if (status<0) {
        printf("Error : LoadDB failed. (%d)\n",status);
        exit(0);
    }
    status = LoadALLDB();
```

```
    if (status<0) {
        printf("Error : LoadDB failed. (%d)\n",status);
        exit(0);
    }

    if ((fp = fopen(argv[SOURCE_NDX], "rb")) == NULL) {
        fprintf(stderr, "Can't Open Target File %s\n", argv[SOURCE_NDX]);
        exit(0);
    }
    if ((tp = fopen(argv[TARGET_NDX], "wb")) == NULL) {
        fprintf(stderr, "Can't Open Target File %s\n", argv[TARGET_NDX]);
        fclose(fp);
        exit(0);
    }
    ret_val = Transfer(fp, tp, src_type, dest_type, alpha_type);
    if (!ret_val) {
        printf("Convert Incomplete.....(File Maybe Error)\n\n");
    }
    else {
        printf("Convert Complete.....\n\n");
    }
    fclose(fp);
    fclose(tp);
}
//---------------------------------------------------------------------------

int LoadDB1(int type)
{
    FILE                *fd;
    CHAR                  filename[20];
    long                recno;


    switch (type) {
        case B5P :
            strcpy(filename, B5P_NDX_FILE);
            break;
        case TAX :
//        case TAX2D :
            strcpy(filename, TAX_NDX_FILE);
            break;
        case CNS :
            strcpy(filename, CNS_NDX_FILE);
            break;
        case UCS :
//        case UT8 :
            strcpy(filename, UCS_NDX_FILE);
            break;
        case DCI :
            strcpy(filename, DCI_NDX_FILE);
            break;
        case IBM :
```

```
            strcpy(filename, IBM_NDX_FILE);
            break;
        case B5G :
            strcpy(filename, B5G_NDX_FILE);
            break;
        case JIS :
            strcpy(filename, JIS_NDX_FILE);
            break;
    }

    if ((fd = fopen(filename, "rb")) == NULL){
        printf("Can not open file: %s\n",filename);
        return(LOADDBFAIL);
    }
    recno = fread(DB1, 1,MAX_UCS_INDEX, fd);
    //printf("recno=%d\n",recno);
    fclose(fd);



    if (recno == 0)
         return(LOADDBFAIL);
    else
         return(TRUE);

}
//---------------------------------------------------------------------------
int LoadALLDB(void )
{
    FILE                *fd;
    long            index,offset;
    char                buf[400000];
    unsigned short twobyte;
    unsigned char               byte1,byte2,byte3,byte4;
    int                 fourbyte;
    int ndx;
    long                recno;

    if ((fd = fopen(ALL_DB_FILE, "rb")) == NULL){
        printf("Can not open file: %s\n",ALL_DB_FILE);
        return(LOADDBFAIL);
    }
    //all_db_max_count = fread(ALLDB, sizeof(DB_BIN_S), MAXDB_NO, fd);

    recno = fread(buf, 1,400000, fd);
    all_db_max_count=recno;
    fclose(fd);

    index=0;
    offset =0;
    while (recno>0) {
//          memcpy(&ALLDB[index],buf+offset,57);
```

```
//        memcpy(&ALLDB[index].Big5E ,buf+offset,2); offset +=2;
        memcpy(&byte1 ,buf+offset,1); offset +=1;        memcpy(&byte2 ,buf+offset,1); offset +=1;
        ALLDB[index].Big5E = byte2 *256 + byte1;
        memcpy(&byte1 ,buf+offset,1); offset +=1;        memcpy(&byte2 ,buf+offset,1); offset +=1;
        ALLDB[index].Big5p = byte2 *256 + byte1;
        memcpy(&ALLDB[index].Radical,buf+offset,1); offset +=1;
        memcpy(&ALLDB[index].Stroke,buf+offset,1); offset +=1;
        memcpy(&byte1 ,buf+offset,1); offset +=1;        memcpy(&byte2 ,buf+offset,1); offset +=1;
        ALLDB[index].Jis = byte2 *256 + byte1;
        memcpy(&byte1 ,buf+offset,1); offset +=1;        memcpy(&byte2 ,buf+offset,1); offset +=1;
        memcpy(&byte3 ,buf+offset,1); offset +=1;        memcpy(&byte4 ,buf+offset,1); offset +=1;
        ALLDB[index].CNSC                                                                    =
(long)(byte4*256*256*256)+(long)(byte3*256*256)+(long)(byte2*256)+byte1;
        memcpy(&byte1 ,buf+offset,1); offset +=1;        memcpy(&byte2 ,buf+offset,1); offset +=1;
        memcpy(&byte3 ,buf+offset,1); offset +=1;        memcpy(&byte4 ,buf+offset,1); offset +=1;
        ALLDB[index].UCS4                                                                    =
(long)(byte4*256*256*256)+(long)(byte3*256*256)+(long)(byte2*256)+byte1;
        offset +=41;
        index++;
        recno = recno-57;
    }
       //test begin
    ndx=0;
    printf("test ALLDB[%d]\n",ndx);
    printf("big5e,big5p,radical                :                %d                %d                %c
\n",ALLDB[ndx].Big5E,ALLDB[ndx].Big5p,ALLDB[ndx].Radical);
    printf(",stroke,jis,cns,ucs4                        :            %d            %d            %d            %d
\n",ALLDB[ndx].Stroke,ALLDB[ndx].Jis,ALLDB[ndx].CNSC,ALLDB[ndx].UCS4);
    ndx=1;
    printf("test ALLDB[%d]\n",ndx);
    printf("big5e,big5p,radical                :                %d                %d                %c
\n",ALLDB[ndx].Big5E,ALLDB[ndx].Big5p,ALLDB[ndx].Radical);
    printf(",stroke,jis,cns,ucs4                        :            %d            %d            %d            %d
\n",ALLDB[ndx].Stroke,ALLDB[ndx].Jis,ALLDB[ndx].CNSC,ALLDB[ndx].UCS4);
    //test end

