第三章 工作媒合系統的探討

本章旨在探討工作媒合系統的相關文獻，首先探討國內外工作媒合系統的發展歷史，接下來探討工作媒合系統的各種媒合方法，作為本研究發展工作媒合系統的參考。

第一節 工作媒合系統

一、國外的經驗

在美國，工作媒合系統的發展已經有一段很長的時間，Nathanson (1975)就針對當時的一個工作媒合系統 Cleff Job Matching System (CJMS)進行評鑑。該系統可以針對白領與藍領階級所需要的低技能或半技能工作，進行求才者與求職者之間的工作媒合。資料來源係由供需雙方以紙筆完成 16 個向度的問卷填答，並轉換成為數值資料後輸入至 CJMS。然後 CJMS 藉由雙方資料的相互關聯與其他的統計技巧，將媒合的分數排序輸出。再藉由媒合分數與應徵者就業後停留在工作職位上的時間來進行效度分析，在追蹤 142 個樣本中發現停留時間由 14 到 21 個月不等，媒合分數越高者的停留時間也越長。因此，CJMS 的媒合分數可以作為就業停留時間長短的有效指標。另外經由和人力資源的職員及主管就業的官員的面談中發現，CJMS 對於評量個人能力與職業諮詢方面極有幫助，報告中也建議 CJMS 應擴大應用於全國的就業部門。

Nathanson (1975)的報告顯示了工作媒合系統的發展非常的早，當時的年代只有中大型主機，採用集中式的處理與終端機連線，文字模式的操作介面，供需雙方也必須要先填答問卷，再轉換成輸入資料。在硬體的普遍性與軟體的應用性方面都受到了極大的限制，儘管如此，工作媒合系統還是能夠發揮其成效，並且大大的被接受。

在微電腦出現之後，由於價格逐漸下降且處理能力提升，許多的軟體被發展出來，並且進入人類的日常生活與專業生活中。其中一項為電腦化的工作媒合系統 (computerized job matching systems)。所謂電腦化的工作媒合系統就是一組軟體程式，可以媒合應徵者的專業檔案與工作職位的需求，並且決定應徵者的資格是否符合或超過工作職位的需求。Botterbusch (1986)比較了全美 15 個電腦化的工作媒合系統，這些系統有五項共同的性質：

1、基於特徵與因素 (trait-and-factor basis)的媒合：所有的工作媒合系統基本上都是特徵與因素的邏輯延伸，也就是說，每一件工作或任務都是許多特定需求的組合。這些需求是可以辨識與量化的，當應徵者的技能與態度等於或超過工作需求時，媒合的結果就會成功。因此，可以用下列的式子來加以判斷：

應徵者的能力 >= 工作需求

換句話說，就是：

應徵者的特徵 >= 工作表現的因素

儘管這樣的概念很簡單，但是事實上的媒合方法卻不容易。為了要辨識應徵者的能力，必須要進行工作分析、任務分析、工作經驗分析、心理測驗、結構化訪談，或其他方法等，用來預測應徵者是否有能力與意願來執行工作。在工作需求的方面，也要進行

工作分析、任務分析、與工作分類分析，以便形成標準化的工作需求。而媒合系統就是要拿應徵者的能力與標準化的工作需求做媒合，以便計算媒合的分數。

2、使用職業分類典中工作分析的術語 (use of DOL job analysis terminology)：大部份的工作媒合系統都是基於兩種資料庫，一為美國勞工部所發展的職業分類典第四版，及職業試探指引 (Guide of Occupational Exploration; GOE)，大部份的系統都要求輸入下列變數：一般教育發展檢定 (General Education Development Test；GED)、特定職業準備 (Specific Vocational Preparation；SVP)、態度 (aptitudes)、實體技能需求與限制、環境條件與兩極興趣 (bipolar interests)等。這些變數的數據都必須依賴標準化的文件，如果不瞭解 DOL、GOE、GED、與 SVP 等，根本無法輸入資料，也無從解釋輸出結果。

3、方法論不是新的 (methodology is not new)：工作媒合是要拿應徵者的特徵與 DOT 的因素作媒合，這樣的概念已經發展了數十年。這 15 個系統也是使用一樣的概念，並非新的方法論。

4、垃圾進垃圾出 (garbage in garbage out)：由於媒合系統的輸入變數很多，如果不是經由詳細規劃與評量所得的資料，就會造成垃圾進垃圾出的情況。

5、增加專業的責任 (increased professional responsibility)：由於大部份的系統都是根據一些標準化的資料庫，系統的輸入變數與輸出結果很多。在解釋方面必須要擁有相關的知識，同時還需要經過訓練，而供需雙方與專業人員諮商會是一個較佳的作法。

加州政府就業發展部 (California Employment Development

Department)在 1997 年時就發展了網頁式的工作媒合系統，稱為加州工作機會瀏覽系統 (California Job Openings Browse Systems；CalJOBS)。選擇全球資訊網的原因是網際網路是一個開放式的環境，允許和不同的電腦系統之間相互溝通，而且連結網際網路的單位與個人越來越多。Web 的圖形介面不需要太高的電腦技能門檻，該系統採用主從架構、分散式處理機制、Informix 資料庫管理系統、物件導向的電腦語言 Delphi 等。提供的功能包括：雇主可以直接張貼工作機會的訊息、也可以搜尋應徵者的履歷表資料；應徵者可以搜尋工作機會，也可以輸入或寄送電子履歷表給有潛力的雇主等，其搜尋機制是採用複合式的關鍵字搜尋。

全球資訊網盛行之後，工作媒合系統的平台顯然已經轉移到了網際網路上，並且以工作銀行 (job bank)的型態出現，其數量非常的多。為了提供使用者搜尋工作銀行，美國勞工部還建立了一個網站，可以搜尋全美各州所有的工作銀行 (Careeronestop, 2007)。除了平台轉移之外，所有的操作介面也由文字模式轉換成圖形模式，並且由集中式的操作模式轉換成分散式的模式。而早期基於職業分類典與相關標準的工作媒合方式也早已不復見，可能的原因是美國的職業分類典(DOT)到目前為止仍然停留在 1991 年的第四版。十餘年來職場的變化太太，DOT 的職業名稱早已經無法反應現實的狀況。因此，工作媒合系統上早已經看不到以 DOT 的職業名稱或代碼來搜尋工作相關資訊的選項 (Dice, 2007； Monster, 2007)，而改成直接以技能名稱、職務名稱、與地理位置等來做關鍵字的搜尋。

二、國內的情況

在國內部份，遍查了所有的文獻之後，並沒有發現任何有關工作媒合系統、就業媒合系統、或人才(力)媒合系統的相關文獻。顯示國內早期並未發展電腦化的工作媒合系統，可能的原因是國內有完善的就業輔導網路，由政府單位主導的就業服務中心到處林立，再加上報紙或雜誌的就業廣告，所能提供的工作機會資訊應該可以滿足需求。到了網際網路普遍深入各行各業與家庭之後，以全球資訊網為基礎的人力銀行相繼成立，台灣地區的人力銀行總數已經超過 25 家，其中大部份是綜合性的人力銀行，少部份是單一職業的人力銀行，例如餐旅、幼教、傳播與教師等。這些人力銀行所提供的工作查詢或人才查詢，大多是屬於傳統的關鍵字搜尋，相關的文獻已於第一章研究動機中詳述。

人力銀行確實提供了一個求才與求職的訊息交換平台，透過內部的工作媒合系統，可以自動將應徵者的專業檔案資料送到企業所指定的信箱，而企業也透過此平台提供職缺訊息，讓求職者可以自行將個人履歷投至企業中。這就可能造成大量的人才履歷資料充斥信箱，對於企業來說，究竟這些人才履歷表有多少是有效的資料？此外，以現行人力銀行提供給求才者的輸入介面而言，由於無法描述不同項目的重要性，對於用人單位的真正需求，根本無法確切地掌握。因此，只能針對應徵者的一般性資料，例如教育程度、主修專業、工作年資與經驗等進行初步篩選。再將認為合適的應徵者資料轉寄給用人單位，然後由用人單位挑選適當的人選進行約談面試的工作。但是，工作媒合系統認為的合適人選，是不是就等於用人單位主管心目中的合適人選？彼此對於好人才的定義與認知是否一致？如果一個工作媒合系統在這些方面無法獲得解決，會不會就此阻斷了許多優秀人才？因

此，工作媒合系統應該有更有效的方法，才可以更加精準地篩選人才。

三、工作媒合系統的功能

雖然大部份的工作媒合系統都有前述的五種性質，但是也有許多差異之處，有些系統提供多樣的功能，但也有些系統只提供了媒合的功能。Botterbusch (1986)認為一個工作媒合系統至少應該提供下列五種功能：

1、安置的功能 (placement)：這種功能可以讓社會大眾、學生、教師、諮商師、人力部門專員與安置專家等，搜尋與應徵者或求才者相互匹配的工作。

2、轉移技能的功能 (transferable skills)：如果系統可以讓應徵者輸入所有的工作經驗，這些工作經驗的涵蓋面可能很廣泛。系統並且有能力將這些工作經驗中的技能組合起來，形成一個新的技能檔案稱為轉移技能，這位應徵者的工作媒合結果將會更廣泛。

3、職業資訊 (occupational information)：由於 DOT 的職業資訊的定義超過 12000 個，任何一位應徵者或諮商師都不可能全盤瞭解，如果系統提供職業資訊的功能，就可以讓應徵者印出所有適合的職業，提供完整的職業資訊。

4、諮商的功能 (counseling)：工作媒合的媒合是以靜態的資料為主，其輸出是應徵者可以擔任的工作。如果系統提供諮商的功能，就可以在此基礎之上，以自動或手動的方式來改變某些變數的內容，以獲得更多或更好的工作機會。讓應徵者瞭解應該加強或學習某些新的技能，達成諮商的功能。

5、教育與訓練的功能 (education and training)：如果工作媒合系統可

以和正規的教育系統或訓練計畫相互關聯，就可以提供教育與訓練的資訊。

四、工作媒合系統的問題

Wong (1992)指出，雖然電腦化的工作媒合系統有其特定的功效，但是仍然存在一些問題。首先是當時的系統都是基於政府機關所訂定的職業分類典 DOT，供需雙方在職業名稱的選擇上只能根據 DOT 的職業名稱代碼。而 DOT 內部所描述的工作需求、技能需求、態度需求與其他向度的需求等，也必須要符合當下的工作環境。但是 DOT 的修訂不可能反應當下工作市場的實際情況，以致於造成資料輸入的困擾，甚至是無法使用媒合系統的情況；第二個問題出在媒合系統是一個封閉的系統，媒合的分數往往只是二分法的適合或不適合，不像人工媒合可以根據情況客觀的賦予媒合分數；第三個問題出在工作媒合系統輸出的適合度，該適合度完全要依賴輸入資料的信度與程式內部搜尋演算法的效度。為了符合大多數人的需求，程式通常選擇適應輸入資料的彈性，而犧牲輸入資料的精確度。由於資料的精確度往往不容易控制，例如利用兩種類似的標準化測驗，就可能導致精確度的顯著差異。

Wong (1992)也指出，根據明尼蘇達工作適應理論（Minnesota Theory of Work Adjustment)，工作媒合應該同時考慮工作滿意度 (job satisfaction)，也就是人與工作環境的適配程度(person-job environment fit)，與工作適任度（job satisfactoriness)，也就是人與工作需求的適配程度 (person-job requirement fit)。但是，大部份以職業分類典為導

向的工作媒合系統，都只強調個人與工作需求的適配程度，諸如技能、態度等。這種情況可能會導致人與工作環境適配程度不佳的問題，例如一位擁有高中文憑的卡車司機，已經具有多年的卡車司機資歷，也具有高階的辦公室技能。工作媒合系統很容易替他找到一份辦公室職員的工作，但是在人格上、工作型態上，這個人可能不容易適應辦公室的工作。

電腦化工作媒合系統的最大問題可能還是在於系統設計上的缺失，大部份的系統使用高度結構化的傳統資料庫模式，導致系統的搜尋機制非常死板。如果無法完全匹配，該筆查詢就被認定是失敗。因此，工作媒合系統的決策完全是直線式的，並且假設 DOT 的所有資料都具有高度的信度與效度。但事實上 DOT 可能無法即時反應現狀，導致本來應該是高度匹配的個案，被媒合系統認定為完全無效，這種天差地遠的結果導致工作媒合系統的不穩定性。

針對電腦化工作媒合系統的諸多缺失，Wong (1992)提出了第二代工作媒合系統的設計架構，其媒合對象是身心障礙的人員。對於一個有身心障礙的人而言，每一個人適應工作環境的變異可能非常的大。因此，人與工作環境的適配程度必須是工作媒合系統設計的一個重點，一個可能的策略是發展一個以知識庫為基礎的專家系統，由集體智慧、專家的知識與經驗等所組成，並且可以由媒合之後的反應來學習。在查詢方面也必須揚棄二分法的結果，應該在模糊、不確定與不完整的資訊中進行查詢。要達到這樣的功能，工作媒合系統的架構可以包括下列六個元件：

1、推理引擎 (inference engine)：是工作媒合系統的核心，內含推理機制、互動機制等。

2、知識擷取子系統 (knowledge acquisition subsystem)：首先要建置知識庫，知識的來源有教科書、研究報告、資料庫、個案研究、實證資料與經驗等。

3、知識庫子系統 (knowledge base subsystem)：將擷取到的知識組成推理規則。

4、解釋子系統 (explanation subsystem)：將結果顯示並解釋。

5、使用者介面 (user interface)：讓使用者可以和工作媒合系統互動。

6、模糊處理子系統 (uncertainty handling subsystem)：處理定義不明確的推理。

第二節 語意相似度

一、語意相似度的定義

語法 (syntactic)和語意 (semantic)是兩個相對應的名詞，在一個自然語言的表示中，就存在著語法及語意。語法是指自然語言表示的表面符號，語意則是其內部所含有的意義。例如 Java 這個字的本身是語法，其語意可能是指 Java 電腦程式語言、或是印尼的 Java 島、也可能是指 Java 咖啡。

