Commerce & Management Quarterly, Vol. 2, No.1, pp. 45~ 65 (2001)

語意處理之中文自然語言擷取介面—網際網路平台

CHINESE NATURAL LANGUAGE SEMANTIC PROCESSING IN DATABASE RETRIEVAL - INTERNET PLATFORM

蔡燿全
成功大學企業管理系
楊棠堯
崑山科技大學會計系

Yao-Chuan Tsai
Department of Business Administration,
National Cheng Kung University

Tarng-Yao Yang
Department of Accounting,
Kun-Shan University of Technology

摘　要

傳統結構化查詢語言(SQL)或範例學習(QBE)的查詢方式，使用者不僅需要事先學習或訓練，而且還必須瞭解資料結構及資料間的關聯，造成查詢上的負擔。然而自然語言的擷取方式，在上述的問題均具有令人青睞的優越性，除了以口語方式查詢的友善人機介面外，使用者不須經過任何訓練，即可進行臨時性的(Ad-hoc)查詢。本研究基於上述理由與本土化需求，提出一個以語意處理為主的中文自然語言資料庫擷取介面，透過語意轉換樹文法，將中文自然語言轉換成國際標準的結構化查詢語言(SQL)，達到查詢資料庫的目的。網際網路(Internet)的發展已成為國際趨勢，其亦是現今資訊流通最方便、快速的方式，為了達到查詢地點透明化的目的，本研究擬將上述中文自然語言介面移植到網際網路的平台，藉著其主從架構的優點，以提高本研究的應用價值。由於處理語意轉換樹文法的遞迴需求，本研究以 CGI 後端技術來發展成績資料庫的擷取介面雛形，以驗證本研究之系統架構。

關鍵詞：資料庫擷取介面、中文自然語言處理、語意轉換樹文法

ABSTRACT

Typical query methods of SQL or QBE need users to learn and train in advance resulted in the burden in database retrieval. Whereas natural language query interfaces offer the overwhelming advantage of requiring the least end-user training, and the capability of proceeding to ad-hoc queries. In order to fit in with the demand of localism, we intend to use Chinese natural language based on semantic processing as the database retrieval interface. The core processing of this system is the semantic transition tree grammars. We also try to transplant Chinese natural language retrieval interface into Internet environment. By way of the Client/Server characteristics of internet, information retrieval will be much personalized and efficient. Eventually we choose the CGI technology that can process recursive problem to develop the prototype of Grade database retrieval interface in order to demonstrate the practicability of this study architecture.

***Key words*：** Database retrieval interface, Chinese natural language processing, Semantic transition tree grammar

壹、緒論

資料擷取研究至今已提出多種解決方式，目的均希望能提供更友善的人機介面。傳統結構化查詢語言(SQL)(Astrahan et. al., 1976; Chamberlin & Boyce, 1974)或範例學習(QBE)(Zloof, 1975)的查詢方式，使用者不僅需要事先學習或訓練，而且還必須瞭解資料結構及資料間的關聯（亦即所謂查詢路徑，Navigation path），造成查詢上的負擔。然而自然語言的擷取方式，在上述的問題均具有令人青睞的優越性，使用者以接近口語方式查詢，是最友善的人機介面，同時也可以進行臨時性(Ad-hoc)查詢(Hancock & Chignell, 1989; Allen, 1987)。

本研究基於上述理由與本土化需求，提出一個以語意處理為主的中文自然語言資料庫擷取介面，透過語意轉換樹文法(Winston, 1992)，將中文自然語言轉換成國際標準的結構化查詢語言(SQL)，達到查詢資料庫的目的。但由於中文自然語言處理的斷字與剖析器等技術尚有瓶頸存在，所以本研究以語意轉換樹文法的特性為基礎，採用語意剖析的方式來處理中文自然語言，將限定格式的中文自然語言以動詞來分類，此種方式的優點是可以直接處理中文自然語言的斷字問題。另外因語意轉換樹文法是屬於 Domain dependent，亦即不同的資料庫會有其相對的文法，故本研究將語意轉換樹文法從系統中獨立成一個文法庫，以利系統移植性(Transparent)的後續研究。

網際網路(Internet)的發展已成為國際趨勢，由於其採開放式架構，只要遵循 TCP/IP 通訊協定，便可以連接網際網路，也因此網際網路已是現今資訊流通最方便、快速的方式，而網際網路資料庫(Web database)的研究也如雨後春筍般的湧現(Khurana & Gadhok, 1997; Mohseni, 1996)。為了達到查詢地點透明化的目的，本研究擬將上述中文自然語言介面移植到網際網路的平台，藉著其主從架構的優點，讓使用者能在連上網際網路的任何地方，都可以查詢到所需的資訊，這更提高了本研究的應用價值。也由於本介面是發展於網際網路環境，所以若使用傳統剖析自然語言的技術（包括 Lexicon 及 Parser），可能會因複雜的剖析過程，而造成極長的反應時間。而本研究乃使用已定義的語意轉換樹文法來處理中文自然語言的資訊需求，便可改善上述反應時間的問題。

本文後續章節的描述如下。第二章介紹學者在本文相關技術的研究，包括資料庫介面、其他自然語言介面、網際網路資料庫及語意轉換樹文法等的探討。其中並針對領域獨立等分類，彙整說明相關的自然語言介面，同時也敘述了 Elliptical Query 的處理。第三章描述本研究的研究方法，包括研究限制、系統雛形的資料庫應用領域，接著詳細說明本研究的系統架構，包括各模組的運作及文法庫的內容。第四章主要是雛形系統的示範，首先說明限定語句中分類的原則，接著使用 CGI 後端技術發展的雛形系統來例證完全查詢與不完全查詢的實作，最後說明進行測試的結果。最後一章根據本研究的貢獻及建議，加以陳述結論。

貳、相關研究

本章針對本研究所應用的技術及相關之文獻，分成四部分進行更深入的介紹。第一部份首先探討文獻上已發表的各種資料庫介面之相關研究，說明自然語言與其它資料庫介面的優劣；第二部分接著探討文獻上已發表的英文自然語言介面系統，並整合性地介紹各系統間的比較。在本研究所應用的技術方面，第三部份及第四部份則分別更進一步地介紹網際網路資料庫與語意轉換樹文法，以作為本系統技術發展上的支持。

一、資料庫介面

資料庫應用在絕大部分資訊系統中是存在的，而對於資訊型態的資訊系統而言，友善的資料庫查詢介面更是系統成敗的重要關鍵。所以查詢介面的研究在資料庫領域中已成為學者青睞的一個課題，目前已提出的解決方式，如下整理所述：

1. 利用容易學習的正規查詢語言，例如：SQL(Astrahan et. al., 1976; Chamberlin & Boyce, 1974)、VERDI(Wald, 1985)等。
2. 利用畫面選擇或圖表形式的介面，例如：CUPID(McDonald & Stonebraker, 1975)、SM 介面(蔡燿全等，1999)等。
3. 利用範例的產生，例如：QBE(Zloof, 1975)等。
4. 利用查詢說明的對話方式，例如：RENDEZVOUS(Codd, 1977)等。

圖一 SM 自然語言型式介面(蔡燿全等，1999)

5.利用自然語言查詢方式，例如：LIFER (Hendrix et. al., 1978)、PLANES(Waltz, 1978)、EUFID(Templeton & Burger, 1986)、LUNAR (Woods, 1978)、SESAME (Sabbagh, 1991)、System X (McFetridge et. al., 1988)、TEAM (Grosz et. al. 1987)、KID (Ishikawa et. al. 1987)、Janas86 (Janas, 1986)等。