    printf("db max count :[%d]\n",all_db_max_count);

    if (all_db_max_count == 0)
        return(LOADDBFAIL);
    else
        return(TRUE);
}
```

E. 轉換測試程式

```
#ifndef DEF_CONVERT
#define DEF_CONVERT

#define LONG        long
#define ULONG       unsigned long
#define SHORT       short
#define USHORT      unsigned short
#define CHAR        char
#define BYTE        unsigned char

// No Map Code :
// Big5plus          : 0x99D6              1
// Tax (4A)          : "I250"              2
// CNS exchange : SO 2121            3
// IBM               : 0xF9C3              4
// NewDCI            : 0xA1A1              5
// CDC (C4)          : "^J8E"              6
// EUC (2B)          : 0xA1A1              7
// EUC (4B)          : 0x8EA1A1A1          8
// CNS (5B)          : "12121"             9
// Big5/Big5E        : 0xxA140             A
// Tax (2D)          : 0xE245              B
// EBCDIC            : 0x40                C
// ASCII             : 0x20                D
// Tax (4E)          : C9 F2 F5 F0         E

#define ERROR_B5P_CODE    0x99D6
#define ERROR_TAX_STR     "I250"
#define ERROR_CNS_CODE    0x2121
#define ERROR_UCS_CODE    0x3000
#define ERROR_DCI_CODE    0xA1A1
#define ERROR_IBM_CODE    0xF9C3
#define ERROR_B5G_CODE    0xA140
// 2005.8.12 added by james
#define ERROR_JIS_CODE    0x2121

#define ERROR_UNIC_CODE 0x2605              /* Error Unicode Code */

#define ERROR_CTRL_CODE 0x20                /* Error Control Code */
#define NO_MAP_CODE       0x2178            /* CNS -> BIG5 : No Mapping    */

#define OVER_LEN_CODE     0x217A            /* CNS -> BIG5 : Over Length */
#define NO_MAP_CODE1      0x2177            /* BIG5 -> CNS : No Mapping    */
#define OVER_LEN_CODE1    0x2170            /* BIG5 -> CNS : Over Length */

#define UNIC_ROMAN        0x0100

#define NO_CONVERT        0
#define HALF_FULL         1
```

```
#define FULL_HALF            2
#define MAX_ALPHA            2

#define TRUE                 1
#define FALSE                0
#define ERROR_CODE           0x00
#define BIT_7                0x80
#define NBIT_7               0x7F
#define SYM                  0
#define PLANE1               1
#define PLANE2               2
#define ESC_MAX_COUNT        10

/*
#define B5P                  0
#define TAX                  1
#define CNS                  2
#define UCS                  3
#define DCI                  4
#define IBM                  5
#define B5G                  6
*/

#define LINE_MAX_CHARS    256

#define STEP0                0
#define STEP1                1
#define STEP2                2
#define STEP3                3

/* -------------------    Error Return Code -------------------- */
#define NORMAL                          0
#define MAPPING_TABLE_NOT_FIND    1
#define OVER_MAX_LEN                2
#define INPUT_LEN_TOO_LARGE         3
#define CHINESE_UNPAIR              4

#define MEMORY_NOT_ENOUGH           10
#define LEN_NOT_MATCH               11

#define LOADDBFAIL                  -21

/* -------------------    Assign & Transfer Plane ------------- */
#define G0                  0
#define G1                  1
#define G2                  2
#define G3                  3
#define ASCII               0                   /* ASCII mode */
#define PLANE_1             1                    /* 1st Palne For CNS */
#define PLANE_2             2                    /* 2nd Palne For CNS */
#define PLANE_3             3                    /* 3rd Palne For CNS */
#define PLANE_4             4                    /* 4th Palne For CNS */
```