Giunchiglia 與 Shvaiko (2003)把「媒合」或稱為「比對」(matching)定義為一種二元運算子，接受兩個圖形結構的運算元，例如資料庫的綱要或本體。媒合有兩種不同的方式，一種是語法媒合 (syntactic

matching)，或稱為標記媒合 (label matching)，係利用語法驅動的技巧，來判斷語言學上或結構上的組成是否相互匹配。例如 phone 與 telephone，就有語法上的相似，因為這兩個字具有相同的結構 phone；另外一種是語意媒合 (semantic matching)，係利用兩個標記在意義上或概念上的接近程度來作比較，標記之間可能具有類似的結構，例如 phone 與 telephone 的語意就極為接近；標記之間也可能具有完全不同的結構，例如 construction 與 building，也具有相近的意義。

語意相似度 (semantic similarity)與語意相關度 (semantic relatedness)在意義上是有所分別的，語意相似度是以兩個概念在整體概念架構中所含有的資訊量來衡量，而語意相關度則是以兩個概念相互結合的程度來衡量。舉例來說，汽車與汽油的相關度，比汽車與腳踏車的相關度要高，汽車與汽油在概念上是完全不相似的東西，但是汽車與汽油卻具有高度的相關，汽車沒有了汽油就不能啟動；而汽車與腳踏車的相似度很高，這是因為汽車與腳踏車具有相似的概念，他們同樣是屬於交通工具的一種。簡單言之，兩個概念之間相似的程度，就叫做語意相似度，當兩個概念之間的意義完全不同時，其語意相似度為 0，當兩個概念的語意完全相同時，其語意相似度為 1，也就是說，語意相似度是界於 0 與 1 的小數，也可以用百分比來表示。

二、語意相似度的應用

語意相似度被廣泛的應用在許多不同的領域，例如資訊檢索、資訊萃取、本文分類、基於實例的機器翻譯等。例如要翻譯「張三寫的小說」，通過語料庫檢索得到譯例：「李四寫的小說／the novel written

by Li Si」與「去年寫的小說／the novel written last year」，經過相似度計算發現，「張三」和「李四」都是具體的人，在語意上非常相似，而「去年」和「張三」的語意相似度較低。因此，應該選用第一個實例來進行類比翻譯，就可以得到正確的譯文 (劉群 & 李素建, 2002)。

語意相似度也常被應用在字義的分辨，Agirre 與 Rigau (1995)提出了概念密度 (conceptual density)的方法，係利用 WordNet 的結構化架構與上下文的關係，來作為字義分辨的基礎，一個字的正確意義是和它的上下文有密切的關係。也就是說，一個字的概念密度是和正確意義的字之頻率成正比，換句話說，一個字的正確意義是和上下文中相似度最高的字之字義相似。

語意相似度早就廣泛的被應用在資訊搜尋或資訊擷取的查詢擴展上 (query expansion)，一般的作法是找出關鍵字的同義詞，再利用同義詞來作查詢擴展。另外一種作法是利用先前查詢所獲得的相似度，以相似度高的查詢來作查詢擴展。Vlez, Wiess, Sheldon, & Gifford (1997)利用第二種方式，拿目前關鍵字的語意相似度與先前使用者的查詢作比較，如果先前的查詢和現在的查詢有語意上的相關，就可以建議使用者採用查詢擴展，或是在搜尋引擎中直接進行查詢擴展。

利用關鍵字搜尋已經可以找到很多相關的文件，但是對於基因、藥物、社會網路與國家安全等超大型的領域而言，關鍵字搜尋絕對是不夠的，必須使用語意相似度來搜尋極大型的知識庫。以國家安全領域為例，使用語意相似度的搜尋，也許可以發現可疑的活動，例如洗錢、內部威脅與恐怖活動等，以便事先加以預防或解除。依據這樣的

理念，Aleman-Meza, Haschek-Wiener, Sahoo, Sheth, 與Arpinar (2005)使用一種樣板式的相似度 (template-based similarity)來搜尋與辨識情境，這裡的樣板是把與國家安全事件有關的概念、關係與實例等蒐集起來，並且將其相互連接。因此，樣板事實上就是本體的觀念，當使用與國家安全相關的關鍵字作搜尋時，利用該關鍵字可以找到所建立的樣板，再利用樣板當作查詢依據，就可以發現與定義好的樣板有關的國家安全事件的文件。

語意網的成功與否完全要依賴相關領域的本體是否充足完整，以及網頁是否利用領域本體的詮釋資料來加以註解，問題是這些詮釋資料如何獲得，傳統的作法是人工發展本體與人工註解網頁。如果無法採用自動化的方法，語意網的應用將是遙遙無期。Cimano, Handschuh, 與 Staab (2004)建議採用 PANKOW 取向 (Pattern-based Annotation through Knowledge on the Web)，這是一種自動化、以樣板為基礎的網頁內容自我註解方法，其主要的觀念是利用全球資訊網頁的資料與結構，來當作一個大型語料庫，以便克服資料稀疏的問題，再結合語言學上的語型樣板來辨識某些本體的關係。PANKOW 採用最大證據原則，也就是說，當某一個實例需要註解時，其概念可以由全球資訊網上具有最大語意相似度的概念來加以註解。

前述的應用都是屬於比較大型的應用，事實上，語意相似度也可以應用在極為小型的個案上。Wang, Su, Wang, 與 Ma (2007)在一個 C 語言程式設計的線上考試系統中，利用已經塑模的標準化程式作為基礎，計算每一位學生的程式作業與此標準程式之間的語意相似度，再給予評分。

網際網路出現之後，傳統的人才招募方式顯得又昂貴又沒有效率，全球資訊網帶來了新的人才招募方式，雇主可以將職位空缺及其需求，張貼在公司的網站或人力銀行。同樣的，求職者也可以將自己的履歷表以電子郵件方式直接寄送給公司，或儲存到人力銀行的資料庫中，既然雙方的資料都在網路上面，就沒有理由再使用傳統人工作業的方式。目前大部份的自動化工具傾向以語法方式作媒合，但是求才者對於工作需求的描述與求職者對於個人能力的描述，可能不盡相同，例如「維護資料庫系統」對「維護SQL Server」。如果以關鍵字的「資料庫系統」或「SQL Server」來搜尋，其結果是完全不相關的。因此，以關鍵字搜尋可能影響媒合的結果，也可能會遺漏非常適合的工作或人才，使得媒合的效能打了折扣，不容易達到適才適所的目標。改進的方式是供需雙方應該採取語意網的科技來做自動媒合，也就是要延伸語法導向的媒合到語意導向的媒合。

三、語意相似度的理論基礎

到底應該如何量測語意相似的程度？Lin (1998)作了一個基礎理論研究，認為相似度是一種非常基本而且常用的概念。但是在過去的研究中，大部份相似度的計算都是以實證的研究為主，並且都和某種特定的應用綁在一起，或是假設某一種特定的領域模式，並沒有很明確的說明所根據的理論基礎。因此無法做理論的探究或辨證，Lin 因而提出了相似度的理論基礎，希望能夠達成普遍性與理論驗證兩個目標，其相似度的計算是建立在下列三個觀察上面：

l、物件 A 與 B 的相似度是和他們之間的共同性 (commonality)有關：

共同性是指兩個物件中相同成份的屬性，當相同的成份越多時，

兩個物件的相似度也愈高，表示兩者所含有的共同資訊量就越多，也就是在內涵上越趨近於一致。

2、物件 A 與 B 的相似度是和他們之間的差異性 (difference)有關：當差異性越大時，其相似度也愈低。差異性和共同性是互斥的兩種性質，當差異性越大時，其共同性就越少。

3、當物件 A 與 B 完全一樣時，其相似度最高，一般而言，當兩個物件雷同時，其相似度為 1，當兩個物件沒有一點共同性時，其相似度為 0。

基於前述三個觀察，Lin (1998)進一步提出了六個合理的假說，來定義相似度的量測依據：

1、物件 A 與 B 的共同性可以用 *I*(*common(A, B)*)來衡量，*common(A, B)* 是一個命題，用來陳述物件 A 與 B 之間的共同性，*I* 是指所含有的資訊量，*I*(s)是命題 s 所含有的共同資訊量，而 *I*(s) = -*log P*(s)，此處 *P* 為機率。

2、物件 A 與 B 的差異性可以用 *I(description(A, B))-I(common(A, B))* 來衡量，此處 *discription(A, B)*也是一個命題，用來完整描述 A 與 B 是什麼。因此，A 與 B 的差異性就是兩者的完整性減去兩者的共同性。

3、物件 A 與 B 的相似度可以表示成 *Sim(A, B)*，是 A 與 B 的共同性與差異性的函數，也就是說：

Sim(A, B)=f(I(common(A, B)),I(description(A, B))。

4、兩個相同物件的相似度為 1。

5、兩個相異物件的相似度為 0。

6、如果物件 A 與 B 都可以由兩個不同的角度來觀察，其相似度也可

以由兩個角度來衡量。

最後 Lin (1998)提出了兩個物件 A 與 B 之間相似度的定義，即描述 A 與 B 共同性所需要的資訊量與完全描述 A 與 B 所需要的資訊量之比，如下所示：

$Sim(A, B)=\frac{\log P(common(A,B))}{\log P(discription(A,B))}$，此處的 P 是機率。

根據前述的定義，Lin (1998)也舉出例子來說明相似度的計算：

1、序數的相似度計算：Lin 認為可以利用序數的機率分配與共同資訊量的定義來計算相似度，例如序數 Excellent、Good、Average、Bad、Awful，其機率分配為 5%、10%、50%、20%、15%，則

Sim(Excellent, Good)=

$$\frac{2\times\log P(Excelent\vee Good)}{\log P(Excelent)+\log P(Good)}=\frac{2\times\log(0.05+0.10)}{\log(0.05)+\log(0.10)}=0.72$$

Sim(Excelent, Average)=

$$\frac{2\times\log P(Excelent\vee Good\vee Average)}{\log P(Excelent)+\log P(Average)}=\frac{2\times\log(0.05+0.10+0.50)}{\log 0.05+\log 0.50}=0.23$$

2、字串的相似度計算：字串的相似度可以應用在字串擷取與相關文件的搜尋上，例如要由字典中擷取具有共同字根的字串、或是具有相關的字串，Lin 舉出三種不同的定義，來計算字串的相似度，其中的兩種定義如下：

(1)、第一種定義如下：

$$Sim_{edit}(x,y)=\frac{1}{1+editDist(x,y)}$$

此處 *editDist(x, y)*是指要將 *x* 與 *y* 變成相同的字串所需要編輯的字元個數，例如 grandiloquent 與 eloquent 的差異字元有 grandie 等七個字元，因此 *editDist(grandiloquent, eloquent)*=7，而這兩個字串的相似度為 0.125。

(2)、第二種定義如下：

$$Sim(x,y)=\frac{1}{1+|tri(x)|+|tri(y)|-2\times|tri(x)\cap tri(y)|}$$

此處 *tri(x)*是將 *x* 拆成三個字元為一個單元的單元總數，例如 *tri(grandiloquent)* = {gra, ran, and, ndi, dil, ilo, loq, oqu, que, uen, ent }，*tri(eloquent)* = {elo, loq, oqu, que, uen, ent }，因此

$$Sim(grandiloquent, eloquent)=\frac{1}{1+11+6-2\times5}=0.125$$

Lin (1998)所提出的相似度的計算方式，其中字串相似度又可以利用不同的定義來計算。但是不管如何的定義，所計算出來的相似度只能說是字型上或語型上的相似度，不能說是字義上或語意上的相似度，例如eloquent與ineloquent，不管用前述那一種的定義來計算，這兩個字串的相似度都大於0.3。但是在語意上，這兩個字正好有相反的意義，按照Lin給相似度所下的定義，這兩個字之間應該沒有任何的共同性，也就是沒有任何的相似度。再看另外兩個字串human與person，以前述的標準來計算相似度，所得到的結果應該為0，但是實際上這兩個字在語意上確實存在有一定的相似度。

在序數的相似度方面，序數的應用場合有很多，例如教育程度、

技能等級與問卷調查等。過去並沒有研究針對序數的相似度進行定義，Lin是第一位提出者，但它的定義係使用機率分配的方式來計算相似度，其機率分配的總和最大只能到達100%。如果機率分配總和超過100%，計算的結果就會成為負值，例如按照前述的計算方式，60%與50%的相似度為-0.158。在實務的場合上，經常會將技能的等級加以劃分，例如初學者可能只擁有20%的技能、進階者為50%、專精者為70%、而教練者為90%，這種相似度的計算就不能使用Lin的定義。在字詞的相似度計算方面，Lin利用語法剖析器自所蒐集的語料庫 (corpus)中取得字詞的特徵值，並由其中的共同資訊量來計算相似度，該資訊量基本上還是在語法層次，並沒有深入到語意層次。

第三節 工作媒合的方式

一、基於能力註解比對的工作媒合

傳統的人才招募方式是利用報紙、雜誌、行業新聞、就業博覽會、雇用代理人與人才招募公司等，作為資訊傳播與互換的平台。在過去，這些管道或許已經足夠，但是以今天的資訊傳播環境而言，這些傳統的作法顯示出速度過慢、花費昂貴與資訊傳播量不足等缺點。許多企業與個人開始運用電子化的人才招募方式，求才者與求職者雙方利用網際網路來散播相關的資訊，一種是透過人力銀行提供給供需雙方的標準化介面來輸入求才者所需要的相關條件，與求職者所具備的資格與能力等。另外一種是以自由格式的文字來描述工作的職位或履歷表，並將職位描述或履歷表分送到工作銀行 (job bank)或履歷表銀行 (CV bank)，再利用這些平台來做資訊交換，達成求才或求職的目

的。

Bourse, Harzallah, Leclère, 與 Trichet (2002) 推動了一項 COMMONCV 的計畫。有兩個目的，第一個目的是要讓求才者可以明確的描述工作上的需求，也讓求職者可以描述個人所獲得的能力。第二個目的則是要以正規的方法將需求與能力表示出來，以便進一步提供更有效的工作媒合服務。要達成這兩個目的，必須要定義能力的模式與能力的處理過程，在能力模式方面的定義如下：

$$C_i = (K, B, C, A, o)$$

此處 C_i 是一個能力，K 是獲得 C_i 所需要的知識、B 是獲得 C_i 所需要的行為、C 是獲得 C_i 所需要的基本能力、A 是獲得 C_i 所需要的觀點 (aspect)，也就是不同的環境、o 則是需要達成的目的。在能力處理過程部份，則先要建立一些領域本體，例如行業本體 (sector ontologies)、企業本體 (enterprise ontologies)、技能本體 (skill ontologies)、與行為本體 (behavior ontologies)等。這些本體是用來當作共同的參考系統，如果欠缺共同的參考系統，使用者往往不容易以自然語言來完整與正確的表現自己的需求或能力。再來是利用本體語言如 DAML+OIL 或 RDF/RDFS 等，來註解 (annotate)求才者與求職者的文件，以便機器可以閱讀供需雙方的資料，進一步作電子化的工作媒合。