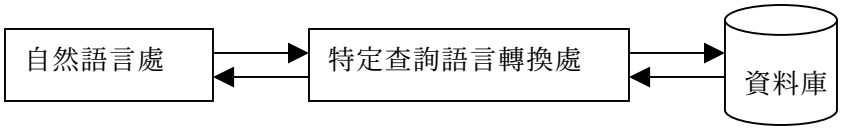
Huang 及 Chen(Huang & Chen, 1989)在其研究中，對於傳統常用的查詢語言有下列的分析與探討：

1.SQL 的語法有格式上的拘束，用字、語意上的考慮亦不盡周詳，而且必須事先學習查詢語法，這些很難讓使用者滿意。

2.QBE 雖易於使用與學習，使用者還是得熟悉資料欄位的名稱和資料的內容，在大型的資料庫系統上使用，也不太方便。另一方面，也需要使用者經常反覆的練習，才能熟練。

而容易讓使用者接受的圖形介面方式（如 ER 介面）亦不需使用者事先學習特定的語法，且使用者只要透過圖形介面便可瞭解資料庫的結構，透過欄位的點選即可找到需要的資料，同時所有欄位的自然合併也可達到大部分的臨時性(Ad-hoc)查詢。雖然圖形介面的查詢方式有其優點，但圖形的抽象化表達方式亦可能讓使用者無法瞭解其中含意，仍然需要反覆練習，且此種方式亦無法提供對於欄位的計算資料查詢。

SM 介面(蔡燿全等，1999)是支援 Structured Modeling 模式化方法的資料



圖二　基本自然語言查詢介面模式（本研究）

庫擷取功能。Structured Modeling 是數量模式表示、管理的方式之一，在模式求解的過程中，其資料庫是由系統根據模式綱目(Model schema)自動產生，且會相對地自動產生擷取資料庫的 Schema -Based Query Interface，即為 SM 介面，而這個查詢介面則是包含資料庫欄位的簡易自然語言型式，如圖一所示。

SM 介面與 QBE 的查詢方式類似，不同的是，前者的資料庫欄位是以自然語言的方式呈現。SM 介面能自動產生的最大優點是可以達到領域獨立(Domain independent)，能夠輕易地移植到不同應用領域的資料庫；且使用者不需經過額外訓練，同時具有查詢多樣化的特性。但相對地亦有一些缺點，其產生出來的自然語言介面只能透過欄位來查詢，雖然可以處理大部分的查詢，但對於欄位聚合運算(Aggregate function)則無法處理；且對於資料新增、刪除、更新等類型的查詢處理，亦無法支援。

二、自然語言介面

在上面的數種資料庫查詢介面方式中，自然語言查詢介面在使用者訓練方面，幾乎有著壓倒性的優點，使用者以口語的語言來查詢資料，似乎最為使用者接受，而且對於複雜的查詢，若使用特定的查詢語言（如 SQL），也有可能造成使用者不會下查詢指令，而查不到資料的窘境，所以在自然語言介面的研究，更是前仆後繼(Hendrix et. al., 1978; Waltz, 1978; Grosz et. al., 1987; Templeton & Burger, 1986; Woods, 1978; Sabbagh, 1991; McFetridge et. al., 1988; Ishikawa et. al., 1987; Janas, 1986; Nabil et. al., 1994)。

Nabil 等 (Nabil et. al., 1994) 及 Kok(1995)均在其研究中即指出，自然語言介面在使用者友善方面是最引人注目的。因為自然語言介面有下列兩項優點：一是在眾多查詢方式中，自然語言介面的使用者僅需最少的訓練即可操作系統；另一項優點則是自然語言介面可以支援不同類型的使用者需求，現在提出的研究均只針對查詢類型的處理，但自然語言介面除了查詢外，尚可支援新增、刪除、修改等不同的需求類型。

大部分的自然語言介面研究，無論使用語法或語意的內部邏輯表示，均是將自然語言的內部表示轉換成資料庫管理系統特定支援的查詢語言格式，例如結構化查詢語言(SQL)，其一般的模式如圖二所示：

EUFID(Templeton & Burger, 1986) 使用語意文法來處理使用者查詢，其分成三個模組：解析器(Analyzer)、對映器(Mapper)及轉換器(Translator)。解析器以

英文自然語言為輸入，產生一個由語意文法與字典組成的剖析樹(Parse Tree)。對映器則將剖析樹轉換成內部邏輯語言，最後轉換器便將內部邏輯語言解譯成 DBMS 可支援的查詢語言。EUFID 藉著兩階段的轉換處理可以提高在不同 DBMS 的移植能力，但相對地，也會因額外的處理而降低系統的效率。

LUNAR(Woods, 1978)本身即具有一龐大的英文自然語言文法庫，同時它亦有一個以擴增轉換網路(ATN)為基礎的剖析器，還有一個語意轉換器(Semantic interpretor)可將語法結構的查詢語言轉換成語意表示格式。在這個系統中，使用者的自然語言查詢會被解譯成 Meaning Representation Language，此種語言是以第一階邏輯述句(First-order predicate calculus)表示。而在處理任何語意解譯的動作之前，LUNAR 會執行完整的語法分析工作，所以這種語法分析與語意分析完全分開的處理，可能會影響系統執行的績效。

PLANES(Waltz, 1978)是一個航空定位資料庫的前端自然語言查詢介面，其自然語言的處理分成三個階段：剖析處理 (Parsing) 、查詢產生 (Query generation)、及結果反應(Response)。在剖析處理階段，如同其他研究一般，乃將使用者輸入的自然語言轉換成中介的查詢格式，不同的是，此系統提出一個 context register 的機制，此機制可用來解決模糊或不完全查詢的情況。而在查詢產生階段則將中介的查詢格式轉換成正規的查詢語言，這包括確認出關連、屬性、運算子及輸出型式。此系統最後的輸出模組則決定輸出型式，查詢結果的輸出可以是表格型式，也可以是圖表型式，甚至可以直接輸出到印表機。

SESAME(Sabbagh, 1991)是先將自然語言轉換成 ER 介面的查詢語言(ER-SQL)，然後再轉換成結構化查詢語言(SQL)。使用者可以自由地輸入自然語言查詢，也可以使用一些功能提示來引導建構查詢。SESAME 是使用語法分析來處理使用者的查詢，且它並不支援 Elliptical queries 的處理。

System X(McFetridge et. al., 1988)首先將使用者的自然語言查詢，根據語法分析得到剖析樹(Parse tree)，然後再將剖析樹轉換成具有資料庫表格或欄位的語意表達，接著把語意表達轉換成內部的邏輯形式，最後再轉換成 SQL。System X 和 LUNAR 一樣，是分開剖析語法規則與語意分析，如此通常會加重系統的複雜性，且 System X 亦不支援 Elliptical queries 的處理。

TEAM(Grosz et. al., 1987)是一個可以移植到新資料庫的自然語言查詢介面系統，它具有取得狀態(Acquisition mode)與查詢狀態(Question-answering)兩種功能。在取得狀態下，資料庫管理者必須提供資料庫結構、及領域的資訊；而在查詢狀態，系統由兩個模組組成：對映器及轉換器。對映器將使用者的自然語言對映到一個邏輯形式，而轉換器則將邏輯形式轉換成資料庫查詢語言。