```
#define PLANE_5              5                    /* 5th Palne For CNS */
#define PLANE_6              6                    /* 6th Palne For CNS */
#define PLANE_7              7                    /* 7th Palne For CNS */
#define PLANE_8              8                    /* 8th Palne For CNS */
#define PLANE_9              9                    /* 9th Palne For CNS */
#define PLANE_10             10                   /* 10th Palne For CNS */
#define PLANE_11             11                   /* 11th Palne For CNS */
#define PLANE_12             12                   /* 12th Palne For CNS */
#define PLANE_13             13                   /* 13th Palne For CNS */
#define PLANE_14             14                   /* 14th Palne For CNS */
#define PLANE_15             15                   /* 15th Palne For CNS */
#define PLANE_16             16                   /* 16th Palne For CNS */
#define UNDEFINE             255                   /* Undefine Plane */

#define ESC                  0x1B
#define TAB_C                0x09
#define LF                   0x0A
#define FF                   0x0C
#define CR                   0x0D
#define EOF_C                0x1A
/* ------------------   Transfer Plane   ---------------------- */
#define SO                   0x0E                  /* Lock Mode, Use G1 */
#define SI                   0x0F                  /* Lock Mode, Use G0 */
#define LS2                  0x6E                  /* Lock Mode, Use G2 */
#define LS3                  0x6F                  /* Lock Mode, Use G3 */
#define SS2                  0x4E                  /* Unlock Mode, Use G2 */
#define SS3                  0x4F                  /* Unlock Mode, Use G3 */
/* ------------------   Assign   Plane   ---------------------- */
#define HEX24H               0x24                  /* 2nd Byte For Assign Plane */
#define HEX28H               0x28
#define HEX29H               0x29
#define HEX2AH               0x2A
#define HEX2BH               0x2B                  /* 3rd Byte For Assign Plane */
#define PLANE_12C            0x3B
#define PLANE_13C            0x3C
#define PLANE_14C            0x3D
#define PLANE_15C            0x3E
#define PLANE_16C            0x3F
/* ------------------------------------------------------- */
SHORT Transfer(FILE *fp, FILE *tp, USHORT src_type, USHORT dest_type, USHORT alpha_flag);
#endif
```