Bourse, Harzallah, Leclère, 與 Trichet (2002) 所提出的 COMMONCV 計畫，可以簡稱為履歷表上的能力(competencies on curriculum vitae)，主要目的是在塑模履歷表上的能力。其作法是先建立領域的本體，再利用這些本體作為基礎，作為註解履歷表或工

作職位需求的參考系統。Bourse 等人也實做了一個系統 (Canadian Common CV, 2007)，可以讓使用者在表單化的欄位中，輸入相關的資料後產生一份履歷表。但是在檢視實際的系統中發現，該系統可能是以所建立的本體作為基礎，使用者可以針對不同的資料項目，以下拉式選單來輸入相關的資料，再以正規的語言來加以塑模。整個應用只是用來產生標準化的履歷表而已，只能稱之為履歷表產生器，雖然可以延伸其應用至語意式的工作媒合，但是在報告中並未提及媒合的方法，在實際的系統中也尚未看到語意式的媒合機制。

二、基於能力陳述比對的工作媒合

能力(competency)是指個人在執行任務或從事某一項工作時，所需要具備的知識 (knowledge)、技能 (skill)與態度 (aptitude)。因此，能力也常常被稱為 KSA。能力是以「動詞+受詞+條件」的句型來陳述 (Mansfield & Mitchell, 1996)，動詞部份是一個及物動詞，通常被稱為行動動詞 (action verbs)，具有行動、表現與執行的涵義，也就是能夠透過行動來展現個人所擁有的能力。因此，行動動詞也被稱為能力動詞 (competency verbs)，常被使用在能力標準系統或職業能力的描述、也常被應用在學科基本能力、教學目標、課程設計與學習評量上、還有個人的專業學習檔案、求職履歷表與公司的徵才啟事等 (Michelin Career Center, 2004)；受詞部份則是能力動詞表現的對象，都是以名詞或名詞組的形態出現；條件部份則是能力表現的環境。如果沒有特定的環境可以省略，例如 write a program 就是一個能力的陳述。中文的能力陳述可以是「動詞+名詞」的型式，也可以是「名詞+動詞」的型式，例如「撰寫程式」與「程式撰寫」，都是能力的陳述

句 (Ven & Chuang, 2007)。

前述的能力陳述是一種標準的能力陳述方式，但是目前的人力資源網站都是以開放的方式，讓求才者來描述工作職位所需要的能力，也讓求職者來描述自己所擁有的能力。因此，整個描述是屬於開放式的自然語言表示，未必會遵循標準的能力陳述方式，即使使用者採用了標準的能力陳述，人力資源網站也未考慮能力動詞的媒合，僅考慮名詞部份的媒合，也就是只針對使用的工具作媒合，例如SQL Server、Java與PHP等。因此，Pan與Farrell (2007)認為一個工作媒合系統應該要能計算能力陳述之間的語意相似度，並且要有語意媒合的功能，他們替IBM公司的人力資源部門發展了一個語意式的工作媒合系統。以IBM公司的專長分類系統作為徵求人才的基礎，該專長系統中有10667個能力陳述，每一個能力陳述都是由18個行動動詞之一作為開頭，例如Advise、Architect、Code與Design等等。但是求職者極有可能使用其他的動詞來描述自己所擁有的技能，因此，在技能媒合的時候，必須排除以關鍵字媒合的方式，而應改採語意媒合的方式來進行工作媒合。Pan與Farrell根據Lin (1998)的語意相似度的定義，即

$$Sim(A,B)=\frac{\log P(common(A,B))}{\log P(description(A,B))}$$

其中*P(common(A,B))*為A與B之間的共同性的機率，而*P(description (A,B))*則為完全描述A與B的機率。在共同性的決定方面，並不是由A與B之間所共有的成份來決定，也和該成份出現在語料庫中的頻率毫無關聯，因為前者可能導致計算所得的相似度很高，但是在實際的語意上卻有不相似的錯誤情況出現，例如：

1. Advise Business Knowledge of CAD functionality for FEM.

2. Advise on Business Knowledge of Process for FEM.

第一個技能陳述是對於IBM公司的有限元素塑模軟體 (Finite Element Modeller；FEM)中CAD功能所需要的企業知識作建議，第二個技能陳述則是對於IBM公司的FEM的處理程序所需要的企業知識作建議。這兩個技能陳述中有許多共享的字，即Advise Business Knowledge of --- for FEM。以共享字取向所計算出來的語意相似度很高，但是以人來判斷，其相似度卻變得很低，主要的原因是因為語意角色 (semantic role)不同所導致，即CAD Functionality與Process是兩個不同的語意角色，由於語意角色的不同，也使得整個能力陳述的語意變得不同。為了萃取語意角色，Pan與Farrell (2007)發展了一個語意角色剖析器，可以從自然語言剖析器剖析過的語法樹中，萃取出18個行動動詞帶頭的技能陳述的語意角色樣板(semantic role pattern)，隨後剖析器由IBM公司的專長分類系統中隨機選擇了75對技能陳述，每一對技能陳述都有相同的語意角色，以三位評分者來作一致性檢驗，結果得到檢驗值0.81。也就是說，該語意角色剖析器的精確度為0.81。為了證明以語意角色的相似度來進行媒合優於單純的共享字取向的媒合方式，Pan與Farrell (2007)計算了兩者的相似度，前者的精確度為0.80，而後者的精確度只有0.71，顯示以語意角色相似度的媒合方式較佳。

Pan與Farrell (2007)所提出的能力陳述之語意相似度計算, 係根據Lin (1998)所提出的理論基礎，也就是利用兩個物件所擁有的共同資訊量，作為語意相似度的衡量基準。其共同資訊量不是以兩個能力陳述的共同成份來衡量，而是以兩者的語意角色來衡量，確實達到了語

意的層次。但是IBM公司的專長分類系統中只有18個能力動詞，當使用者以其他的能力動詞來描述能力時，兩者之間如何計算語意相似度，應該是一項極為重要的實務。例如建立資料庫 (build a database)、建構資料庫 (construct a database)、發展資料庫 (develop a database)、設立資料庫 (setup a database)等能力陳述句，其語意角色都同為資料庫 (database)，但是能力動詞卻完全不同。如果由語意的觀點來看，這些能力動詞存在有極大的相似度，這種相似度的計算方式並未在報告中提及，在實務的應用上應該有一個機制來衡量能力動詞之間相似度的計算。

三、基於 WordNet 比對的工作媒合

儘管全球資訊網上有許多的人力資源網站，求才者與求職者都可以將資料張貼上網，也可以用瀏覽的方式逐筆查詢資料，或是利用搜尋引擎進行關鍵字的搜尋。但是就缺乏交互參考的功能，使得工作媒合的效率與效能無法提升。印尼政府考量經濟發展的狀況與電腦普及率的問題，極需要以自動化與彈性的方式，來協助企業界招募適合的人才。因此，首先設立了一個中央控制型態的大型資料庫，可以讓各省與各地區的求才者與求職者進行工作媒合。但是卻有諸多的問題發生，例如因為資訊基礎建設不足所造成的處理延遲、與關鍵字搜尋所造成效能不彰與可信度存疑等。印尼政府於是將集中式的系統改變成分散式的模式，工作媒合的作業也下放至各省或各地區，使得資料的來源變得更為容易與多樣。在進行媒合作業時，可以利用點對點(peer-to-peer)的方式來提升硬體的效能，但是因為各個地區的資料太過分歧，造成跨區的工作媒合發生困難。甚至不同組織所使用的術語

相同，但是其資料內涵卻有相異的概念發生，導致嚴重的溝通問題與媒合問題。因此 Wicaksana 和 Yetongnon (2006)建議採用語意媒合的方式，其語意相似度的計算可以根據 WordNet 本體。由於 WordNet 是一種英文自然語言的語意詞典，其內部將具有相似意義的英文字詞組織起來成為相似詞集，每一個字詞又有上下位詞的階層架構關係。因此，WordNet 實際上就是一個本體，利用這個本體就可以發現與某個字詞具有相似意義的其他字詞，其字詞之間的語意相似度也就可以利用下列公式來計算：

$$Similarity = max\left|\frac{(2 * depth(LCS(a,b)))}{(length(a,b) + 2 * depth(LCS(a,b)))}\right|$$

此處的 *length(a,b)*是從概念 *a* 到概念 *b* 的路徑數目，*depth(LCS(a,b))* 是從概念 *a* 與概念 *b* 到的根節點間共同的路徑數目。表 3-1 是一個基於 WordNet 的兩個概念之間語意相似度計算的範例。例如員工(employee)和資源 (resources)有比較低的相似度為 0.267，但員工(employee)和人員 (person)具有比較高的相似度為 0.833。

表 3-1 基於 WordNet 的概念相似度

概念-概念	相似度	概念-概念	相似度
employee - resources	0.267	employee - person	0.833
skill - education	0.750	skill - background	0.750
hardware - hardware	1.000	hardware - PC	0.750
testing – quality control	0.889	testing - check	0.880

資料來源：Wicaksana & Yetongnon(2006)

Wicaksana 和 Yetongnon (2006)的工作媒合系統採用語意式的媒

合，其相似度的計算是基於 WordNet。但是因為 WordNet 是一種英文自然語言的詞典，可以稱之為一種通用性的本體，而非特定領域的本體。因此，在計算兩個概念之間的語意相似度時，可能會碰到一詞多義的狀況，也就是會有多條的路徑長度。但是其計算公式是取兩個概念之間的最大相似度，而不管其語意是否真正具有領域的相似度，其結果可能會產生重大的偏差；另外以 WordNet 為依據也可能造成無法計算的結果，因為許多領域的實例可能不會出現在 WordNet 之中。例如 C++、VB (Visual Basic)、PHP、Ruby on Rails 等，如果要計算這四種電腦語言的語意相似度時，就無法達成。而在工作市場上，要求具備前述電腦語言能力的工作廣告、或是具備這些電腦語言能力的履歷表都非常的多。因此，基於 WordNet 來計算語意相似度，比較適合應用在自然語言的處理上，而比較不適合應用在工作媒合系統上。

另外，現在的工作市場大部份要求應徵者具備多重能力 (Surakka, 2005；Ven & Chuang, 2007；溫瑞烘、莊謙本，2007)，各種能力之間也可能會有不同的重要性，也就是應該有不同的加權值。例如一般能力與技術能力可能有不同的加權，技術能力的項目也可能有不同的加權。Wicaksana 和 Yetongnon (2006)所發展的語意式工作媒合系統並未作這樣的考量，可能會影響到工作媒合系統的效能。

四、基於結構模式比對的工作媒合

目前大部份的人力資源網站都會提供制式的表單，讓求才者與求職者輸入工作需求或個人的專業檔案。這種輸入表單都會有不同的欄位，例如求職者的輸入表單可能有：個人基本資料、工作經驗與專業

技能等欄位；求才者的輸入表單可能有：企業的基本資料、工作描述、工作經驗與專業技能等欄位，所有輸入的資料大多儲存在關聯式的資料庫中。當使用者提出查詢時，就會將所輸入的查詢字串轉換成 SQL 命令，並從資料庫中擷取符合 SQL 命令的資料，這和傳統的文件擷取或資訊擷取的功能是相同的。但是可能因為兩種原因影響到查詢的結果，一是使用者都是用自己的模式輸入文字資料，並沒有利用統一的字彙。因此，在查詢時可能因為關鍵字的選擇不當，以致無法查出較佳的結果；另外一個原因是使用者未必會在每一個欄位中都輸入資料，也可能未按照主題來輸入，這都可能影響查詢結果的精確性。一個理想的工作媒合系統應該可以根據一個工作職位需求的描述，來取出條件接近的求職者的履歷表，並且按照匹配程度由高至低排序，讓求才者可以很方便的進一步選擇面談的對象。同樣，一個理想的工作媒合系統也應該可以根據一個求職者的履歷表，擷取一系列條件接近的工作職位，讓求職者可以選擇較佳的工作。為了達到這種功能，一個理想的工作媒合系統就應該提供多種媒合的方法，可以由資料庫中擷取適合的履歷表或工作職位的資料。

基於前述的概念，Yi, Allan, 與 Croft (2007)提出了數種資訊搜尋的方法，他們以 R 代表履歷表的資料集，J 代表工作職位的資料集，另外有過去匹配成功的履歷表與工作職位需求的資料集<r, j>，來當做訓練資料集。工作媒合系統要由資料庫中擷取與工作職位 j 相關的所有履歷表，或擷取與履歷表 r 相關的所有工作職位，以前者而言，所提供的搜尋方法如下：

1、單純語言模式 (simple language modeling approach；SLM)：去除工作職位需求<j>所有的結構，也就是去除所有的欄位描述，連接

所有欄位的資料成為一個長字串，稱為工作職位紀錄。同理，也將每一筆履歷表 R 變成一個長字串，稱為履歷表紀錄。再將此兩種紀錄作為查詢的基礎，以模糊查詢方式，也就是部份字串媒合的方式，取出所有相關的工作職位紀錄或履歷表紀錄。這種查詢方式就是傳統的字串媒合式查詢，也就是關鍵字查詢，其查詢效率較低。最大的缺點是完全沒有查詢延伸的功能，例如紀錄中要求具備 Visual Basic 的能力，但以 VB 來查詢時就無法取得結果。