所謂模糊或不完全查詢(Elliptical queries)即是表示使用者所輸入的查詢是資訊不完全的，這種問題是經常出現在自然語言介面的處理中。使用者基本上是希望系統能從前後文的對話中瞭解這

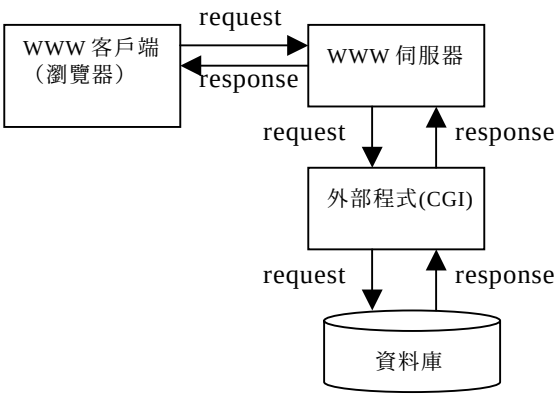
表一　自然語言介面相關研究整理（本研究）

系統	領域獨立	不完全查詢	查詢類型	文法	語言類型
EUFID (Templeton & Burger, 1986)	是	否	擷取	語意	英文
KID (Ishikawa et. al., 1987)	是	未確定	擷取	語法&語意	英文
LUNAR (Woods, 1978)	否	是	擷取	語法	英文
LIFER (Hendrix et. al., 1978)	否	是	擷取	語意	英文
PLANES (Waltz 1978)	否	是	擷取	語意	英文
SESAME (Sabbagh, 1991)	否	否	擷取	語法	英文
System X (McFetridge et. al., 1988)	是	否	擷取	語法&語意	英文
TEAM (Grosz et. al., 1987)	是	是	擷取	語法	英文
Janas86 (Janas, 1986)	否	是	擷取	語意	英文
Nabil (Nabil et. al., 1994)	否	是	擷取	語法&語意	英文

種不完全查詢。一些自然語言介面(Templeton & Burger, 1986; Sabbagh, 1991; McFetridge et. al., 1988)是完全不支援 Elliptical queries，而另外一些(Hendrix et. al., 1978; Waltz, 1978; Grosz et. al., 1987 ; Woods, 1978; Janas, 1986)則是不同程度的支援 Elliptical queries 處理。例如，LUNAR(Woods, 1978)可以處理”its”等代名詞的參考，假設使用者輸入第一個查詢是 ”What is the silicon content of each volcanic sample?”，接著第二個查詢是 “What is *its* magnesium concentration?”，則第二個查詢的”its”便會被名詞” each volcanic sample”所取代，亦即第二個查詢會變成 “What is the magnesium concentration of each volcanic sample?”。而在 LIFER(Hendrix et. al., 1978)系統中，其 Elliptical queries 的處理是採取簡單地與先前查詢做字串比對與取代，假設第一個查詢是 “What is the length of Santa Inez?”，而緊接著第二個查詢是 “of the Kennedy”，與第一個查詢比對的結果，“of the Kennedy”取代了”of Santa Inez”，所以第二個查詢變成 “What is the length of the Kennedy?”。

本研究將文獻中有關自然語言查詢介面研究整理如表一。在第一欄列出了文獻中提出的自然語言介面系統。第二欄的領域獨立表示是否可以移植到新的應用領域，而不需額外的修改介面，當領域獨立時，則表示該系統有支援移植不同領域的能力，而不是領域獨立時，則表示該系統只能適用於某一特定領域。第三欄則顯示是否有支援不完全查詢(Elliptical queries)的能力，如前所述，

由於不完全查詢的情況甚多，所以一般均只支援部分的不完全查詢。第四欄顯示該系統所提供的查詢類型，大部分均只支援資料擷取，因資料更新必須考慮資料庫完整限制，所以一般均未支援。第五欄表示系統用來處理自然語言的文法類型，大部分均會分析自然語言的語法結構，但還是有些系統只使用語意分析來處理自然語言。最後一欄則顯示先前的研究幾乎只出現在國外，而國內似乎尚未有這方面的研究，這也是本研究強烈的動機之一。



圖三　基本的網際網路資料庫模式
(Mohseni, 1996)

三、網際網路資料庫

在現今資訊爆炸的時代，面對龐大複雜的資料與資訊，藉著資料庫的輔助，便可輕易地解決資料收集、處理及應用的問題。不過，雖然資料庫技術已發展良久，但有一個技術核心一直無法突破，那就是各種資料庫之間沒有共通的標準，舉凡語法、資料結構，到應用介面等都是。但是，網際網路的出現，為資料庫這個長久無法突破的技術核心，帶來一絲曙光，網際網路的弗界，標準齊一，再加上資料庫處理資料的強大能力，將會為人類的智慧累積、資訊分享帶來革命性的影響。

大多數開發使用者介面的沒有考慮到跨平台的問題，所開發的軟體只能在某特定作業平台上使用，隨著 Web 的出現，軟體無法跨平台使用的窘境，獲得了解決，因為大部分瀏覽器都支援跨平台使用。因此無論我們選擇那一種語言做為資料庫介面，Web 資料庫將可以跨平台使用，而無需經過重新編碼。跨平台使用的方便性，使得網際網路資料庫的應用更具價值，而且維護容易，成本低廉。

使用 Web 資料庫的另一好處，是使用介面的一致性，即使在不同平台上，藉由瀏覽器的幫助，使用者介面的一致性很高。使用者在不同電腦或不同作業平台，使用該資料庫時不必重新學習，同時程式設計者也不必花費太多時間撰寫使用說明，因此在成本節省上有很大助益。

一般網際網路資料庫的架構模式如圖三所示。隨著技術日新月異，在 WWW 伺服器與資料庫之間的連結，已有更視覺化的軟體問世，諸如 IntraBuilder、PowerBuilder 等，其物件化的設計環境，將資料庫的連結隱藏在物件當中，更簡化了開發系統的時間 (Mahar & Henderson, 1997)。但由於本研究處理中文自然語言的遞迴需求，在資料庫連結的部分，仍以 CGI 方式透過 ODBC 驅動程式來存取資料庫。