**(5) 內嵌特殊中文字型元件之網頁實證結果**

使用者電腦端因 Windows 編碼並非是 CNS 編碼字型，並不相同，因此有提供給保險人或是外單位的應用系統功能部份，須提供轉製 PDF 檔案功能，先在 Web 伺服器端進行產生 PDF 後，才傳至 Client 端，如此一來即可正確呈現相關字型。本計畫中，測試部分頁面呈現結果如圖 10、圖 11。

B_1 測試第一筆資料


**圖 10** 本研究結果測試第一筆資料結果

B_2 測試第 2 筆資料


**圖 11** 本研究結果測試第二筆資料結果

# 5. 研究發現

## 5.1 本研究計畫發現

(1) 提供內部資訊系統面臨中文特殊字型一個好的解決方案

在本規劃中建議將本案之醫療機構使用字碼改為 CNS 編碼，因 CNS 字庫相當完整，因此在使用上會較少遇見須造字的部份；但還是可以接受造字需求，在 CNS 未提供該特殊字之前，可以先利用造字工作，存成影像檔字型，有特殊字展現需求時，則透過中文字判別 API，進行圖形字體下載，以便提供使用。重點摘要如下：

- 若不採用中推會 API(需授權費用)，則須自行架設圖形字型伺服器，亦可解決問題。
- CNS 尚未造字部份，透過圖形字檔進行存檔使用。
- 中文特殊判斷 API 僅針對重要欄位使用，如投保單位負責人、單位地址。
- 通訊錄、保險人資料、保險人地址等欄位，以避免系統效能上的問題，以減少網路負擔。

(2) 本計畫可以解決其他醫療單位或是民眾在家使用 Web 應用程式展現問題

民眾在家使用 Web 應用程式展現問題，因為無法判斷是否使用 CNS 字型，且大都個人電腦都是使用 Windows 內定字型，因此考量使用者功能需求，在本案之醫療機構資訊室同意開放的服務下，在前端設計使用 Web 開發工具來設計使用者展現頁面(會自動將頁面轉製成 PDF 型態)，使用者下載相關 PDF Reader 免費軟體即可使用；後端的部分與上方案例相同。繪製如下圖 12。

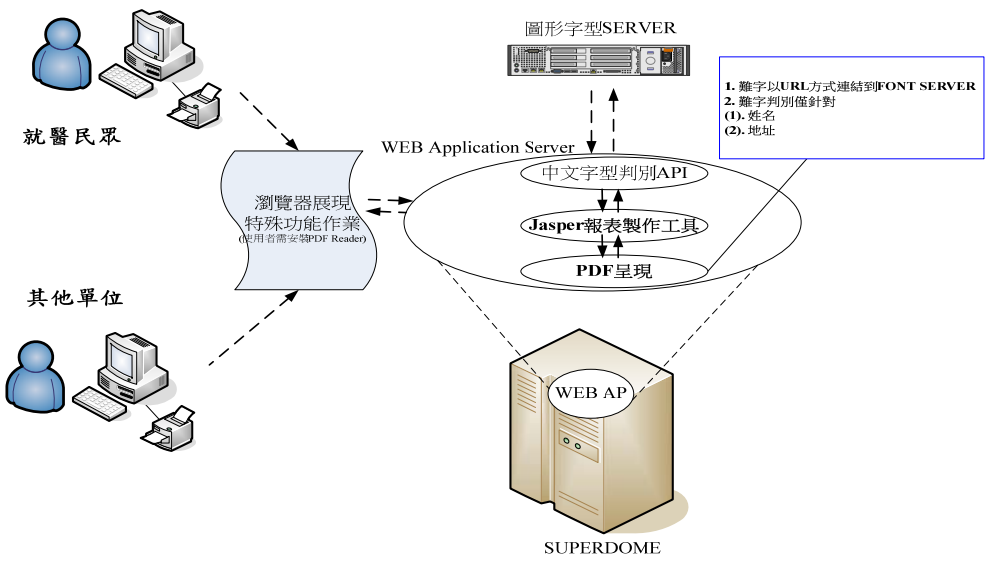
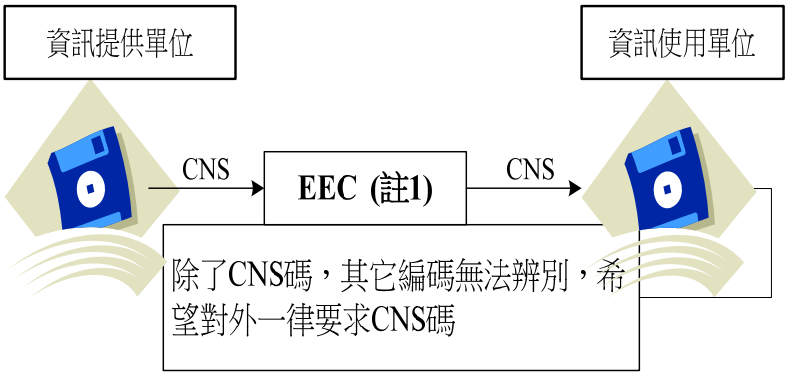



**圖 12** 本計畫外單位使用者中文字型使用流程圖

(3) 本計畫可解決各醫療機構資訊交換問題解決方案

有關應用系統下載提供檔案與各系統交換或是與外單位檔案資訊交換問題，將建議本案之醫療機構 HIS 相關系統以 CNS 字碼為標準；如使用 BIG-5、Unicode 3.0、EUC 等內碼之處理方式，不再因各單位自造字不同問題，無法進行交換的情勢發生，說明如圖 13。




註 1：EEC 指電子病歷互通交換平台

**圖 13** 與外單位交換資料示意圖

(3) 本計畫可解決有大量醫療統計報表列印需求之機構

本計劃醫療統計報表強調效能與列印之正確性，使用 XML 與 PRN 分流成果來處理報表，因此印表機如遇見特殊區段編碼之中文字型，將自動至本案之醫療機構現行中文字型伺服器中取用字型後列印，現行本案之醫療機構中文字型伺服器是採用 Unicode 3.0 編碼，建議更改為 CNS 編碼後，與線上程式使用相同之字型；但是如遇見 CNS 未造字型；則因為 PRN 檔中無法直接插入 URI 語法，因此須將相關需求內崁於報表設計程式碼中，透過 URI 後直接存取 URL 語言所需的特殊字型，說明如圖 14。

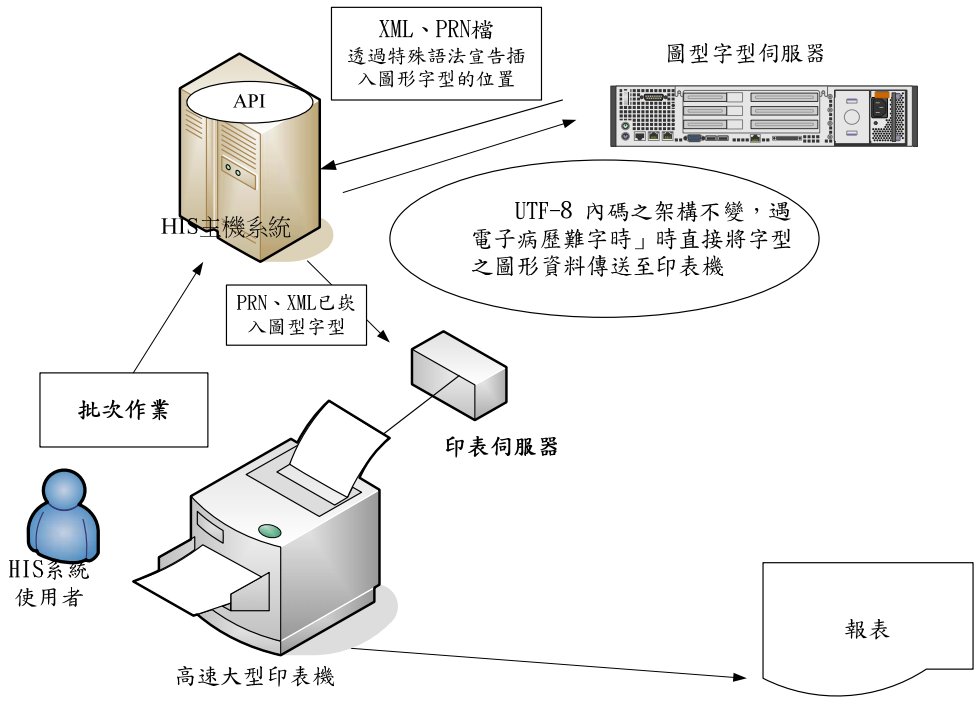



**圖 14** 本計畫規劃之醫療統計大量報表列印解決方案

## 5.2 研究發現成效

本研發成果預期可以讓報表以及使用電子病歷互通平台之醫療機構，得以展現自造字與罕見字，因此確實解決媒體交換與戶役政系統罕見字與自造字展現問題。

### 5.2.1 創新性

(1) 首創整合醫療資訊系統內碼與 BIG-5 碼入 Unicode 碼

在本研究計畫預期將原先使用醫療資訊系統內碼漢字自造字區約 1,732 字，與 BIG-5 平時與各機關團體交換所需之 2650 自造字區，進行比對整合，並建立關聯資料庫表單，並進行與戶役政機關所使用之特殊自造字區政整合，統一使用於 Unicode 碼特殊字碼區段。並由中文推行委員會所提供之特殊字圖形伺服器 URI 為標準，整合為一體，建置一個可以容納整合性醫療資訊系統內碼造字區與 BIG-5 造字整合之系統，此為國內地方單位第一完成之有效中文難字解決方案，其創新度可見非常可貴。

(2) 創新即時線上系統提供自造字與罕見字服務

採用 URI 方式連至中推會所提供之圖形伺服器，並提供民眾可以在家中連上系統即可得到正確之中文特殊造字字型，無論多罕見的字體，只要是 CNS 11643 收錄字體內展現幾乎無問題，CNS 11643 收納字外，也可提供由中推會造字後使用，與政府各機關媒體交換不會再有問題；民眾有了正確的中文字體的醫療繳費報表，申請各項補助或是各公務機關媒體交換更加便利。

### 5.2.2 前瞻性

本研發預期使用 URI 整合來處理中文自造字型的展現；未來將持續進行支援輔助器具輸入後，判斷特殊字型之系統發展；並結合 Web 2.0 之相關成果，來提供一個符合 HL7 標準之 XML 交換介面，相信對醫療體系將有更大貢獻。

### 5.2.3 實務應用上具體價值

(1) 解決異質系統中文字碼交換問題

本研發成果將於實務上進行應用，且對政府機構資訊部門之異質資訊系統平台之中文字碼交換，提供一個有效的解決方案，且借用中推會統一管理之圖形伺服器，不會有編碼不一致的問題。同時，讓政府公務機關之資訊系統由封閉系統走向開放系統過程中，解決大量中文字碼對照的問題。

(2) 增加各政府機關電子媒體交換正確度