2、真實關聯模式 (true relevance model；TRM)：同樣將工作職位的所有欄位資料壓平成為一個工作紀錄的字串，再利用該筆工作紀錄來查詢先前配對成功的訓練資料集<r, j>，取出一系列相關的工作職位的資料 j，再取出所有與 j 配對的履歷表 r，利用 r 來建立一個關聯模式，再利用此關聯模式來查詢所有的履歷表紀錄。同樣的道理，也將每一個履歷表資料壓平成為一個履歷表紀錄字串，再利用該筆履歷表來查詢先前配對成功的訓練資料集<r, j>，取出一系列相關的履歷表 r，再取出所有與 r 配對的工作職位資料 j，利用 j 來建立一個關聯模式，再利用此關聯模式來查詢所有的工作職位紀錄。這種真實關聯模式的查詢是單純語言模式查詢的延伸，係利用過去配對成功的資料集當作訓練資料，例如<r, j>中有一筆紀錄為<Visual Basic, VB>，則會建立 Visual Basic 與 VB 的關聯，使得以 Visual Basic 作查詢時，還會延伸以 VB 作查詢。同理 VB 也會以 Visual Basic 作延伸查詢。

3、結構化關聯模式 (structured relevance model；SRM)：前述的兩種查詢模式是先去除所有的攔位描述，形成單一的字串後再以字串媒合方式作查詢。結構化關聯模式是以 j 的每一個欄位來查詢 J 的對應欄位，將查詢所得的資料依欄位建立關聯模式，再利用此

不同欄位建立的關聯模式來查詢 R 的對應欄位。結構化關聯模式的基本觀念在於「一個欄位的資料可能由其他的欄位推導出來」，例如在工作職稱的欄位中是資料庫管理師，則利用過去配對成功的經驗，適當的候選人應該具有 SQL Server 或 MySQL 的能力。

在初步的驗證中，SLM 的表現最差，因為 SLM 就如同傳統的關鍵字搜尋；TRM 的表現次之，雖然在查詢方法上類似於關鍵字搜尋，卻打破了欄位結構的限制；SRM 的表現最佳，因為 SRM 除了最少可以達到 TRM 的效能之外，還利用已經完成的<r, j>配對來建立關聯模式，進一步作延伸的查詢。

雖然 Yi, Allan, 與 Croft (2007)所提出的工作媒合系統有三種不同的查詢方式，但是基本上還是屬於關鍵字查詢的範疇，其查詢的精確度往往會受到關鍵字選擇的影響。而在查詢的處理上，以字串媒合的方式會使得查詢的效率大打折扣。但是真實關聯模式與結構化關聯模式，都利用過去成功的配對資料作為延伸查詢的依據，確實可以最大化查詢的結果，而第三種的結構化關聯模式查詢可以設定不同欄位的加權值，也可以達到篩選的功能。

五、基於補償式比對的工作媒合

人力資源管理是知識密集公司的關鍵成功因素，其中最基本而且最重要的任務，就是要適才適所。為了達到這個目的，許多公司已經開始使用技能管理系統 (skill management system)，這是一種典型的資料庫查詢，允許搜尋具備某一種特定技能的人員，也就是使用關鍵

字查詢的方式。但是關鍵字查詢往往無法反應匹配的精確度，因此，在實務應用上，人力資源主管往往傾向於只需要找到大致符合資格的人員，以便進一步透過面談與其他雇用程序來招攬適合的人才。為了達成這樣的功能，Sure, Maedche, 與 Staab (2000)發展了一個工作媒合系統 ProPer，可以在技能需求與技能供給之間進行大致的媒合，其架構如圖 3-1 所示。申請者透過網際網路或企業內部網路送出申請檔案或專業檔案，儲存在專業檔案資料庫之中，再進行媒合並把結果送回給申請者。

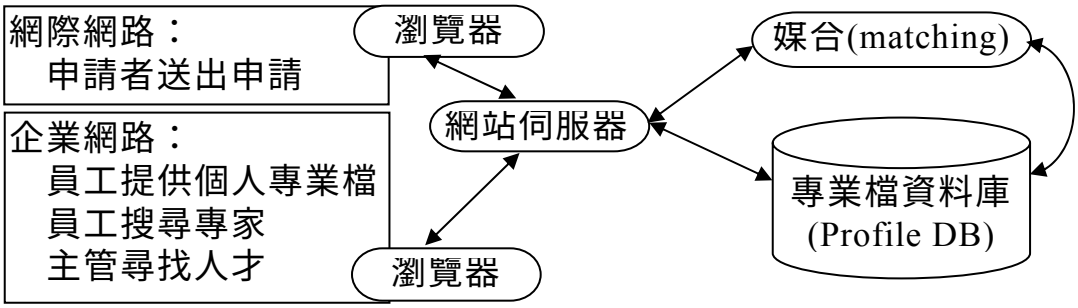

圖 3-1 ProPer 的系統架構（Sure, Maedche, & Staab (2000))

圖 3-1 的系統架構中最重要的成份是專業檔案資料庫與媒合程式，專業檔案資料庫 P_{DB} (profile database)由申請者 A (applicant)、員工 E (employee)、與需求 R (requirement)所共同組成，即：

$$P_{DB} = A \cup E \cup R = \{p_i \mid i = 1, m\}$$

$$A \cap E = \emptyset \quad \wedge \quad A \cap R = \emptyset \quad \wedge \quad E \cap R = \emptyset$$

$$p_i^T = (p_{i,1}, p_{i,2}, \ldots, p_{i,n})$$

前述的三個式子中，第一個式子表示資料庫是由 m 個專業檔案所組成。第二個式子表示申請者 A、員工 E、與需求 R 三者之間是無相互關聯的。第三個式子表示每一個專業檔案是由 n 個能力項目所組成。

當一個雇主在招募人才時，可能會根據工作職位的性質來考量兩種不同的媒合方法。一種稱為非補償式方法 (non-compensatory method)，另一種稱為補償式方法 (compensatory method)。前者是一種二分法，個人所有的專業能力必須等於或大於工作上的需求，只要有一項不合格，就被認為不具資格。這種方法適用於高度專業的職業，並且所需要的專業能力項目不多者；補償式的方法則允許能力不足者或缺乏某些能力者，也有被選擇的機會，希望在眾多申請者之中，將條件較佳者依序列出，以便進一步作後續處理。

在技能程度的分級方面，ProPer 使用 0 表示沒有任何知識、1 表示初學程度、2 表示普通程度、3 表示專家程度。除此之外，Proper 還可以考慮不同技能項目的加權值，其加權值可以使用矩陣 W 表示如下，其中的 $w_1 \ldots w_n$ 為不同的加權值，例如用 0 表示無加權、1 表示不重要、2 表示重要、3 表示極重要等。

$$W = \begin{Bmatrix} w_1 & 0 & \ldots & 0 \\ 0 & w_2 & \ldots & \\ \ldots & \ldots & \ldots & \ldots \\ 0 & 0 & \ldots & w_n \end{Bmatrix}$$

有了不同的加權之後，就可以按照下列公式，來計算工作需求所需的能力 p_i 與應徵者所具備的能力 p_j 之間的補償式的媒合結果 M_c。

$$M_C(p_i, p_j) = \frac{p_i^T \times (W * p_j)}{p_j^T \times (W * p_j)}; p_i, p_j \in P_{DB}$$

舉例來說，假設有一個工作的能力需求如下：Windows 2000 需要初學的程度、Java 需要專家的程度、JavaScript 需要普通的程度、Visual Basic 需要初學的程度，按照能力的程度分級可以建立一個專業檔案

p_1，如下所示：

$$\left.\begin{array}{ll}\text{Windows 2000} & \text{(Beginner)}\\ \text{Java} & \text{(Expert)}\\ \text{JavaScript} & \text{(Intermediate)}\\ \text{Visual Basic} & \text{(Beginner)}\end{array}\right\} \quad p_1=\begin{pmatrix}1\\3\\2\\1\end{pmatrix}$$

假設有一位申請者的專業能力如下，所建立的專業能力檔案為 p_2。

$$\left.\begin{array}{ll}\text{Windows 2000} & \text{(None)}\\ \text{Java} & \text{(Intermediate)}\\ \text{JavaScript} & \text{(Expert)}\\ \text{Visual Basic} & \text{(None)}\end{array}\right\} \quad p_2=\begin{pmatrix}0\\2\\3\\0\end{pmatrix}$$

而雇主所需要的技能項目及其加權矩陣如下所示：

$$\left.\begin{array}{ll}\text{Windows 2000} & \text{(unimportant)}\\ \text{Java} & \text{(very important)}\\ \text{JavaScript} & \text{(important)}\\ \text{Visual Basic} & \text{(important)}\end{array}\right\} \quad W=\begin{pmatrix}1&0&0&0\\0&3&0&0\\0&0&2&0\\0&0&0&2\end{pmatrix}$$

因此，補償式的媒合相似度的計算如下：

$$M_C(p_2,p_1)=\frac{0*1*1+2*3*3+3*2*2+0*1*2}{1^2*1+3^2*3+2^2*2+1^2*2}=79\%=0.79$$

Sure, Maedche, 與 Staab (2000)所提出的工作媒合系統 ProPer，允許在多重技能之間設定不同的加權，也允許能力達不到要求的申請者成為候選人。但是，在計算補償式媒合相似度時，申請者所具備的技能與雇主所要求的技能之間，必須是一對一的對應，不允許用其他技能來取代。例如在前述的範例中，雇主需要四種不同的技能：Windows 2000 (Beginner)、Java (Expert)、JavaScript (Intermediate)、

與 Visual Basic (Beginner)，但是申請者缺乏 Windows 2000 與 Visual Basic 的技能，因此，這兩項技能的媒合分數為 0。在實務應用上，許多技能之間可能有繼承的關係，具有極大的相似度。例如 C、C++、Visual C++，或 Windows 2000、Windows XP、Windows 98 等，都具有極大的相似度。如果缺乏不同技能名稱的相似度計算機制，也就是只利用表面的名稱作媒合，而不以內涵來作媒合，該工作媒合系統仍然不完整。

六、基於多重技能比對的工作媒合

利用網際網路來進行人才招募已經是資訊社會的一種趨勢，但是大部份的人力資源網站仍然停留在關鍵字媒合的層次，無法發現求才廣告與求職履歷表之間技能的相似度。主要的原因在於傳統的媒合系統只重視表面語法層次的媒合，難以深入瞭解兩者之間的語意概念，結果導致媒合效果不佳。

例如有一個求才者希望徵求 Delphi 程式設計師，另外有一位求職者具有 Visual C++與 Pascal 的程式設計經驗。如果只進行表面的媒合，這兩者之間是完全沒有交集的，顯示這個求職者完全不適任該項工作。但是深入一層來看，這個求職者已經有了 Pascal 的實務經驗，也有物件導向程式語言 Visual C++的經驗。以程式設計的觀點而言，這一位程式設計師可能只需要很短的時間，就能以 Delphi 來設計程式。也就是說，Delphi 與 Visual C++和 Pascal 之間是具有相似度的，Delphi 與 Visual C++都是物件導向程式語言，在程式設計的內涵上具有一定的相似度。例如類別、繼承、封裝、動態連結等，在任何一種

物件導向程式語言上都是類似的；Delphi 與 Pascal 也有語法繼承之間的關係，因為 Delphi 係由 Pascal 演進而來，所以這個求職者與求才者在技能供需之間具有某種程度的相似。一個工作媒合系統如果可以按照相似度的高低列出所有的求職者或求才者，必定使得工作媒合系統的效能大大提升。

每一個工作職位的需求或履歷表的內容，大致可以分成兩個部份，一部份是一般性的資料，例如年齡、教育程度、薪資、與地理位置等。這些資料大都是數值型態、列舉型態、或字串型態。其媒合方式可以由數值或列舉型態的距離計算出來；另外一個部份是能力的資料，又可以分成軟性技能 (soft skill)與硬性技能 (hard skill)。前者就是一般能力，也就是概念性的能力，例如物件導向、關聯式資料庫、與團隊合作等；後者就是技術技能 (technical skill)，也就是工具能力，例如 Java、PHP、SQL Server 等。但是技能部分的媒合方式就無法由單純的距離來計算相似度，最好的辦法就是利用技能本體來指定技能的加權值，再來計算技能之間的語意相似度。

Mohaghegh 與 Reza Razzazi (2004)建議了一個應用技能本體來進行工作媒合系統的架構，如圖 3-2 所示，其主要成分描述如下：

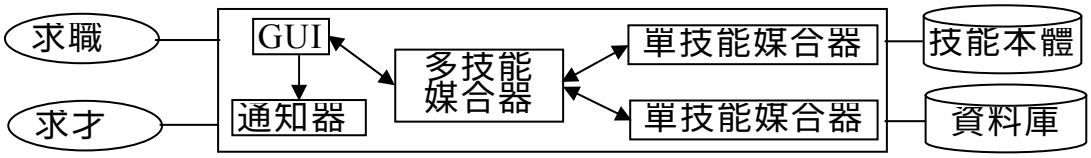

圖 3-2 多技能媒合系統架構(資料來源：Mohaghegh & Reza Razzazi , 2004))

1、技能本體 (skill ontology)：技能本體的範例如圖 3-3 所示，每一個節點代表一項軟性技能或硬性技能。由於許多技能之間無法截然劃分。因此，技能本體採用多重繼承的觀念，例如 Object Pascal 是一種物件導向語言，是由 Pascal 演化而來。因此，Object Pascal 就分別繼承了 O. O.與 Pascal 的特性。本體中的每一個節點內部有一個加權值，代表技能本身的貢獻度。每一個邊上也有一個加權值，代表父技能對子技能的貢獻度。節點本身的加權與所有繼承邊的加權總和為 1, 例如 C++本身的貢獻為 0.45, O. O.貢獻給 C++的加權為 0.3，另外 C 貢獻給 C++的加權為 0.25，三者之和為 1。這些節點與邊的加權值係由領域的專家所決定。