四、語意轉換樹文法

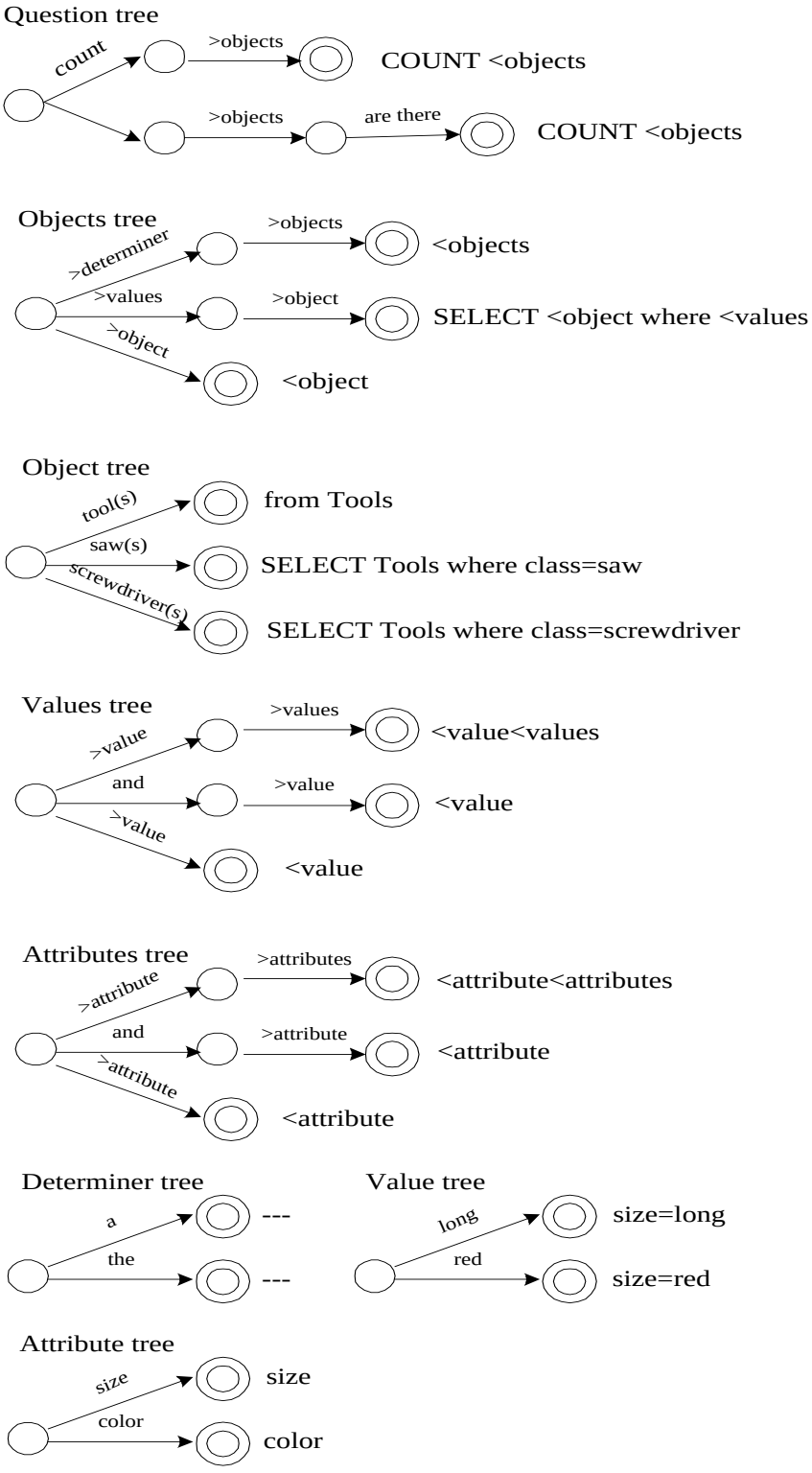
語意轉換樹文法是由擴充轉換網路(ATN)演變而來(Winston, 1992)，與 ATN 不同的是，它是樹狀的結構，而不是網路的結構，原因在於可以清楚地表示出路徑。因為資料庫屬於特定領域，且牽涉到語意的問題，所以選用此種包含語意的文法來做為自然語言的內部表示方法。

圖四顯示了語意轉換樹文法的一個例子(Winston, 1992)，如其名之意，此文法是以語言的語意字彙為基本元素，而使用樹(Tree)結構來組成整個文法。整個文法有一個主樹，其它均為子樹。使用者需求是採用動詞來分類，例如 List、Give、Identify、Count、How many 等，每個動詞在主樹中便是一個語意分支，最後會產生其相對應的查詢語言。每棵樹有許多分支，每一分支代表一種文法，只要成功地追蹤某一分支，立即停止追蹤此樹，而傳回追蹤的結果，若為主樹則傳回最後的 SQL，若為子樹則繼續其它子樹；但如果整棵樹的所有分支均追蹤失敗，則停止所有追蹤，並宣告無法識別使用者的自然語言輸入。

圖四中，每條分支的箭號代表語意文法追溯，若是箭號旁邊附屬語意字彙（如圖四的 Count），表示輸入的自然語言中必須有此語意字彙才算成功追蹤，本研究的中文斷字即是利用此特性來處理；但若是箭號旁邊附屬子樹變數（如圖四的 >objects），則表示必須再追蹤名為 objects 的子樹。每條分支的單圓圈表示非終端節點，亦即其後還有路徑可以追蹤，而雙圓圈則表示終端節點，亦即已經成功地追蹤完成此分支，並亦代表追蹤此子樹成功。

樹中每個分支的終端節點各有一個存值的樣本(Pattern)，而樣本又可分成兩種不同的型式，一種是模版樣本(Template-like pattern)，又稱為樹變數(Tree variable)，它置放於終端節點，以”<”符號表示，用來儲存追蹤完一個子樹後的結果（例如圖四的<objects），與”>”符號不同的是，後者代表追蹤另一個子樹。而另一種樣本則是自由變數樣本(Variable-free pattern)，它也是置放於終端節點，其所存放的是常數值，（例如圖四的 size、color 等）。綜而言之，當每次一個子樹被追蹤成功後，子樹的名字便成為一個變數，其儲存了所追蹤成功路徑終點的樣本值。

現根據圖四的文法舉例說明，假設使用者輸入”count the long screwdrivers”的限定自然語言。首先走 Question tree，第一個字是”count”，所以選擇子樹的第一條路徑，根據>objects，所以再接著追蹤 Objects tree。先選擇子樹的第一條路徑，追蹤 Determiner tree，Determiner tree 的第二條路徑符合第二個字”the”，於是 Determiner tree 的第二條路徑追蹤成功，但因其終端節點的樣本沒有存值，所以模版樣本<determiner 沒有存值而返回中斷點。接著在 Objects tree 中的第一條路徑下，繼續追蹤 Objects tree，先走 Objects tree 的第一條路徑發現是錯誤的，所以選擇第二條路徑，根據>values，因此追蹤 Values tree，同理先走 Values tree 的第一條路徑{*}，發現要追蹤 Value tree，在 Value tree 的第一條路徑中，符合了第三個字”long”，所以模版樣本<value 儲存「size=long」後返回。回到



圖四 語意轉換樹文法(Winston, 1992)

Values tree 中斷點後，繼續再走 Values tree，但此時 Values tree 的三條路徑均不能符合第四個字”screwdrivers”，表示在{*}處所走的第一條路徑是錯誤的，此時需將模版樣本<value 的值清除，同時從第三個字”long”再重頭開始執行，也就是說，要從 Values tree 的第二條路徑重新追蹤。經過詳細的演算過程後，使用者輸入”count the long screwdrivers”的自然語言查詢要求，經語意轉換樹文法的解析後，轉換成內部的查詢語言邏輯型式 ”COUNT [SELECT [SELECT from Tools where class= screwdrivers] where size=long”，轉換成標準的結構化查詢語言後，再送至資料庫取得資料。

參、研究架構

一、研究限制

如同傳統研究，本研究所提出的架構尚未包含資料的新增、刪除及修改等更新處理，所以在本系統架構的資料庫部分有以下幾點限制：

(1) 資料庫管理系統(DBMS)只限於能夠處理標準 SQL 語法。
(2) 資料庫內的資料假設是不會更動的，亦即新增、刪除及修改等是不成立。
(3) 資料庫內只限於一個資料表(Single table)，亦即尚不處理多資料表的情況。