原本政府公務機關之資訊系統，業務上需求，電子媒體交換非常頻繁，以往在遇見罕見字體或是自造字體時，通常會以一個特殊符號來替代(俗稱方塊字)，但此種方式，常常發生申請資料之民眾不滿，主要因素為如果方塊字出現在民眾之姓名，則通常無法至其他機關辦理公務；因此本研發成果讓印出之表單更正確，也讓各公務機關媒體交換更順暢。

(3) 處理中文字碼整合，不需額外添購伺服器或是工具

本研發成果提供一個有效的方案，讓需要進行中文字碼整合之機關團體或是中小企業，只要於原本資料庫中新增相關中文字型對照欄位，即可讓自造字或是罕見字呈現，而不需要再花費高額的設備或是工具購置成本，對資訊系統

服務業之貢獻非常大。

(4) 部門經營效益提升

此研發成果提供一個一勞永逸的中文字碼解決方案，也讓該政府機構內之資訊管理人員，不需要再費心進行中文特殊字型造字以及管理，也讓資訊系統更有成效，對於整體部門經營績效也有顯著的功效。

### 5.2.4 對教學實務上的貢獻

本研發計畫預期可以解決了中文字型交換碼與中文字型展現的重大議題，可應用於授課的教師在程式設計時對於中文系統使用轉碼之參考；對學生而言，可了解中文造字的奧妙，也讓學生學習一個中文轉碼實務上應用的最佳範例，而對程式設計類的學生而言，必然會很有助益。

### 5.2.5 對其它方面貢獻

本研究成果對社會大眾，能在家中放心的使用已授權之醫療證明或是申請病歷資料，不會再因為特殊字體無法呈現，而發生甲單位申請，乙單位不接受之問題，對於政府推行電子病歷之美意，使得到更好更精確的展現，也增加醫療機構服務的效率。

# 6. 結論與建議

## 6.1 研究成果

(1) 改善現行資料庫自造字碼處理方式

在此部分需要撰寫 CNS 與本案之醫療機構現行自造字對應碼轉碼 API，針對本案之醫療機構現有的自造字字型進行轉碼後置回，此一部分是為了驗證現行自造字碼都已經和 CNS 編碼字型，進行比對無誤。

(2) 應用程式頁面與報表處理圖形字型方式

在本案之醫療機構使用者 Web 頁面中崁入圖形字體，並無問題；但是有兩項成果：

(a) 本案之醫療機構報表要崁入圖形字，則與報表工具有關，現行報表工作採用 Client 端展現字體，但如此一來，得將報表工具產生字體前，即先行處理圖形字體，才能正確列印出圖形字體。

(b) 但若是使用者電腦端，則會因 Windows 編碼並非是 CNS 編碼字型，並不相同，因此有提供給保險人或是外單位的應用系統功能部份，須提供轉製 PDF 檔案功能，先在 Web 伺服器端進行產生 PDF 後，才傳至 Client 端，如此一來即可正確呈現相關字型。

(3) 圖形字型伺服器配置解決方式

目前電子病歷相關系統中，並無圖形伺服器之設備，最好的規劃是使用本案之醫療機構現行圖形伺服器，可以避免不同步問題；但是如果無法進行上述

方案，則須考量現行字型伺服器改版問題，且須經過嚴謹的測試與驗證。

(4) 醫療統計報表圖形字體解決方式

醫療統計報表因 PRN 檔中僅能放置純文字檔案，如要放置圖形字體，則須在產生報表的 XML 檔與 PRN 檔中，透過特殊語法宣告插入圖形字型的位置，同時，印表機控制程式亦須要加入對應的承接，如此可以解決醫療統計報表列印圖形檔案的問題。使用 UTF-8 內碼之架構不變，遇「難字」時直接將字型之圖形資料傳送至 KP 印表機。成果上方法如下：

(a) 由大型印表機中新增字型圖檔接收指令

(b) 指令格式如下:

ESC f HH VV FontImageData

說明: HH ---- 2 bytes，width of raster image，圖形資料水平點數；VV ---- 2 bytes，height of raster image，圖形資料垂直點數；HH and VV 為十六進制單位。其中 HH 須為 8 之倍數。

舉例說明：

若 HH 是 0x01，0x80，十進制 3840，

VV 是 0x01，0x70，十進制 3680，

則字型圖形資料須為 3840*3680 bits，或 480*3680 bytes。同時圖形資料之傳送順序為自下而上，自左而右。本功能為未來提供給 EIS 大型印表機使用之指令，尚須進行實際上應用程式的架接測試。

## 6.2 研究結論

由第 4 章研究發現與第 5 章研究實證中所得結果，本計劃歸納結論如下，作後續研究之參考。

(1) 現行電子病歷交換檔案雖已由 EEC 集中管理，但是中文特殊字型缺漏仍就佔了一部分比例，此部分在各自醫院交換後，可能會發生張冠李戴的情形，雖有身分證字號以及出生年月日比對，但仍會發生交付民眾後，因資料的錯誤，而產生民眾抱怨問題，因此政府之 EEC 平台的字型整合計畫，須越早進行越好，對電子病歷交換正確率有極大幫助。

(2) 現行當櫃台人員遇見特殊中文字型時，處理程序應該先在標準字庫區找尋，其次到全字庫中找尋，且運用全字庫可以避免不統一的問題；最後真的標準字庫和全字庫均找不到時，才於公告造字區間進行造字。如此可以將須造字的機會降到 0.05%以下，同時可以避免字碼不統一的問題；亦即。

(3) 解決中文特殊造字的方法若不盡速統一，將影響政府資料交換之美意。由本計畫執行過程中發現，若在不統一各醫療院所之造字，在交換後錯誤的比例相當高；另有一種方式，就是未來將病患之個資部分展現時，不讀取 EEC 平台或是各醫院之字碼，而是採用戶役政已經完成整理之標準字碼來處理，則亦可以解決此項問題。

(4) 本研究所檢索之特殊中文字比對表，可以作為各家醫院以及 EEC 平台處

理中文特殊字型時比對參考。研究中透過檢索以及程式設計得到的對照表有如下清單，詳細內容請參閱附錄 A~附錄 E。