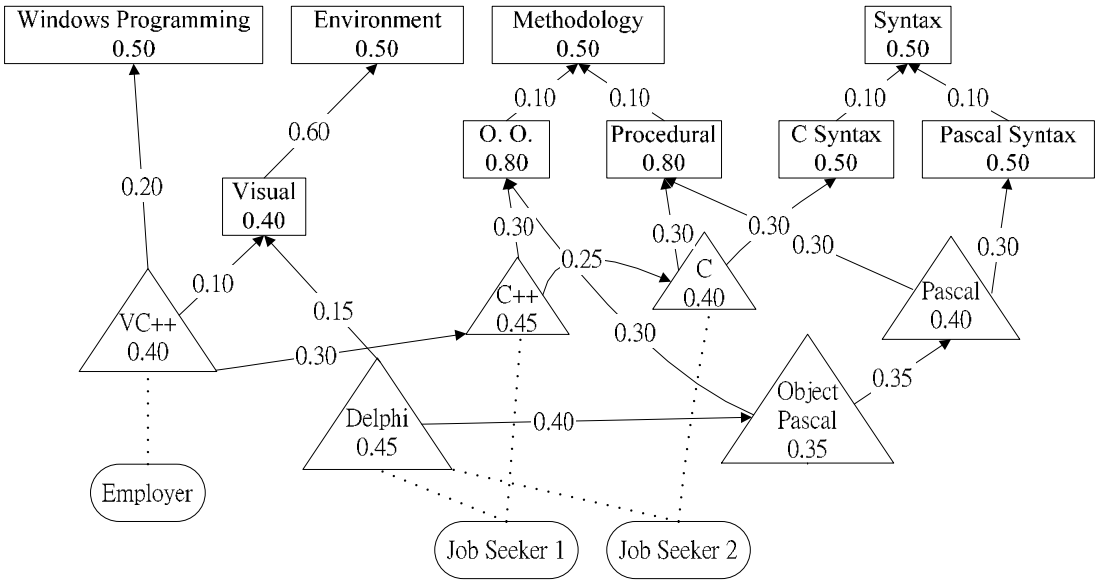

圖3-3 以貢獻度表示的技能本體範例（資料來源:Mohaghegh & Reza Razzazi (2004)）

2、資料庫：儲存求職者與求才者的資料。

3、GUI 圖形介面：提供求職者與求才者輸入相關資料或進行查詢。

4、單技能媒合器 (single-skill matchmaker)：負責單一技能之間的相似度計算。當兩個節點之間有 isa 的繼承關係時，子節點與父節點

的語意相似度就是父節點對子節點的貢獻度，也就是繼承邊的加權值。而父節點與子節點的語意相似度為子節點的加權乘以繼承邊的加權。因此，父節點與子節點的語意相似度會大於子節點與父節點的語意相似度。以圖 3-3 為例，假如要計算 C++與 C 的相似度時，因為 C++是繼承自 C，因此，C++與 C 的相似度是 C 貢獻給 C++的貢獻度 0.30；反過來說，假如要計算 C 與 C++的相似度，則利用 C++節點的加權值 0.45 乘上繼承邊的加權值 0.25，也就是 0.1125。因此，C++與 C 的相似度為 0.30，而 C 與 C++的相似度為 0.1125，前者相似度較高的原因在於：擁有 C++技能者必定也具備 C 的能力，但是具備 C 能力者未必瞭解 C++。

5、多技能媒合器 (multi-skill matchmaker)：負責將多項的技能加以劃分之後，再依次呼叫單一技能媒合器做相似度的計算。例如要計算一個求才廣告與所有履歷表的相似度時，多技能媒合器就必須要將求才廣告所需要的技能加以一一分離出來，也要將每一份履歷表所擁有的技能一一分離出來，再將這兩種技能一一配對並呼叫單一技能媒合器進行相似度的計算, 並找出相似度較高的配對。

6、通知器 (notifier)：將媒合結果按照相似度總分排序，並通知使用者，使用者可以選擇達到某一個門檻值的資料，並通知應徵者進一步的面試。

在計算語意相似度時，藉著每一個節點與邊的加權，就可以計算每一個求才廣告與任何一個求職履歷表之間的語意相似度分數。假設一個求才廣告與一個求職履歷表分別對應到節點 n_i 與 n_j，如果 i 等於 j，則表示求才者所要求的技能和求職者所具備的技能完全相同，應該獲得最高的語意相似度分數 1。如果 i 不等於 j，則可以由節點 i 開

始，利用先廣後深搜尋 (breadth first search；BFS)或先深後廣搜尋 (depth first search；DFS)，來依次計算所經過路徑的路徑分數 (path score)，這必須藉由每一個節點與邊的加權來計算。當兩個節點之間有繼承關係時，子節點與父節點的語意相似度就是繼承邊的加權值，而父節點與子節點的語意相似度為子節點的加權乘以邊的加權。

以圖 3-3 為例，雇主 (Employer)要求 VC++的能力，而求職者 1 (Job Seeker 1)擁有 Delphi 與 C++的能力，求職者 2 (Job Seeker 2)擁有 Delphi 與 C 的能力，雇主(Employer)與求職者 1 (Job Seeker 1)及雇主與求職者 2 (Job Seeker 2)之間的語意相似度計算方式如下：

Job Seeker 1：

C++對 VC++的貢獻度為 0.30，也就是 VC++對 C++的語意相似度為 0.30；Delphi 對 VC++的貢獻度為 0.15+0.4*0.1=0.19，也就是 VC++對 Delphi 的語意相似度為 0.19；故 Job Seeker 1 與 Employer 的相似度為 0.30+0.19=0.49

Job Seeker 2：

Delphi 對 VC++的貢獻度為 0.15+0.4*0.1=0.19；C 對 VC++的貢獻度為 0.25*0.3=0.075；故 Job Seeker 2 與 Employer 的相似度為 0.19+0.075=0.265

整個相似度計算的演算法如表 3-2 所示，輸入的參數為雇主要求的一個技能，輸出是相似度達到某一個門檻的履歷表。

表 3-2 多重技能比對工作媒合系統的演算法

```
Algorithm Gen_Resume_List(Input Sub-advertisement A, Output List L) {
   Max_Priority_Queqe PQ
   Node N=A.Skill
   N.Path_Score=N.Weight
   PQ.Enqueue(N)
   Mark N as visited
   while (PQ is not empty) {
      N=PQ.Dequeue
      if (N is a hard skill node) then {
        for all resume R attach to N {
          R.Score=Total_Score(A, R, N.Path_Score))
          if R.Score>Score_Floor then L.Insert(R)
        } // for all resume
      } // if (N is a hard skill node)
   for all node M that has inheritances relationship I with N in the ontology {
      if N is not visited then {
          mark N as visited
          M.Path_Score=N.Path_Score*I.Weight
          If (M.Path_Score>Score_Floor) then PQ.Enqueue(M)
      } // if N is not visited
   } // for all node
  } // while
} // Algorithm
```

資料來源：Mohaghegh & Reza Razzazi (2004)

Mohaghegh 與 Reza Razzazi (2004)所建議的工作媒合系統中，技能本體是最重要的成份，該技能本體和其他媒合系統技能本體不同的地方，在於其概念或實例是可以採用多重繼承的觀念。圖 3-3 的技能本體中就有許多多重繼承的範例，由技術演進或產品發展的過程來看，這些多重繼承的範例事實上都有其根據；另外一個相異之處在於本體的每一個邊都指定有一個加權值，這是父節點對子節點的貢獻度，也就是相似度。以繼承的觀點來看，這種相似度的指定也是非常的合乎邏輯，但是如果要由領域專家來指定相似度，可能在實務上會遭遇到困難，應該朝向由其他方式自動找出此相似度；另外一個問題在於一般資訊的媒合，有時候求才者與求職者之間，也往往會考慮一般資訊的媒合，例如教育程度、工作經驗等。該媒合系統只是以技能的媒合為考量，也有不足之處，再來是該研究將所有的技能項目一視同仁, 無法設定不同的重要性加權值, 在實務應用上也可能不符需求。

七、基於最短途徑矩陣比對的工作媒合

Lv 和 Zhu (2006)指出，中國大陸已經有許多大學、企業與政府機關所設立的人力資源入口網站，提供基本的個人資料管理、求職、與求才的搜尋等，但是因為利用網際網路來招募人才的需求持續提升，這些傳統網站運作的結果產生了三個主要的缺點：

1、關鍵字查詢的缺點：網站所提供的查詢功能都是以關鍵字方式搜尋，無法確保回覆率與精確度，可能導致不正確或多餘的資料充斥。另外，以單一的關鍵字查詢往往顧此失彼，例如以職務名稱查詢時，就無法考慮重要的技能面向，查詢獲得的結果還要花費許多的時間進一步媒合或篩選。

2、遺失有潛力的資訊：由於利用關鍵字作媒合，當關鍵字選擇不適當時，可能會遺失具有潛力的資訊。例如有位雇主要徵求 JSP 的專家，有一位應徵者宣稱有 ASP 與 Java 經驗，另一位應徵者具備 VC++與 ASP.NET 的能力。如果以 JSP 當關鍵字來查詢，這兩位應徵者的資料都不會被搜尋到，但是由程式設計的觀點來看，這兩位應徵者都具有 JSP 的潛力，如果改以相似度取向的搜尋方式，這兩位應徵者的資料可能都會被列舉出來。

3、缺乏預選或篩選功能：當一個工作職位有許多人申請時，雇主或人力資源主管希望能夠很有效率的選擇較適合的求職者，再進一步通知合格的人員面談。同樣的道理，以求職者的觀點而言，也希望能夠應徵不同的工作機會，但是現有的搜尋機制中卻沒有篩選的功能，也無法列出配合程度的優先順序，求才者與求職者都無法知道那些人才或工作是比較適合的。

為了改善前述的缺點, Lv 和 Zhu (2006)提出了一個以技能本體為基礎的語意式工作媒合系統，其技能本體的建立係根據下列三個原則：

1、某個領域的技能本體可以由許多子本體 (sub-ontology)所組成，每一個子本體用來表示某一個領域的子題，例如圖 3-4 的本體是一個程式語言技能的子本體。

2、子本體是一個圖形結構，每一個節點可以代表一個技能概念，例如 Windows Programming、Database Concept，也可能是一個技能實例如 VC++、Delphi。

3、節點與節點之間可以建立兩種關係，如果是 isa 的繼承關係，則在邊的加權值可以設定最大相似度為 1，表示繼承的貢獻度。如果

是 partof 的部份整體關係，則邊的加權值應設定小於 1 且大於 0 的相似度，表示整體對於部份的貢獻度。

圖 3-4 是一個兼具 isa 與 part of 兩種關係的技能本體的範例，在 C++與 VC++之間有兩個邊相連，其中一個邊的加權值為 1.0，表示 VC++ isa C++，顯示 VC++係繼承自 C++，也就是 VC++與 C++的相似度為 1，換句話說，具有 C++能力者也必定具有 VC++的能力；另外一個邊的加權為 0.3，表示 C++ is partof VC++，顯示 C++只是部份的 VC++，也就是 C++與 VC++的相似度為 0.3，換句話說，具有 VC++能力者未必完全具有 C++的能力。

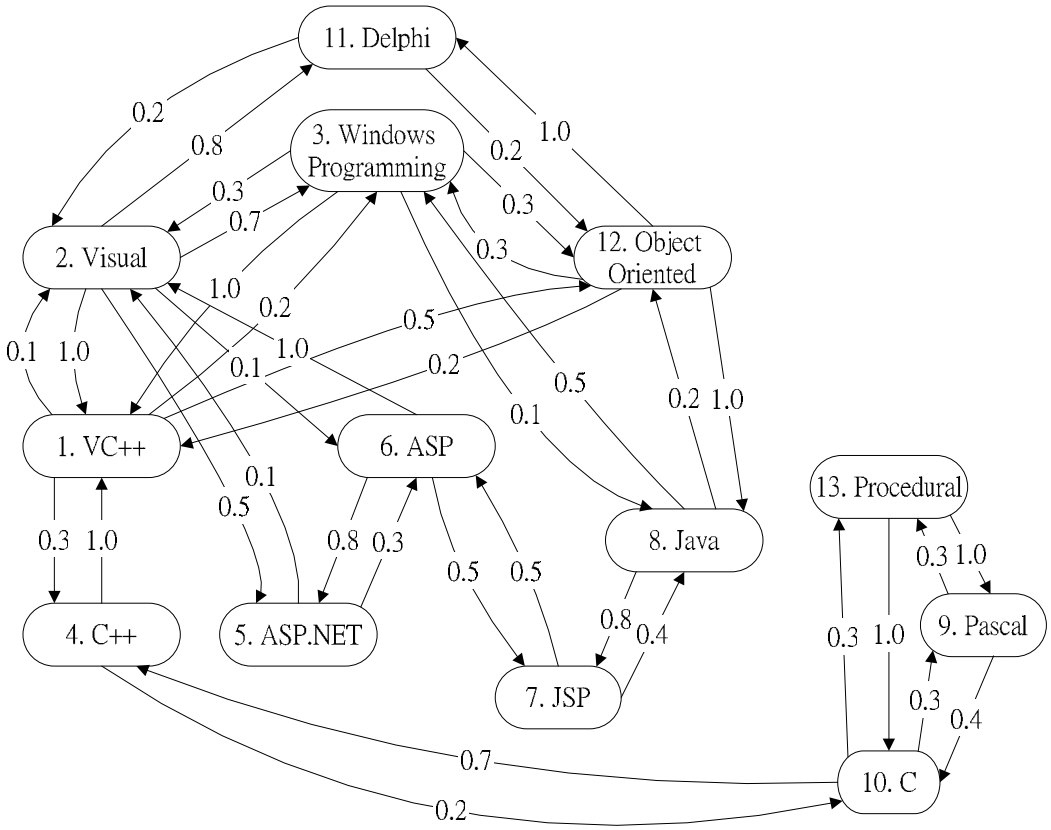

圖3-4 兼具isa與partof關係的技能本體範例（資料來源: Lv & Zhu (2006)）

有了技能本體之後，就可以根據該本體來作工作媒合，其方法是將整個技能本體當作是一個圖形結構，利用 Dijkstra's 的演算法(Dijkstra's algorithm)來求出此圖形結構的最短途徑矩陣，其演算法分成三個步驟，敘述如下：

1、前置處理：建立本體的鄰接矩陣資料表示法，就是取出每一個向量邊的開頭節點、終止節點與加權值，分別表示如表 3-3 的第一、第二與第三欄。

表 3-3 以鄰接矩陣表示本體資料

節點一	節點二	加權
1	1	0
1	2	0.1
…	…	…
13	10	1.0

資料來源：Lv & Zhu (2006)