二、資料庫結構

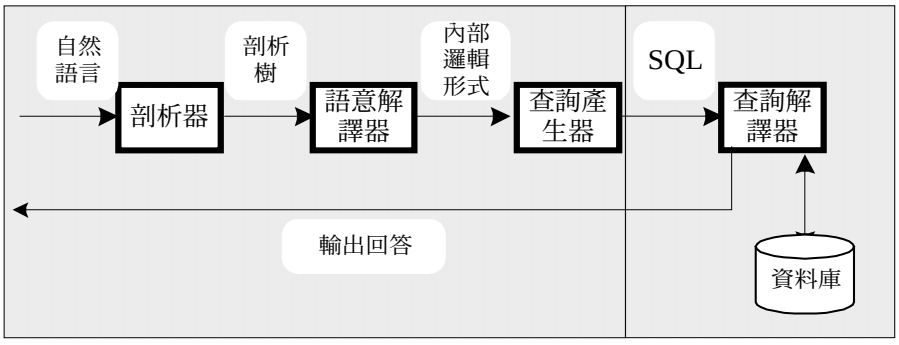
在資料庫應用領域的選取上，本研究以學生成績資料庫為例，根據上述研究限制所述，資料庫內只有一個資料表，名為「成績」資料表，其由六個欄位所組成，分別是「學年」、「學期」、「課程名稱」、「學生學號」、「學生姓名」、及「成績」。實務上，成績是由學年期、課程及學生等資訊共同參與而產生的，所以「成績」資料表的主鍵是由「學年」、「學期」、「課程名稱」、「學生學號」等欄位組合而成的複合主鍵。且因本研究範圍不包括資料的新增、刪除及修改等處理，所以各筆記錄的內容亦不會變動。資料庫的資料結構與詳細的記錄內容如表二所示。

三、系統架構

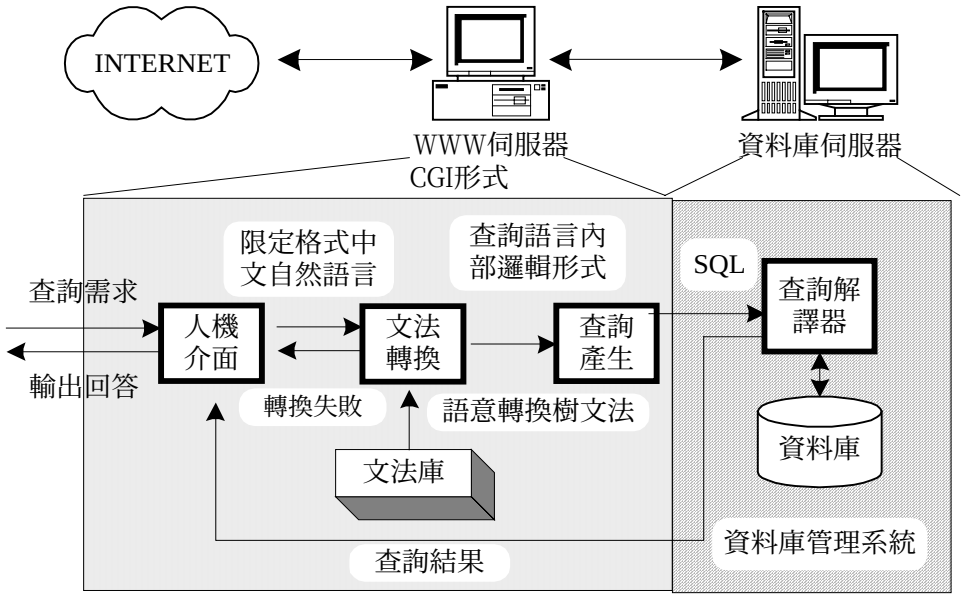
Kao 等(Kao et. al., 1988)提出了一個傳統的自然語言介面系統架構，如圖五所示。圖五中的剖析器與語意解譯器機制是自然語言的前置處理。一個完整的自然語言系統，由原始的自然語言輸入，透過剖析器進行語法分析，先判斷在文法上是否有錯誤，若無則產生相對的剖析樹(Parser tree)，其格式即是所謂的擴增轉換網路(ATN)等；接著分析語意層面，一般這是較困難的階段，因為一個字可能有多種不同的意義，且不同的句子也有可能代表相同意義，語意分析的結果通常都是以語意網路(Semantic network)的內部邏輯形式來表示(Winston, 1992)。如前文獻探討所述，傳統架構中把語法分析及語意分析分開，雖然具有領域獨立的部分優點，但對於系統執行的績效卻大打折扣，更何況於本研究的網際網路平台。

表二 本研究之學生成績資料庫結構（本研究）

學年	學期	課程名稱	學生學號	學生姓名	成績
87	1	作業系統	C870Q043	李育全	87
87	1	人工智慧	C870Q098	楊婉璇	77
87	1	人工智慧	C870Q100	張莉莉	75
87	1	人工智慧	C870Q057	楊道元	60
87	1	人工智慧	C870Q043	李育全	66
87	1	資訊管理	C870Q026	張克彰	80
87	1	資訊管理	C870Q098	楊婉璇	68
87	1	資訊管理	C870Q100	張莉莉	75
87	1	資訊管理	C870Q057	楊道元	71
87	1	資訊管理	C870Q043	李育全	79
87	1	作業系統	C870Q026	張克彰	82
87	1	作業系統	C870Q098	楊婉璇	74
87	1	作業系統	C870Q100	張莉莉	80
87	1	人工智慧	C870Q026	張克彰	11
87	1	作業系統	C870Q057	楊道元	71
87	2	作業系統	C870Q043	李育全	87
87	2	人工智慧	C870Q098	楊婉璇	90
87	2	人工智慧	C870Q100	張莉莉	78
87	2	人工智慧	C870Q057	楊道元	60
87	2	人工智慧	C870Q043	李育全	81
87	2	資訊管理	C870Q026	張克彰	80
87	2	資訊管理	C870Q098	楊婉璇	60
87	2	資訊管理	C870Q100	張莉莉	73
87	2	資訊管理	C870Q057	楊道元	55
87	2	資訊管理	C870Q043	李育全	79
87	2	作業系統	C870Q026	張克彰	60
87	2	作業系統	C870Q098	楊婉璇	64
87	2	作業系統	C870Q100	張莉莉	30
87	2	作業系統	C870Q057	楊道元	20
87	2	人工智慧	C870Q026	張克彰	89



圖五 傳統的自然語言介面系統架構(Kao et. al., 1988)



圖六　本研究的中文自然語言擷取介面系統架構（本研究）

基於績效的考量且由於中文自然語言的技術瓶頸，在析器與語意解譯器等兩部分尚未有成熟的技術公布，所以本研究如前所述乃以有限定格式的中文自然語言為輸入，以動詞為分類，將資訊需求的語意轉換樹文法事先定義在文法庫中，文法庫的詳細說明見於下一節，雖然限定格式的輸入可能造成系統可用性的降低，但本研究是以有系統的分類方式，來建構文法庫，其可為系統接受的句型已涵蓋大部分口語的型式，也彌補這方面的缺陷。而且本研究在後續的研究中，亦擬針對限定格式的缺點，提出學習方式的解決方案，應可增加系統的實用性。