a. 附錄 A 細明體對照 Unicode 字碼表(一) 1~9,000 字

**說明：**本檔案提供了若電子病歷主機採用 Unicode 編碼，則新細明體字之字面與 Unicode 之對照，因本研究探索前 18,000 字，因此檔案相當大，將之切開為兩個檔案包含 1~9,000 字為一個檔；9,001~18,000 字為另一檔案。

**用法：**匯入 Excel 檔案後，可以快速輸入 Unicode 就可以快速看到字面，查詢是否有特殊字使用。

b. 附錄 B 細明體對照 Unicode 字碼表(二) 9,001~18,000 字

**說明：**本檔案為附錄 A 之 9,001 字後提供了若電子病歷主機採用 Unicode 編碼，則新細明體字之字面與 Unicode 之對照，因本研究探索前 18,000 字，因此檔案相當大，本檔區間為 9,001~18,000 字。

**用法：**匯入 Excel 檔案後，可以快速輸入 Unicode 就可以快速看到 9,000 字以後之字面，查詢是否有特殊字使用。

c. 附錄 C 新細明體對照 BIG-5 與 Unicode 字碼表(一) 1~9000 字

**說明：**本檔案為本研究花費最多心力之對照表，主要將 1~9000 字電子病歷主機採用 Unicode 編碼與使用者端電腦使用之 BIG-5 碼進行對照，其中則包含新細明體字之字面與標楷體兩種字型之

BIG-5 與 Unicode 之對照，因本研究探索前 18,000 字，因此檔案相當大，本檔區間為 1~9000 字。

**用法：**本檔案可以作為後續研究者探索或開發程式，驗證正確性時使用，對於後續貢獻相當大。

d. 附錄 D 新細明體對照 BIG-5 與 Unicode 字碼表(二) 9,001~18,000 字

**說明：**本檔案為本研究花費最多心力之對照表，主要將 1~9000 字電子病歷主機採用 Unicode 編碼與使用者端電腦使用之 BIG-5 碼進行對照，其中包含則新細明體字之字面與標楷體兩種字型之 BIG-5 與 Unicode 之對照，因本研究探索前 18,000 字，因此檔案相當大，本檔區間為 9,001~18,000 字。

**用法：**本檔案可以作為後續研究者探索或開發程式，驗證正確性時使用，對於後續貢獻相當大，此為第二個檔案。

e. 附錄 E 自造字區細明體對照 Unicode 字碼表(6,500 特殊字)

**說明：**本檔案為使用 Unicode 後若超過正常字碼範圍之特殊字，本研究將之按現行中文推行委員會上公告之特殊字區間找出後進行檢索排序，此檔案可以應付現行醫療作業系統中 90~95%以上之特殊造字，本表可以提供後續研究或是查詢特殊字現行已正式編碼參考。

**用法：**若導入電子病歷之醫院未購買全字庫，則可以考量使用此表作為簡易之替代方案，此 6,500 字特殊字為比較輕量特殊字型庫。

## 6.3 研究限制

本研究是設定在使用大型主機，且使用衛生署共用醫療資訊系統之醫院，其環境為使用 UNIX I64 位元下，使用 JIS、BIG-5 或是 Unicode 等中文字碼編碼主機系統；對於其他系統包含 MAC 環境或是手持式裝置之環境，因為可載入字碼限制，可能會衍生其他問題，此為本研究之一大限制。

另，許多本研究所開發之程式碼，通常均是以 BIG-5 與 Unicode 作為對照轉換的基礎，若醫療院所非採用上述中文內碼，則可能會有轉換率不夠高的問題，此又為本研究之另一項限制。

# 附錄

**附錄 A 細明體對照 Unicode 字碼表(一) 1-9000**

**附錄 B 細明體對照 Unicode 字碼表(二) 9001-18000**

**附錄 C 新細明體對照 BIG-5 與 Unicode 字碼表(一) 1~9000 字**

**附錄 D 新細明體對照 BIG-5 與 Unicode 字碼表(二) 9,001-18,000 字**

**附錄 E 自造字區細明體對照 Unicode 字碼表(6,500 特殊字)**

**附錄 F 本計劃程式碼以及 API 引用光碟**

# 參考書目


[1] 莊德明 (2007)，漢字數位化的困境及因應：談如何建立漢字構形資料庫，第五屆兩岸三院資訊交流與數位資源共享研討會，台北。

[2] 全字庫網站(2007)，抓取日期 2011/07/26，網址： http://www.cns11643.gov.tw

[3] 文鼎公司網站資料 (2007)，2011/07/22，網址資料：http://www.openfoundry.org/index.php?option=com_content&Itemid=1&id=632&lang=en&task=view

[4] 謝育平，吳政泓，項潔，可攜式字集資源-用以解決缺字問題，台灣大學資訊工程研究所，國科會計畫 NSC 92-2811-E-002-027 及 NSC 92-2213-E-002-005 。

[5] Kun-Yuan Tsai, Jian-Xiang Liu, Chien-Tsai Liu (2009), Development and Prospect of Electronic Medical Records in China, *Journal of Healthcare Quality*, 3(6), 4-14.