2、單一技能媒合：將雇主的 n 個技能需求表示成 $R(r_1,r_2,\ldots,r_n)$，求職者的 m 個技能供應表示成 $S(s_1,s_2,\ldots,s_m)$，將 R 與 S 表示成為一個 $A=n*m$ 的相似度矩陣，如下所示，其中的 $sim(r_i,s_j)$是由表 3-3 而來，表示節點 i 到節點 j 的加權。

$$A=\begin{bmatrix} sim(r_1,s_1) & sim(r_1,s_2) & \cdots & sim(r_1,s_m) \\ sim(r_2,s_1) & sim(r_2,s_2) & \cdots & sim(r_2,s_m) \\ \cdots & \cdots & \cdots & \ddots \\ sim(r_n,s_1) & sim(r_n,s_2) & \ddots & sim(r_n,s_m) \end{bmatrix}$$

接著找出矩陣 A 中每一列的最小值，再組成矩陣 B，如下所示，第一行的 r_1，r_2，…，r_n 就是雇主的需求 $R(r_1,r_2,\ldots,r_n)$，第二行的 s_i，s_j，…，s_t 等，分別是相似度最小值的位置，最後將第三行的相似度值加總，就可以獲得 R 與 S 的相似度。

$$B=\begin{bmatrix} r_1 & s_i & sim_1 \\ r_2 & s_j & sim_2 \\ \cdots & \cdots & \cdots \\ r_n & s_t & sim_n \end{bmatrix}$$

3、優先次序媒合：雇主在招募人才時，往往會設定某些技能項目有較高的程度，例如在徵求 Java 程式設計師時，需要 Java 技能達到專家的程度，但 PHP 技能可能只需要初學的程度。反過來說，在徵求 PHP 程式設計師時，其優先次序正好相反，PHP 技能需要專家的程度，但 Java 技能可能只需要初學的程度。為了要因應不同的優先次序，應該要能夠設定不同技能項目的重要性加權，但必須合乎下列條件：

$$w_i > 0 (i = 1,2,\ldots,m) \text{ 且 } \sum_{i=1}^{m} w_i = 1$$

也就是所有項目的加權總和為 1，例如設定加權：初學為 0.1、普通為 0.2、佳為 0.5、極佳為 0.7。例如有一位雇主需要徵求 ASP.NET 能力-佳、C 語言能力-普通、網路管理能力-普通、SQL-出學，這四個能力的加權值可以形成一個加權值矩陣 $w=[0.5,0.2,0.1,0.2]$，然後再利用加權值矩陣 w 乘以相似度矩陣，就可以獲得媒合的分數。例如有兩個求職者在 ASP.NET、C、網路管理、SQL 等四種技能的相似度矩陣如下：

求職者一：$Similarity_1 = \begin{bmatrix} 0.78 \\ 0.3 \\ 0.9 \\ 0.8 \end{bmatrix}$、求職者二：$Similarity2 = \begin{bmatrix} 0.37 \\ 0.53 \\ 0.40 \\ 0.90 \end{bmatrix}$

再利用 $w*Similarity_1$ 與 $w*Similarity_2$，可以分別獲得 0.7 與 0.511，因此，求職者一比較適合該工作職位。

Lv 和 Zhu (2006)所提出的語意式工作媒合系統，係把技能本體當作是一個有向的圖形結構，並利用 Dijkstra 的演算法來計算技能之間的相似度，確實簡化了相似度的計算。但是最大的問題是在本體的兩種關係，如果是 isa 的繼承關係，則兩個技能之間的相似度可能為 1；如果是 part of 的部份整體關係，則兩個技能之間的相似度小於 1。按照 Lin (1998)對相似度的定義，兩個物件要完全雷同，其相似度才可能為 1，因此，本體中 isa 繼承關係的相似度絕對不可能達到 1。另外，觀察圖 3-4 的技能本體可以發現，兩個技能之間都有兩個有向的邊，分別代表 isa 與 part of 兩種關係，其中 isa 邊的相似度明顯大於 partof 邊的相似度，這兩個相似度應該如何決定，在文中並未提及，因此，在實務應用上可能會碰到障礙。

八、基於整合四種模式比對的工作媒合

Biesalski, Breiter, 與 Abecker (2005)在其應用研究中，提出了一個整合式的人力資源管理架構。該架構的核心是一個擁有七百個技能的技能本體，以及利用此本體作為人才雇用的應用系統。該系統可以利用四種不同的媒合方式，來計算求才者的技能需求與求職者的技能

供應之間的相似度。求才者可以針對不同的技能需求給予不同的加權，以顯示技能重要性的不同。每一種技能需求也可以要求精通的程度，例如初學 (beginner)、進階 (advanced)、專家 (expert)與教練 (trainer)等。相似度的值通常是界於 0 到 1 的數值，但是在計算補償相似度時，其數值可能會大於 1。底下是四種相似度計算的公式，其中 s 是求才者的技能需求，i 是求職者的技能供應，S 是技能需求的集合，I 是技能供應的集合，$weight(s)$是求才者技能需求 s 的加權，$sim_{value}(s,i)$是求才者技能需求與求職者技能供應程度之間的相似度，$|S|$是技能的總數。

1、正確媒合 (exact matching)：當一個求才者所要求的技能需求數目不多時，很可能要求求職者完全符合所訂定的技能需求，此時就可以使用正確媒合的方式，其計算公式如下：

$$sim_{exact}(S,I)=\frac{\sum_{s\in S,i\in I} weight(s)*sim_{value}(s,i)*sim_{skill}(s,i)}{|S|}$$

其中 $sim_{skill}(s,i)$的技能需求 s 與技能供應 i 要完全相同。當 i 的技能程度大於或等於 s 時，所獲得正確媒合的相似度為 1，否則為 0。公式中的分母是技能需求的總數，分子則是所有相同技能正確媒合的相似度乘以重要性加權的加總。

2、分類相似度 (taxonomic similarity)：當一個求才者所要求的技能需求數目很多時，使用正確媒合可能無法找到適合的人才。此時可以使用分類相似度來找出相似度較高的人才，但是先決條件是要有技能的分類目錄或技能本體，其計算公式如下：

$$sim_{taxonomic}(S,I)=\frac{\sum_{s\in S,i\in I} weight(s)*sim_{taxonomic}(s,i)}{|S|}$$

只要 $sim_{taxonomic}(s,i)$的技能需求 s 與技能供應 i 兩者在相同的分類項目之下，也就是本體的 isa 關係，不管兩者的技能程度如何，其相似度都是 1。如果 s 與 i 不在相同的分類目錄下，其相似度為 0。公式中的分母是技能需求的總數，分子則是所有技能的分類相似度乘以對應的重要性加權的加總。

3、比例媒合 (proportional matching)：雇主在考慮用人的標準時，也可能使用部份合格的概念，只要是技能項目相同，但是在技能程度上可能達不到需求，雇主可以考慮晉用之後再給予職前訓練。此時就可以使用比例相似度，其計算公式如下：

$$sim_{proportional}(S,I)=\frac{\sum_{s\in S,i\in I} weight(s)*sim_{proportional}(s,i)*sim_{skill}(s,i)}{|S|}$$

其中 $sim_{skill}(s,i)$的技能需求 s 與技能供應 i 要完全相同。當 i 的技能程度大於或等於 s 時，所獲得的比例相似度為 1，否則必須要根據技能程度之間的比例賦予比例相似度。以前述的四個技能程度為例，如果雇主要求 Java 的專家程度，但應徵者只有 Java 的進階程度，其比例相似度為 0.75、如果應徵者的 Java 程度為初學，其比例相似度為 0.5。公式中的分母是技能需求的總數，分子則是所有相同技能比例媒合的相似度乘以重要性加權的加總。

4、補償媒合 (compensatory matching)：補償媒合是比例媒合的相反，當應徵者的程度超過了雇主的要求時，雇主如果獲得這種人才，

就可以立即投入工作崗位。因此，雇主可以考慮按照超過的比例給予補償，對於不足的部份則給予懲罰，其計算公式如下：

$$sim_{compensatory}(S,I)=\frac{\sum_{s\in S,i\in I} weight(s)*sim_{compensatory}(s,i)*sim_{skill}(s,i)}{|S|}$$

其中 $sim_{skill}(s,i)$的技能需求 s 與技能供應 i 要完全相同。以前述的技能程度為例，假如要求一個 Java 的初學者，但卻來了一位 Java 的專家，則補償相似度為 1.5。反過來說，當要求一個 Java 的專家，但卻來了一位 Java 的初學者，則補償相似度為 0.5。公式中的分母是技能需求的總數，分子則是所有相同技能補償媒合的相似度乘以重要性加權的加總。

在前述的四種相似度計算中，分類相似度的計算是利用技能的分類來計算相似度，只要兩個技能在相同的分類項目之下，其相似度就是 1，否則為 0。這種計算方式可能造成極大的扭曲現象，以圖 3-5 為例，三種電腦語言技能 C++、PHP、與 Java 是歸類在 IT (information technology)之下，任何兩個技能的分類相似度都是 1，雖然電腦語言之間有其共通之處，但是任何兩種電腦語言的相似度都不可能達到 1；再以三種自然語言 English、French、與 Italian 等為例，他們都是歸類在 Language 之下，要說這些語言之間的相似度是 1 也是太過扭曲事實。

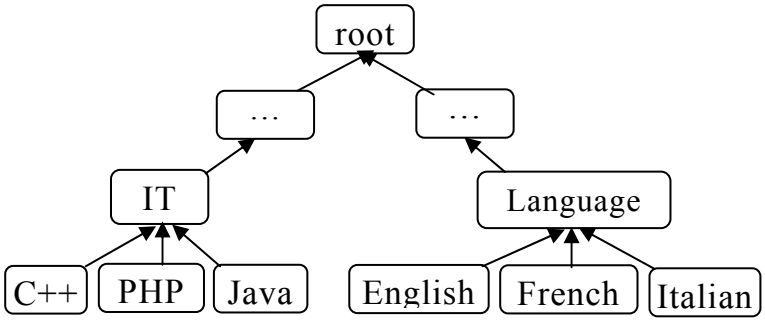

圖 3-5 提供分類比對的技能本體（資料來源：Biesalski, Breiter, & Abecker (2005))

表 3-4 四種相似度計算的範例比較

求才者	求職者	正確媒合	分類媒合	比例媒合	補償媒合
English (Beginner)	English (Expert)	1	1	1	1.5
C++ (Advanced)	PHP (Expert)	0	1	0	0
Java (Expert)	Java (Advanced)	0	1	0.75	0.75
French (Beginner)	French (Trainer)	1	1	1	1.75
平均值		0.5	1	0.6875	1

資料來源：Biesalski, Breiter, & Abecker (2005)

表 3-4 是這四種相似度計算的比較，其中的分類相似度係根據圖 3-5 的技能本體，並且假設求才者四種技能需求的重要性加權值都是 1。觀察求才者與求職者的技能匹配情形，在正確媒合方面，只要求才者與求職者的技能相同，不管雙方技能程度的差異，其媒合的結果都是 1，只要是技能名稱不同，其結果就是 0；在分類媒合方面，四種媒合的結果都是 1，如果對照四種技能的名稱，這種結果有可能會造成誤導，因為其中的 C++和 PHP 是兩種不同型態與目的的程式語言，只因為在同樣一個 IT 分類項目下，就認定相似度為 1，確實是

過度的高估；在比例媒合方面，只要是技能名稱不同，媒合的結果就是 0，又似乎是過度的低估，因為 C++與 PHP 同為程式語言的一種，至少有一點類似的地方，例如在敘述的語法與變數型態上；在補償媒合方面，不同技能名稱媒合的結果也是 0，但是可能因為過度補償的結果，使得平均值大於或等於 1，也可能造成誤導。

九、基於能力模式比對的工作媒合

傳統的工作媒合系統採用文字媒合的方式，也就是利用關鍵字的字串比對搜尋方式，其效能受到極大的限制。在相似度計算理論和語意網科技出現之後，越來越多的人力資源管理改採語意式的媒合方式。其主要的概念是延伸語法導向的內容到語意導向的內容，使得機器可以瞭解供需雙方的文件。

語意式的工作媒合系統的實作方法有兩種：一種是利用領域的本體，來註解求才者的工作描述與求職者的履歷表，使得雙方的文件都具有詮釋資料，再發展媒合系統來作自動化的媒合；另一種是利用領域的本體來建立雙方文件的語意索引，再進行媒合。

Yahiaoui, Boufaïda, 與 Prié (2006)利用前述的第一種概念，發展了一個語意註解與語意媒合的工作媒合系統，其架構如圖 3-6 所示，整個運作狀況係由使用者啟動。使用者可以是求職者也可以是求才者，首先將文件儲存到文件伺服器中，再透過註解介面，將文件伺服器中所儲存的文件加以註解，再經由呼叫媒合元件並基於 ER-本體 (electronic recruitment ontology)，進行表面語意式的媒合或能力本位

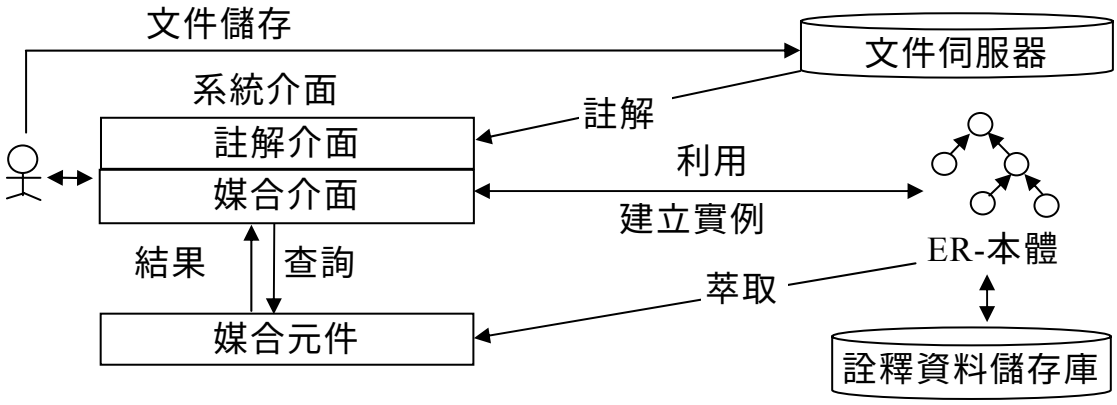

圖 3-6 語意註解與比對的系統架構(資料來源：Yahiaoui, Boufaïda, & Prié, 2006))