由上述改良後的架構，再加入網際網路的主從架構，便成為本研究所提出的中文自然語言擷取介面的系統架構，如圖六所示。中文自然語言擷取介面的系統核心於網際網路的平台上發展，以 CGI 方式於 Web 伺服器內執行，主要將使用者的輸入轉換成 SQL，而成功處理後的 SQL 則透過 ODBC 驅動程式向 Web Database 要求查詢結果。使用者透過瀏覽器輸入查詢需求，系統的文法轉換模組會利用文法庫內已事先規劃的文法知識，將中文自然語言轉換成查詢語言的內部邏輯形式，最後再轉換成標準的結構化查詢語言(SQL)，以傳送至資料庫伺服器處理查詢。

表三 Wheres 子樹的文法知識表示（本研究）

>Cvalues >D >Attrs
>Cvalues >Rvalues >D >Attrs
>D >Cvalues >D >Attrs
……

圖六中的人機介面模組，顧名思義，即為使用者介面，在網際網路環境下，此模組存在的方式即為網頁(Homepage)的型式。為了讓使用者不須事先瞭解資料庫結構就能查詢資料，在此使用者介面中，系統亦提供文字形式的資料庫結構說明（見於下一章)。且在輸入媒介方面，中文鍵盤輸入方式亦可由語音輸入所取代, 以加速查詢的時間。

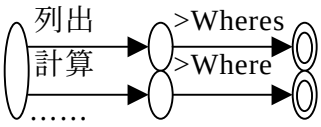
文法轉換模組便是本中文自然語言擷取介面的核心所在，它主要的工作即是將限定格式的中文自然語言轉換成 DBMS 可以支援的查詢語言(SQL)。由圖六可以看出，此模組的執行是根據文法庫內的語意轉換樹文法知識，每個子樹均為模組中的一個副程式（如圖四的 Objects tree），每個語意變數（如圖四的 Count）代表斷字，而子樹變數（如圖四的>objects）即為呼叫副程式執行，副程式執行後會傳回模版樣本（如圖四的 <objects）的值，當最後成功執行到達主樹的終端節點，便會傳回內部型式的 SQL 語法。因解譯過程中，時常會發生自我呼叫的情形，所以遞迴的處理便成為模組的主要技術，此即為本研究選擇 CGI 後端技術的主要原因。

四、文法庫

如前所述，為了移植性的後續研究，本研究將文法庫獨立於系統核心之外，以利後續研究移植到其它資料庫時，文法能自動建立。文法庫的語意轉換樹文法是必須事先建立的，如前研究限制所述，本研究以單一資料表的學校成績為雛形資料庫，其欄位組成為「學生姓名」、「課程名稱」、「學年」、「學期」、「成績」。而本研究的限定中文自然語言亦採用與 Winston(1992)相同的動詞分類，部分的語意轉換樹文法如圖七所示。由下面的文法可以看出，本介面亦可支援聚合函數（例 Average, Max 等）的運算查詢。另一項值得一提的是，當使用者所下的需求中，查詢條件有資訊不完全時（例如“ 列出資訊管理的所有成績 ”， 查詢條件中的資訊管理缺乏「課程名稱」的欄位資訊），本系統亦能解決此 Elliptical query 的問題。

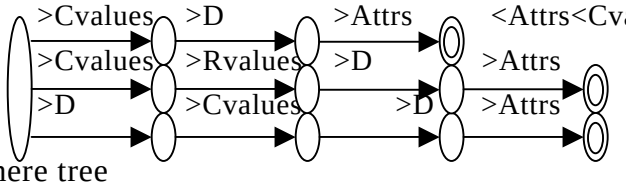
如前所述，文法庫乃獨立於系統核心之外，故本研究的文法知識表示是以文字檔(Text files)的型式存在，每棵樹均各自有其文法檔案（例如圖七的 Wheres 子樹的 wheres.txt 檔案），目的乃為後續之研究，其一是為了移植性（當移植到其它資料庫領域時，文法能自動建立），其二則是為了學習性（透過學習，能自動加強文法庫的文法知識）。文法檔案中，每一列代表子樹的一個分支，列

Question tree



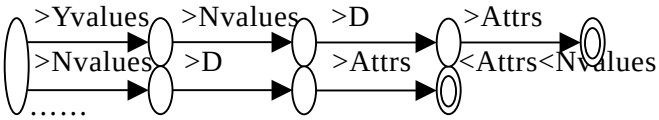
SELECT <Wheres
SELECT <Where

Wheres tree



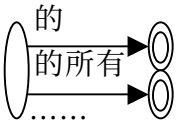
<Attrs<Cvalues
<Attrs<Cvalues<Rvalues
<Attrs<Cvalues

Where tree

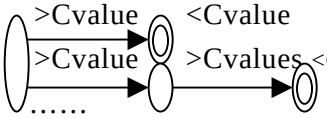


<Attrs<Yvalues AND <Nvlues
<Attrs<Nvalues

Determiner tree

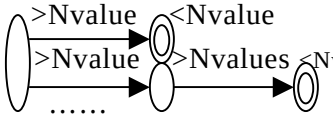


Cvalues tree



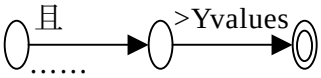
<Cvalue OR <Cvalues

Nvalues tree

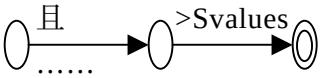


<Nvalue OR <Nvalues

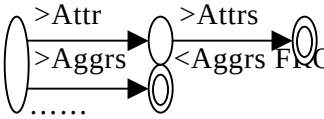
Yvalues tree



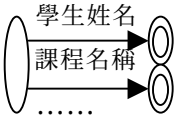
Svalues tree



Attrs tree

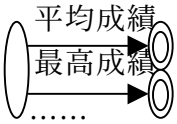


Attr tree



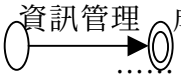
成績.學生姓名
成績.課程名稱

Aggrs tree



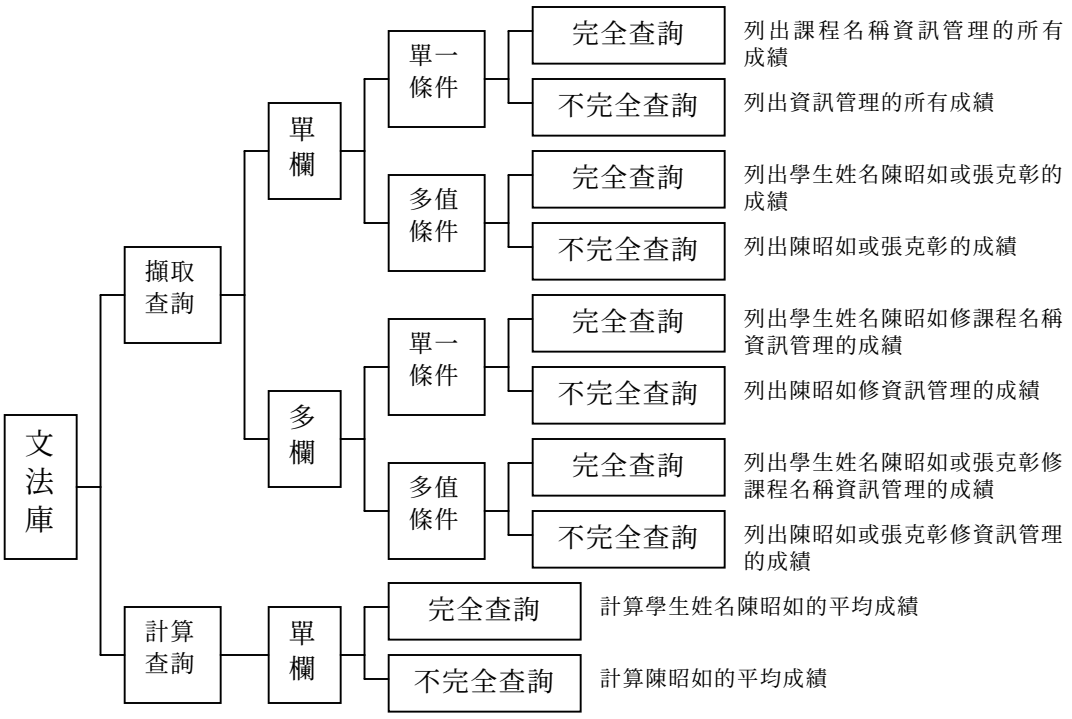
AVERAGE(成績)
MAX(成績)