[6] Tri-Service General Hospital (2010), Introduction to Electronic Medical Records, Website: http://wwwu.tsgh.ndmctsgh.edu.tw/proj/page1/page1.html Online retrieval date: 3 March, 2012.

[7] Clyde, S. (2010), Data Capture in Electronic Medical Records. *The Journal of Taiwan Association for Medical Informatics*, 19(2), 79-85.

[8] Hsin-Ginn Hwang, Cho-Hsun Lu, Ju-Ling Hsiao, Rai-Fu Chen (2009), Factors Influencing Benefits of Electronic Medical Records Exchange: Physician Perspectives, *Journal of e-business*, 11(1), 95-118.

[9] Hamdan O. Alanazi, et al., (2010), securing electronic medical records transmissions over unsecured communications: An overview for better medical governance, *Journal of Medicinal Plants Research, Academic Journals*, Vol. 4(19), 2059-2074, ISSN 1996-0875.

[10] Hsin-Tsai Wen, Wen-Shan Chien, You-Chuan Li, Hsiao-Ju Chen (2009), Terminology Standard of Interoperable Electronic Health Record-The Status Quo and Development of Systematic Nomenclature of Medicine-Clinical Term, *Journal of Medical Record Management*, 9(1), 56-67.

[11] Guang-Ming Guo, Ming-Jian Hung, Ru-Ling Xiao, (2004). Exchange of Electronic Medical Records the Key Factor, *MIST*.

[12] Ean-Wen Huang, Der-Ming Liou (2004), A Study of the System Architecture of HL7 Query Messages for Medical Record Exchanges, *The Journal of Health Science*, 6(3), 207-218.

[13] Po-Hsun Cheng, Jin-Shin Lai, Sao-Jie Chen (2004), Exploring the Information Exchange Standards for Electronic Health Records, *Journal of the Formosan Medical Association*, 8(6), 803-806.

[14] Department of Health, Executive Yuan (2012), Production and management methods of electronic medical records to medical institutions, Website: http://www.doh.gov.tw/CHT2006/index_populace.aspx. Online retrieval date: 3 March, 2012.

[15] Charng-er Shyu, Yu-pei Chien (2010). Development of Electronic Medical Records, *Regulations and Policies*, 9(2), 1-18.

[16] In Chang, (2010), Stakeholder Perspectives on Electronic Health Record Adoption in Taiwan. *Asia Pacific Management Review*, 19(1), 133-145.

[17] Liang-Wen Wang, Li-Li Wen, Chien-Tsai Liu (2009). An Empirical Approach to Adoption of LOINC-from Laboratory Department Point of View. *The Journal of Taiwan Association for Medical Informatics*, 18(2), 15-28.

[18] Wei Chen, Hsien-Liang Lin, Hsiao-Hsien Rau, Huai-Ko Chen (2009), Embedding Stream Media in Accessing Medical Imaging for the EMR System. *The Journal of Health Science*, 12(4), 255-264.