語意式的媒合，最後將媒合的結果送回給使用者。整個架構包括下列主要的成份：

1、ER 本體：電子化人才招募本體，是由數個相關的本體組成，用來註解求職者的履歷表與工作職位需求的相關資訊，所有用來註解的詮釋資料被儲存在詮釋資料儲存庫 (metadata repository)中。

2、文件伺服器 (document server)：用來儲存準備註解的履歷表與工作職位需求，文件的格式可以是 HTML 或 XML。

3、系統介面 (system interface)：包括註解介面 (annotation interface)與媒合介面 (matching interface)。前者可以讓一般使用者藉由 ER 本體來註解自己的文件，並將註解好的文件儲存在文件伺服器中；後者可以讓使用者提出查詢，使得求職者可以找到最適合自己能力的工作職位，也可以讓求才者找到最適配工作職位的人才。

4、媒合元件 (matching component)：用來解譯使用者的查詢命令，以便獲得文件的儲存位址 URI 與語意媒合的種類，然後開始計算語意媒合的係數，並可以利用表面語意媒合 (surface semantic matching)，來獲得表面語意媒合係數。或是透過能力本位語意媒合 (competency-based semantic matching)，來獲得能力本位語意媒

合係數，這些係數是一種百分比係數。媒合元件可以接受求職者與求才者的要求，如果使用者是一位求職者，媒合元件會以該求職者的履歷表為基礎，計算所有工作職位需求與此履歷表的語意媒合係數；如果使用者是一位求才者，媒合元件會以工作職位的需求為基礎，計算所有求職履歷表與此需求的語意媒合的係數，數值愈大表示相似度愈高，兩者之間也就愈適配。

在圖 3-6 的語意註解與媒合系統架構中，最重要的部份是 ER-本體，係由五個不同的本體組合而成，整體架構如圖 3-7 所示。其內容就是工作職位描述或履歷表中的重要內容，可以讓求職者與求才者使用相同的字彙與相同的參照基準，來描述個人所擁有的能力及工作職位的需求，使得自動化的媒合變得可行。詳細的本體內涵描述如下：

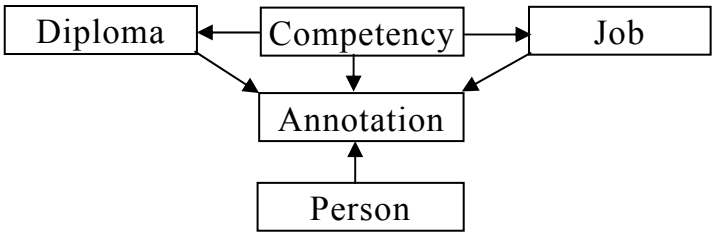

圖 3-7 ER 本體的架構 (資料來源：Yahiaoui, Boufaïda, & Prié (2006))

1、人員本體 (Person ontology)：只有一個概念 Person，用來描述供需雙方的個人基本資料，包括性別、年齡、服役情形、居住地與家庭狀態等屬性。
2、文憑本體 (Diploma ontology)：用來描述和文憑與訓練有關的概念，包括文憑種類、所屬領域與文憑參照系統等屬性。
3、工作本體 (Job ontology)：用來描述和工作經驗有關的概念，包括工作類別、領域與工作參照系統等屬性。

4、能力本體 (Competency ontology)：用來描述能力模式與能力物件的階層架構，如圖 3-8 所示，此能力本體中的行為能力和領域無關，但是科學與技術能力必定是領域相關，每一個節點資料有一個屬性 weight，是一個百分數的加權值，用來記錄該節點的主題對父主題的貢獻分數。

5、註解本體 (Annotation ontology)：註解本體分別取得個人資料、工作經驗、與能力檔案，以便進一步加以註解，其中的概念 Resource 以屬性 URI 記錄等待接受註解的文件，以屬性 type 記錄文件的型態為履歷表 CV (curriculum vitae)或是工作需求 OF (job offer)。

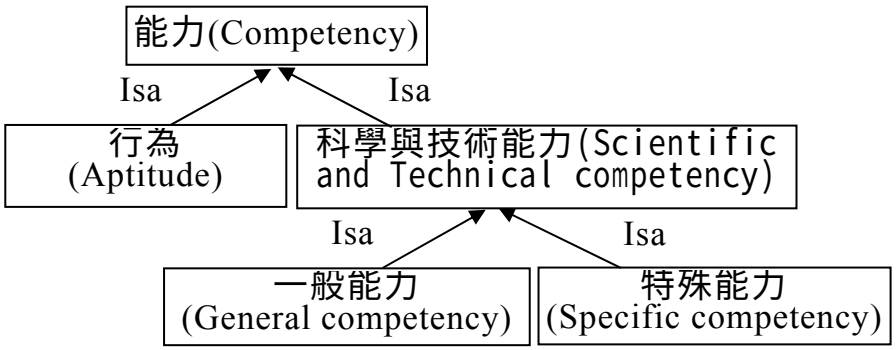

圖 3-8 能力模式 (資料來源：Yahiaoui, Boufaïda, & Prié (2006))

在前述的五個本體中，Diploma、Job 與 Competency 的實例是由系統管理人員所建立，Diploma 與 Job 兩種本體都有標準的參照系統，前者可以根據教育與訓練的制度或分類，後者則可以根據職業或行業的標準分類，因此可以很容易的將本體建立起來，而 Competency 本體則是領域的本體，必須靠領域專家來建立。至於另外兩個本體 Person 與 Annotation 的實例必須依靠使用者來建立，包括：

1、建立類別 Resource 的實例，用來描述要註解的文件。

2、建立類別 Person 的實例，用來描述求職者或求才者的個人資訊。

3、建立類別 JobExperience 的實例，用來描述求職者或求才者有關工作經驗的資訊。

4、建立類別 AcquiRequi 的實例，用來描述求職者所獲得的所有能力，或工作上的所有需求。

5、建立類別 Annotation 的實例，來連結應該註解的資源。

媒合元件所媒合的內容是能力 (competency)，此能力的模式如圖 3-8 所示，包括行為 (aptitude)、科學與技術能力 (scientific and technical competency)。前者是指個人屬性，例如誠實、正直等，可以根據標準化的行為分類來建立或識別；後者是知識與技術能力，是和特定的領域有關，又分為一般能力與特殊能力。一般能力是指一般性的知識，例如數學、物理學等。特殊能力還有四種不同的程度分級：基礎級 (basic)代號為 B，相當於精通程度為 20%、應用級 (application)為 A 或 50%、熟練級 (mastership) 為 M 或 70%、與專精級 (expert)為 E 或 90%。

Yahiaoui 等人的語意媒合系統提供兩種不同的媒合方式：

1、能力本位語意媒合：這種媒合方式是要媒合隱含在履歷表或工作需求表中的能力，依次取出求才者所需要的每一項能力，並搜尋求職者所獲得的能力，如果搜尋成功，其加權就被累加起來，如果搜尋失敗，則利用所建立的能力本體，來找出能力的對應位置，也把加權值累加起來，整個能力本位媒合的演算法如表 3-5 所示。

2、表面語意媒合 (superficial semantic matching)：表面語意媒合是比對供需雙方在某一個項目的內容是否存在，如果存在則獲得語意相似度為 1，如果不存在則相似度為 0。媒合的項目包括能力、文

憑、工作經驗、專業檔案與個人資訊等。在這四個項目上也可以設定不同的加權值，並且可以隨時調整，來反應不同的重要性，最後的匹配分數為計算所得加權除以總加權，再乘以 100，也就是說，其相似度是以百分數來表示。

表 3-5 能力本位媒合的演算法

```
Algorithm Competency_based_matching (R_OF, R_CV) {
  /* R_OF: 求才者需要的能力集                    */
  /* R_CV: 求職者所獲得的能力集 (履歷表) */
  tot_weight←0     /* 所需能力係數的總和 */
  calc_weight←0    /* 所獲能力係數的總和 */
  Extraction_Competencies (C_CV, C_OF) /* 取得所需能力與所獲能力 */
  FOREACH C_r∈C_OF REPEAT { /* 針對每一項所需能力執行迴路 */
   IF Aptitude(C_r) THEN { /* 如果 C_r 是行為能力 */
    tot_weight←tot_weight + Coef_type (Aptitude) /* 所需能力加總 */
    IF C_r∈C_CV THEN   /* 如果搜尋所獲能力集成功 */
     calc_weight←calc_weight + Coef_type (Aptitude) /*所獲能力加總*/
   }
   ELSE { /* 屬於科學或技術能力 */
     IF GeneralCompetency (C_r) THEN        /* C_r 屬於一般能力    */
      C_1←Coef_type (GeneralCompeency)   /* 取得一般能力係數 */
     ELSE                                  /* 屬於技術能力        */
      C_1←Coef_type (SpecificCompetency) /* 取得技術能力係數 */
     tot_weight←tot_weight + C_1          /* 所需能力係數加總 */
     X←{c∈C_CV/c.hasObject=C_r.hasObject} /* 是否為相同的技術 */
     IF X=Ø THEN                          /* 不是相同的技術    */
      lev←Evaluate_subtopics(class (C_r.hasObject), 1) /* 搜尋本體 */
     ELSE {
      C_A←c∈X/(∀y∈X, y.level≤C_A.level)   /* 選擇最佳的等級   */
```

```
        lev←C_A.level                                   /* 取得最佳的等級   */
       }
       IF (lev≥C_r.level) THEN                         /* 所獲等級>=所需等級 */
        calc_weight←calc_weight+C_l                    /* 所獲能力係數加總   */
       ELSE {
        K←integer((C_r.level-lev)/20)                  /* 計算等級差異       */
        calc_weight←calc_weight+(C_l*(1-K/4)) /*所獲能力係數加總   */
       }
     }   /* end ELSE scientific and technical competency */
    }    /* end FOREACH */
    C_match←(calc_weight/tot_weight)*100     /* 最後獲得的係數     */
} /* Algorithm */
```

```
Algorithm Evaluate_subtopics (T, coef) { /* 由概念 T 開始搜尋能力本體*/
   lev←0                        /* 設定 lev 為 0          */
   F←{F_i, i≥0 / F_i→T}         /* 取得 T 的子概念 F      */
   IF F= Ø THEN                 /* 沒有子概念了           */
    return(lev*coef)            /* 回傳等級乘加權係數     */
   ELSE {
    FOREACH F_i∈F REPEAT        /* 針對每一個子概念執行迴路 */
     C←{c∈C_CV/F_i(c.hasObject)} /* 所獲能力中有一個類別 F_i 的實例 */
     IF C≠Ø THEN {              /* 選擇所獲最佳能力       */
      x←c∈C/(∀y∈C, y.level≤ x.level)
      lev←lev+(x.level*(x.hasObject.coef))
     }
     ELSE
      lev=lev+Evaluate_subtopics(F_i,coef(F_i, T)) /* recursively go deeper */
    } /* end foreach */
    return (lev*coef )
   } /* end ELSE   */
} /* end algorithm */
```

資料來源：整理自 Yahiaoui, Boufaïda, & Prié (2006)

十、基於 milestone 比對的工作媒合

在德國，網際網路已經變成線上招募人才的重要管道，儘管如此，網際網路上的人才招募程序卻不是很完美。許多的工作入口網站紛紛成立，有關工作的資訊雖然很多，好像離人們也很近，但事實上這些資訊卻像孤島一般，應徵者可以瀏覽所有的工作資訊，可是卻無法選擇與自己較為匹配的所有工作機會。因此，Bizer 等人 (2005)提出了使用語意網的徵才過程，如圖 3-9 所示，求才者的工作機會必須使用領域的控制字彙來註解，同樣的，應徵者的申請書也必須使用領域的控制字彙來註解，所謂領域的控制字彙就是領域的本體。

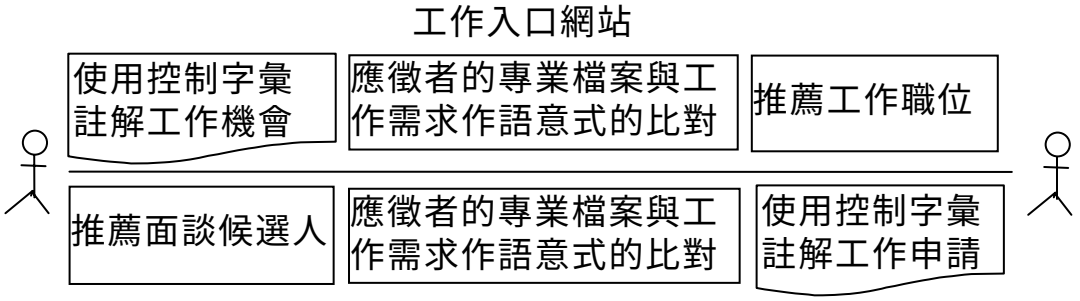

圖 3-9 使用語意網科技的徵才過程 (資料來源：Bizer (2005))

經過註解的文件就可以進行雙方的語意媒合，也就是計算兩個概念的語意相似程度。因此，對於每一筆求才資訊而言，可以分別計算與所有求職資訊的匹配程度，並且經過排序處理之後輸出。同樣的道理，對於每一筆求職資訊而言，也可以分別計算與所有求才資訊的適配程度。兩個概念之間語意相似程度的計算，係利用兩個概念在本體中的相對位置來計算 (Zhong, Zhu, Li, & Yu, 2002)，說明如下：

1、設定本體樹上每一個節點的 milestone 值：每一個本體上的節點都有一個值，稱為 milestone，計算方式如下：

$milestone(n) = 2^{-1} \times \frac{1}{2^{l(n)}}$ ($l(n)$表示節點的深度，根節點的深度為 0)，

因此，根節點的 milestone(0)之值為 0.5、第一階節點的 milestone(1)之值為 0.25、第二階節點的 milestone(2)之值為 0.125。

2、計算兩個概念之間的距離：兩個概念之間的距離定義如下：

$d_c(c1, c2) = d_c(c1, cp) + d_c(c2, cp)$，$cp$ 是概念 $c1$ 與 $c2$ 的共同父概念，而 $d_c(c1, cp)$ 則是 cp 的 milestone 值減去 $c1$ 的 milestone 值。