Cvalue tree



成績.課程名稱='資訊管理'

圖七　學校成績資料庫的部分語意轉換樹文法（本研究）



圖八 文法庫分類及範例（本研究）

開頭即為起始節點，列結尾即為終端節點，而中間節點則以空白字元表示，以圖七的 Wheres 子樣為例，其文法檔案格式如表三所示。

如圖七的文法所示，文法庫是由主樹(Question tree)的分支來分類，主要分成「擷取查詢」與「計算查詢」兩類。文法庫的系統方式分類與各自分類的範例，如圖八所示。「擷取查詢」可區分為「單欄」與「多欄」,「單欄」與「多欄」又可以各別區分為「單一條件」及「多值條件」兩類，最後每一類又再細分成「完全查詢」與「不完全查詢」,「擷取查詢」對應 SQL 處理的格式如下，因研究限制為單一表格，所以 FROM 部分可以省略。

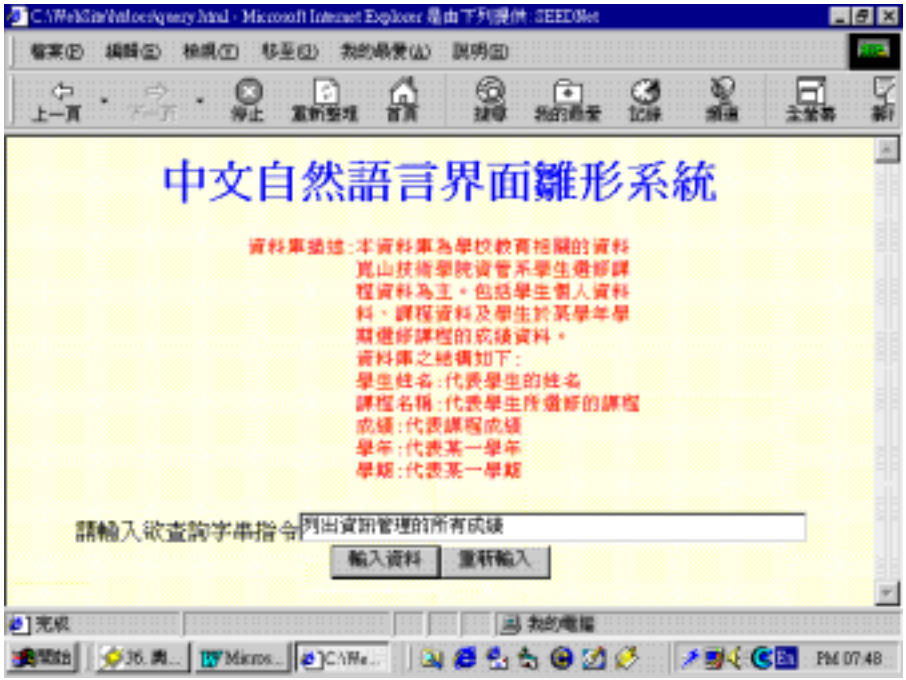
SELECT 欄位 … [FROM 表格] WHERE 條件…

而「計算查詢」則只支援「單欄」的文法，且亦無支援單一條件與多值條件的分類，但仍區分為「完全查詢」與「不完全查詢」兩類，其對應 SQL 處理的格式如下：

SELECT 聚合函數(欄位) [FROM 表格] WHERE 條件

聚合函數：SUM, AVERAGE, MAX, MIN, COUNT 等

肆、系統範例



圖九　本雛形系統之人機介面（本研究）

現今網際網路應用的支援技術中，主要可以分類為前端（客戶端）及後端（伺服器端）兩種技術，前端的技術僅限於瀏覽器所能支援的能力，例如 Java、JavaScript、VbScript 等，而這些語言的程式處理能力並不如預期強大。至於後端的技術一般都是與程式語言直接結合，例如 CGI、ASP 等，但 ASP 在支援遞迴處理似乎不像 CGI 般強大，所以本研究主要以 CGI 方式來建構本系統的雛形介面。

圖九即為本雛形系統的人機介面，因使用者完全不須知道資料庫的內容及查詢路徑，所以在查詢介面中會以簡要的描述說明資料庫內的組成，減少使用者不知如何進行查詢的窘境。

現假設使用者輸入“列出資訊管理的所有成績”的資訊需求，如前一章所述，會送入文法轉換模組進行解譯，依據圖七的語意轉換樹文法文法，此資訊需求會追蹤 Question、Wheres、Cvalues、Cvalue、Determiner、Attrs、Attr 等子樹，當 Wheres 子樹成功傳回參數值，則 Question 主樹便成功地到達第一條分支的終端節點，亦即使用者輸入的中文自然語言已被系統接受。得到的 SQL 語法傳送給資料庫伺服器，產生的查詢結果便以 HTML 的方式回傳給使用者，其查詢結果如圖十所示。

使用者除了不須知道資料庫的內容及查詢路徑，且也不須具備任何查詢語言的電腦知識，只要使用平常口語化的

圖十 “列出資訊管理的所有成績”的查詢結果（本研究）

中文，在限定格式下即可獲得其所要的資料。至於“列出”、“給我”、“顯示”等相同意義的動詞字彙，本系統已將大部分常用的內建於文法庫中，換句話說，在“列出資訊管理的所有成績”的資訊需求中，使用者以“給我”替代“列出”，亦能得到圖十的結果。

在上述的範例中，亦是不完全查詢(Elliptical query)的例子，因資訊需求中並沒有告訴系統“資訊管理”是那個欄位的值，所以本系統是可以支援處理部分的不完全查詢。值得一提的是，在人機介面的輸入方面，本系統藉由與語音輸入軟體的結合，取代了傳統的鍵盤輸入方式，更增加了系統操作的友善性。

至於系統績效的評估方面，本系統伺服器的硬體環境是 PentiumⅢ個人電腦、64M RAM，配合支援 CGI 的 Website 2.0 伺服器軟體，實際執行後的反應時間測試，其平均反應時間均在半分鐘以內，然此項標準其實亦有多方面影響的因素，諸如電腦硬體設備的等級，或者調整主從架構的執行負載，亦即可將部分處理移轉到客戶端執行，這些都可以加快系統的反應時間，故在一般的應用中，本系統的執行績效是可以被接受的。

伍、結論與建議

一、結論

有鑑於友善介面與本土化的目的，本研究發展一個中文自然語言的資料庫擷取雛形介面，以語意轉換樹文法來簡化中文自然語言的斷字及文法剖析等複雜處理，提供了與傳統中文自然語言處理不同的另一種方式。在語意轉換樹文法方面，雖然此種表示方法不是本研究所提出，但以往的應用只侷限在英文自然語言，且英文與中文的語法並不相同，本研究將其應用在中文自然語言，建構出適用本土語言的語意轉換樹文法，日後更將朝向一般化的原則，以提高不同應用領域的移植性。