[19] Gobi M, Vivekanandan K (2009). A New Digital Envelope Approach for Secure Electronic Medical Records, *IJCSNS*, 9(1): 1.

[20] Ferreira A, Cruz-Correia R, Antunes L, Palhares E, Marques P, Costa P, Costa-Pereira A (2004). Integrity for electronic patient record reports, *IEEE Computer Society*.

[21] Bos L, Laxminarayan S,Marsh, Medical and care compunetics 1, *Ios Pr* Inc, (2004).

[22] Brandner R, Van der Haak M (2002). Electronic Signature of Medical Docu- ments: Integration and Evaluation of a Public Key Infrastructure in Hospitals. Methods of Information in Medicine-Methodik der Information in *der Medizin*, 41(4): 321-330.

[23] McGuire AL, Fisher R (2008). "Confidentiality, privacy, and security of genetic and genomic test information in electronic health records: points to consider." *Genet. Med.*, 10(7): 495.

[24] Janbandhu PK, Siyal MY (2001), Novel biometric digital signatures for Internet-based applications, *Info. Manage. Compu. Secur*, 9(5): 205-212.

[25] Anderson JG., Security of the distributed electronic patient record: a case-based approach to identifying policy issues, *Inter. J. Med. Infor.*, 60(2): 111-118, 2000.

[26] O'Brien DG, Yasnoff WA (1999), Privacy, confidentiality, and security in information systems of state health agencies, *Am. J. Prev. Med.*, 16(4): 351-358.

[27] De Meyer F, Lundgren PA (1998). "Determination of user requirements for the secure communication of electronic medical record information." *Inter. J. Med. Inform.*, 49(1): 125-130.

[28] Epstein MA, Pasieka MS (1998). Security for the digital information age of me-dicine: issues, applications, and implementation. *J. Digital. Imaging*. 11(1): 33-44.

[29] Rind DM, Kohane IS, Szolovits P, Safran C,Chueh HC,Barnett G. (1997). "Maintaining the confidentiality of medical records shared over the Internet and the World Wide Web." *Ann. Intern. Med.* 127(2): 138.

[30] 許壽椿(1999)，網絡時代的漢字全面解決方案和漢字本體研究，99 漢字應用與傳播國際學術研討會，北京。

[31] 新北市立聯合醫院(2010)，電子病歷推動專區宣導資料，網頁資料：http://netreg.tpch.gov.tw/EMR/%E7%B6%B2%E9%A0%81.htm抓取日期：2011.07.26。

[32] 法務部(2000)，個人資料保護法，全國法規資料庫，抓取日期：2000.04.17，網頁資料：http://law.moj.gov.tw/LawClass/LawContent.aspx?pcode=I0050021

[33] 行政院衛生署(2009)，醫療機構電子病歷製作及管理辦法，衛署醫字第 0980261732 號令修正發布，2009.08.11。

[34] 中文推行委員會(2009)，中文圖形字型對照API引用標準，網站資料：http://www.cns11643.gov.tw/AIDB/download.do?name=%E4%B8%AD%E6%96%87%E5%85%B1%E9%80%9A%E5%B9%B3%E5%8F%B0，抓取日期：2011/07/26