自然語言的查詢方式與其它的傳統介面比較，本就具有壓倒性的友善優越性，未來必成為人機介面的主要趨勢。同時本研究更將整個系統移植到網際網路環境，透過主從架構的平台，建構一地點透通性的查詢介面，更可以得到單機系統所無法獲得的優點。在本研究架構下，最後並以 CGI 後端技術發展出一個雛形系統，使得本系統架構獲得可行性的驗證。

至於有關反應時間的測試，在本雛形系統的實際執行下，其平均反應時間均在半分鐘以內，然此項標準其實亦有多方面影響的因素，諸如電腦硬體設備的等級，或者調整主從架構的執行負載，亦即可將部分處理移轉到客戶端執行，這些都可以加快系統的反應時間。在中文自然語言輸入部分，使用者除了可以用鍵盤輸入外，亦可結合語音輸入軟體，以說話方式輸入中文自然語言，更可提高系統的友善性。

二、建議

對於本自然語言介面系統後續的研究，可以針對下列兩方面說明：

(一)文法庫學習方面

由於中文自然語言處理在斷字、語意分析等技術尚未有進一步地突破，故本研究現階段是以限定中文自然語言格式的方式來輸入，如此系統實用的限制也相對地增加。也就是說，文法庫內的語意轉換樹文法是有限的，造成系統解譯失敗的機率也會提高，為了能夠降低系統解譯失敗的機率，便是系統本身具有學習文法的能力，在多次的使用之後，系統能夠自動地強化文法庫，讓往後使用的解譯失敗機率降低，提高系統實用性。

(二)資料庫管理方面

由於本研究所提出的自然語言介面架構，主要是針對單一資料表的查詢處理，所以提出下列幾點建議以供後續研究參考：

1.為了防止更新異常, 資料正規化是必須的步驟, 但這也導致多資料表存在的普遍現象. 所以本系統的後續研究乃希望能適用於多資料表的資料庫介面, 如此才能達到實際推廣應用的目的。

2.在資料庫應用中, 查詢處理是最主要且頻繁的功能, 但新增、刪除、修改的處理亦屬重要. 為了減少資料庫管理的不便, 所以後續的研究希望能夠將新增、刪除、修改等資料處理功能納入自然語言介面, 提供使用者一致性的使用與管理。

3.在本研究所提出的自然語言介面中, 如前所述, 除了能夠支援資料庫內現有欄位資料的查詢, 還能提供欄位的聚合運

算查詢。但是對於衍生欄位與計算欄位的查詢則尚未支援，所以在後續研究中，希望能夠實現上述的功能於自然語言介面中，以活潑查詢的多樣化，提供更有效率及價值的資訊。

參考文獻

一、中英部份

1.蔡燿全、楊棠堯，(1999)，「關連式資料庫之 SM 查詢介面研究」，1999 中華民國科技管理研討會論文集，第一集，頁 567-577。

二、英文部分

1.Allen, J. (1987). Natural Language Understanding. The Benjamin / Cummings Publishing Company. Menlo Park. California.

2.Astrahan, M. Blasgen, M. Chamberlin, D. Eswaran, K. Gray, J. Griffiths, P. King, W. Lorie, W. McJones, P. Mehl, J. Putzolu, G. Traiger, I. Wade, B. and Watson, V. (1976). System R:A Relational Approach to Database Management, ACM Transactions Database Systems, 1, 97-137.

3.Chamberlin, D. and Boyce, R. (1974). SEQUEL-A Structured English Query Language. in Proc. ACM SIGMOD Workshop Data Descript.

4.Codd, E. (1977). Seven Steps to RENDERZVOUS with the Casual User. North-Holland.

5.Grosz, B.J. Appelt, D.E. Martin, P.A. and Pereira, F.C. (1987). TEAM:An Experiment in the Design of Transportable Natural Language Interfaces. Artificial Intelligence, 32, 173-243.

6.Hancock, P.A. and Chignell, M.H. (1989). Intelligent Interfaces:Theory, Research and Design. North-Holland, New York.

7.Hendrix, G.G. Sacerdoti, E.D. Sagalowicz, D. and Slocum, J. (1978). Developing a Natural Language Interface to Complex Data. ACM Transactions on Database Systems, 3(2), 105-147.

8.Huang, H.C. and Chen, C.T. (1989). The Intelligibility, Naturalness, and Applicability on Future Trend of The Query Language. Database, 12.

9.Ishikawa, H. Izumida, Y. Yoshino, T. Hoshiaoi, T. and Makinouchi, A. (1987). KID: Designing a Knowledge-Based Natural Language Interface. IEEE Expert, 2(2), 56-71.

10. Janas, J.M. (1986). The Semantics-based Natural Language Interface in Relational Databases. Cooperative Interfaces to Information Systems, 143-188.

11. Kao, M. Cercone, N. and Luk, W. (1988). Providing Quality Responses with Natural Language Interface: the Null Value Problem. IEEE

Transactions on Software Engineering, 14(7), 959-984.

12. Khurana, G.S. and Gadhok, S.S., (1997). Microsoft Visual Basic Web Database Interactive Course. Waite Group Press, Corte Madera, CA.

13. Kok, A.J. (1995). The Design and Implementation of an Intelligent Query Tool For Relational Databases. Expert Systems, 12(4), 347-351.

Mahar, P. and Henderson, K. (1997). Teach Yourself IntraBuilder in 21 Days. Borland Press, New York.

14. McDonald, N. and Stonebraker, M. (1975). CUPID- The Friendly Query Language. in Proc. ACM Pacific Conf., San Francisco, CA.

15. McFetridge, P. Hall, G. Cercone, N. and Luk, W.S. (1988). System X: A Portable Natural Language Interface. Proceedings of the Seventh Biennial Conference of the Canadian Society for Computation Studies of Intelligence, 30-38.

16. Mohseni, P. (1996). Web Database Primer Plus. Waite Group Press, New York.

17. Nabil, R.A. Aryya, G. and James, C. (1994). A Form-Based Approach to Natural Language Query Processing. Journal of Management Information Systems, 11(2), 109-135.

18. Sabbagh, S. (1991). SESAME: An Application of Entity Relationship Models to a Natural Language User Interface. Entity-Relationship Approach: The Core of Conceptual Modeling, 319-331.

19. Templeton, M. and Burger, J. (1986). Considerations for the Development of Natural-language Interfaces to Database Management Systems. Cooperative Interfaces to Information Systems, 67-99.

20. Wald, J. (1985). Problems in Query Interface (VERDI). Ph.D. dissertation, Dep. Computational Sci., Univ. Saskatchewan, Saskatoon, Canada.

21. Waltz, D.L. (1978). An English Language Question Answering System for a Large Relational Database. Communications of the ACM, 21(7), 526-539.

22. Winston, P.H. (1992). Artificial Intelligence. 3th Ed, Addison Wesley Publishing Company, New York.

23. Woods, W.A. (1978). Semantics and Quantification in Natural Language Question Answering. Advances in Computers, 1-87.

24. Zloof, M. (1975). Query by Example. in Proc. NCC, AFIPS, 44, 431-438.

2000 年 11 月 07 日收稿
2000 年 11 月 15 日初審
2001 年 02 月 05 日接受