3、計算兩個概念之間的語意相似度：兩個概念之間語意相似度的定義為：$sim_c(c1, c2) = 1 - d_c(c1, c2)$，也就是 1 減兩個概念之間的距離。

以圖 3-10 為例，概念 D 與概念 E 的語意相似度為：

$sim_c(D, E) = 1 - d_c(D, E) = 1 - ((0.5 - 0.125) + (0.5 - 0.125)) = 0.25$、概念 D 與概念 F 的語意相似度為：

$sim_c(D, F) = 1 - d_c(D, F) = 1 - ((0.25 - 0.125) + (0.25 - 0.125)) = 0.75$

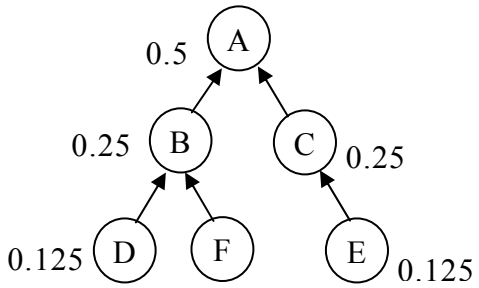

圖 3-10 以 milestone 計算語意相似度

Zhong, Zhu, Li, 與 Yu (2002)的語意相似度計算方式，會使得越下層的相似度越來越大，以圖3-10為例，B與C的相似度為0.5、但D與F的相似度為0.75、再往下一層其相似度為0.875、到了第三層時的相似度就變成了0.9375，因此，在建立本體時必須考量概念的繼承特性與層次。

十一、基於關聯式資料庫比對的工作媒合

所有的關聯式資料庫管理系統都提供了結構化查詢語言(structured query language；SQL)，SQL 中一個極為重要的命令為 SELECT 查詢命令。SELECT 的查詢方式可以是完全比對，例如：SELECT * FROM served_food WHERE cuisine = 'Latin American'; 這個命令是要搜尋資料表 serverd_food 的欄位 cuisine 的內容為 'Latin American'的紀錄；SELECT 的查詢方式也可以是部份比對，例如：SELECT * FROM served_food WHERE cuisine like '%American%'; 這個命令是要搜尋資料表 served_food 的欄位 cuisine 的內容有子字串 'American'的紀錄。

由於以本體為基礎的應用已經擴充到了各個領域，尤其是語意網的概念出現之後，相關的研究更是持續增加，全球資訊網組織 W3C (World Wide Web Consortium)更組織了專門的團隊來進行研究 (W3C, 2007)。語意網的目的乃在提供一個共通的架構，讓網頁資料可以被機器所瞭解，並且能夠被不同的應用程式分享與重複使用。要達成這樣的目的，必須有一個領域共同的字彙來作為相互溝通的基礎，那就是領域的本體。因此，語意網的重要成份包括本體、網頁資料等，儘管本體的應用範圍很廣泛，但是因為本體的特殊性，許多研究都發展新型態的本體工具來儲存本體架構與資料，鮮少使用關聯式資料庫綱要來設計本體架構，也鮮少使用關聯式資料庫來儲存本體資料。但是，未來語意網要能成功，本體的儲存還是需要使用有共同標準的關聯式資料庫管理系統，並且可以透過方便的 SQL 命令來達成語意相似度的計算。

Das, Chong, Eadon, 與 Sriniasan (2004) 利用 Oracle 關聯式資料庫管理系統 (RDBMS)的延伸架構，定義了數個資料表，並且發展了應用程式介面。可以將 OWL (Web Ontology Language)所表示的本體當做輸入，由其中取得本體資料後儲存在這些資料庫表之中，這些資料庫表的結構與作用如下：

1、Ontologies (OntologyId, OntologyName, Owner, …)：本體資料表，用來儲存本體的基本資訊。主要欄位有本體識別碼、本體名稱、擁有者等，其中 OntologyId 是主鍵。

2、Terms (TermId, OntologyId, Term, Type, …)：術語資料表，用來儲存所有的術語，也就是概念名稱、屬性名稱與實例名稱等。主要欄位有術語識別碼、本體識別碼、術語名稱、術語型態等，其中 TermId+OntologyId 是主鍵、Type 則是用來辨識術語的型態是屬於概念、屬性、或是實例。

3、Properties (OntologyId, PropertyId, DomainClassId, RangeClassId, Characteristics)：屬性資料表，用來儲存屬性相關的資訊。主要欄位有本體識別碼、屬性識別碼，領域類別識別碼、範圍類別識別碼與屬性的特徵等，前四個欄位都指向前一個資料表 Terms，其中 OntologyId+PropertyId 是主鍵。

4、Restrictions (OntologyId, NewClassId, PropertyId, MinCardinality, MaxCardinality, SomeValuesFrom, AllValuesFrom)：屬性限制資料表，用來儲存屬性限制的資訊。主要欄位有本體識別碼、新類別識別碼、屬性識別碼、最小基數、最大基數與值的來源等。每一個屬性值的限制資訊都會產生一個唯一的 NewClassId。

5、Relationships (OntologyId, TermId1, PropertyId, TermId2)：關係資料表，用來儲存關係的資訊。主要欄位有本體識別碼、術語一識

別碼、屬性識別碼與術語二識別碼等。關係的表示為 TermId1 PropertyId TermId2。

6、PropertyValues (OntologyId, TermId, PropertyId, Value)：屬性值資料表，用來儲存屬性的值。主要欄位有本體識別碼、術語識別碼、屬性識別碼與值等，即術語 TermId 的屬性 PropertyId 的值為 Value。

除了前述的六個資料表用來塑模本體之外，Das 等人也發展了四個應用程式，可以處理本體的語意式運算，這四個應用程式以運算子的方式表示，可以直接和 SQL 命令結合使用，敘述如下：

1、ONT_RELATED (Term1, RelType, Term2, OntologyName)：RETURNS INTEGER：用來決定兩個名詞 Term1 與 Term2 在本體 OntologyName 中，是否具有 RelType 的關係，RelType 可以是單一運算子如 isa 的關係，也可以是複合運算子如 isa 或 eqv (equivalent)的關係。如果 Term1 與 Term2 具有 RelType 的關係，則 ONT_RELATED 會傳回 1，否則傳回 0，例如: SELECT * FROM served_food WHERE ONT_RELATED (cuisine, “isa”, “Latin American”, “Cuisine_ontology”)=1；這個 SQL 命令是要查詢本體 Cuisine_ontology 中 cuisine isa Latin American 的所有紀錄。

2、ONT_EXPAND (Term1, RelType, Term2, OntologyName) RETURNS ONT_TermRelTableType：這個運算子接受四個參數，其意義和前一個運算子相同，但回傳一個資料表的陣列，包括五個欄位，其名稱與定義如下：

```
CREATE TYPE ONT_TermRelType AS OBJECT (
    Term1Name          VARCHAR(32),   --名詞一
    PropertyName       VARCHAR(32),   -- 屬性名稱
```

```
    Term2Name          VARCHAR(32),    --名詞二
    TermDistance       NUMBER,         -- 名詞距離
    TermPath           VARCHAR(2000) -- 名詞路徑
  );
  CREATE TYPE ONT_TermRelTableType AS TABLE OF
  ONT_TermRelType;
```

ONT_EXPAND 運算子會針對 Term1 RelType Term2 的關係，分別計算兩個名詞 Term1 與 Term2 之間的距離 TermDistance 與經過的路徑 TermPath，並以結構陣列方式傳回。

3、ONT_RELATED 的輔助運算子 ONT_DISTANCE(NUMBER) RETURNS NUMBER; 與 ONT_DISTANCE_ALL(NUMBER) RETURNS TABLE OF NUMBER：前者只傳回兩個名詞之間的最短距離，後者則傳回兩個名詞之間的所有路徑之距離，距離的計算是每一個邊的距離為 1。例如 SELECT * FROM served_food WHERE ONT_RELATED (cuisine, “isa”, “Latin American”, “Cuisine_ontology”, 123)=1 ORDER BY ONT_DISTANCE(123)；這個 SQL 命令會由資料表 served_food 的欄位 cuisine 中搜尋關係是“isa Latin American”的所有紀錄，並且計算最短距離並回傳。而 SELECT * FROM served_food WHERE ONT_RELATED (cuisine, “isa”, “Latin American”, “Cuisine_ontology”, 123)=1 ORDER BY ONT_DISTANCE_ALL(123)，則會傳回所有的距離，這兩個命令中的參數 123 是用來讓 RDBMS 辨識所使用的是 ONT_RELATED 的輔助運算子。

4、ONT_RELATED 的輔助運算子 ONT_PATH(NUMBER) RETURNS NUMBER; 與 ONT_PATH_ALL(NUMBER) RETURNS TABLE

OF NUMBER：與前項的功能類似，但不是計算距離而是路徑，前者只傳回兩個名詞之間的最短路徑，後者則傳回兩個名詞之間的所有路徑，例如： SELECT * FROM served_food WHERE ONT_RELATED (cuisine, “isa”, “Latin American”, “Cuisine_ontology”, 123)=1 ORDER BY ONT_PATH(123) 或 SELECT * FROM served_food WHERE ONT_RELATED (cuisine, “isa”, “Latin American”, “Cuisine_ontology”, 123)=1 ORDER BY ONT_PATH_ALL(123)。

傳統上，以關聯式資料庫來儲存本體，使用者必須自行定義資料庫表，所有基於這些資料庫表的應用程式也必須自行發展。但是因為本體資料的特殊性，大部份的研究都另外發展本體相關的工具，用來編輯本體的階層架構與儲存本體，或是考量應用的屬性而自行定義本體的資料結構。使用 RDBMS 來儲存本體資料的案例極少，Das 等人(2004)所提出的解決方案，是直接在 RDBMS 的系統層次來定義資料庫表。這些資料庫表可以儲存任何領域的本體，因此，是一種領域獨立的本體儲存方式。所有的概念名稱、實例名稱、屬性名稱、關係名稱等，都是以隱性方式儲存，和大部分的本體編輯工具以顯性方式來表示概念等資料不同。Das 等人也發展了四種不同的運算子，可以直接和 SQL 命令結合使用。但是，這四種運算子所提供的功能僅止於查詢本體資料、查詢概念之間的最短距離與所有距離、及查詢概念之間的最短路徑與所有路徑等。對於基本的查詢應用尚可應付，但是對於比較複雜的應用，例如要計算兩個概念之間的相似度時，就受限於這個資料庫表的結構而無法擴充。

十二、工作媒合系統的彙總

本章探討了 11 種工作媒合系統，彙總如表 3-6 所示，其中使用本體來作媒合的方法共有 8 種，分別為基於能力註解比對的工作媒合、基於 WordNet 比對的工作媒合、基於多重技能比對的工作媒合、基於最短途徑矩陣比對的工作媒合、基於整合四種模式比對的工作媒合、基於能力模式比對的工作媒合、基於 milestone 比對的工作媒合、與基於關聯式資料庫比對的工作媒合等。基於能力陳述比對所使用的 IBM 專長分類系統也可以算是一種鬆散的本體。在工作媒合方式上，大致可以分成基於語法的關鍵字媒合、與基於語意的相似度媒合。前者有基於結構模式比對的工作媒合、基於補償式比對的工作媒合、與基於整合四種模式比對的工作媒合等 3 種，其他的 8 種都是基於語意方式的媒合。

表 3-6 工作媒合系統的彙總

方法論與提出者	媒合方式	特點
基於能力註解比對 (Bourse, Harzallah, Leclère, & Trichet, 2002)	使用領域本體的字彙來註解履歷表上的能力，再進行媒合	僅能稱為履歷表產生器，未談及媒合的機制
基於能力陳述比對 (Pan & Farrell, 2007)	使用專長分類系統來計算能力陳述之間的語意相似度	以語意角色所含有的共同資訊量來計算語意相似度
基於 WordNet 比對 (Wicaksana & Yetongnon, 2006)	使用 WordNet 的同義詞集與上下位詞的關係來計算語意相似度	WordNet 可能無法涵蓋領域的概念，導致無法計算語意相似度或造成偏差

基於結構模式比對 (Yi, Allan, & Croft, 2007)	關鍵字查詢機制, 能利用過去媒合成功的紀錄作延伸查詢	利用整個結構做比對, 改進了單純關鍵字查詢的缺點
基於補償式比對 (Sure, Maedche, & Staab, 2000)	能力供應大於能力需求時給予補償, 小於需求時給予懲罰	允許設定能力的重要性加權, 允許能力不足者成為候選人
基於多重技能比對 (Mohaghegh & Reza Razzazi, 2004)	使用本體節點本身的貢獻度與對子節點的貢獻度來計算語意相似度	採用多重繼承的關係, 加權值由領域專家決定可能導致實務上的困難
基於最短途徑矩陣比對 (Lv & Zhu, 2006)	把技能本體當作是一個有向的圖形結構, 利用 Dijkstra 的演算法來計算語意相似度	本體關係可以是 isa 或 partof，兩者的相似度不同。某些 isa 關係的相似度為 1，可能過度高估
基於整合四種模式比對 (Biesalski, Breiter, & Abecker, 2005)	正確媒合、分類相似度、比例式媒合、與補償式媒合	分類相似度與補償相似度可能大於 1 造成誤導
基於能力模式比對 (Yahiaoui, Boufaida, & Prie, 2006)	使用 ER 本體計算能力本位語意媒合, 包括能力、文憑、經驗、專業檔案與個人資訊等	ER 本體由五個子本體組成, 是一種完整的能力模式
基於 milestone 比對 (Bizer et all, 2005)	使用本體節點的 milestone 值來計算語意相似度	越下層節點的相似度越來越大
基於關聯式資料庫比對 (Das, Chong, Eadon, & Sriniasan, 2004)	使用四種不同的運算子直接和 SQL 結合使用, 可以查詢概念間的最短距離與所有距離	Oracle 的延伸功能, 可以儲存領域本體資料, 但相似度的計算要另行定義

資料來源：本研究

第四節 本章小結

在工作媒合系統的發展方面，美國早在 1975 年使用中大型主機的年代，就已經有了工作媒合系統的出現。但其使用情形極為繁複，除了要瞭解職業分類典 DOT 與其他標準化的文件外，求才者與求職者都必須填答問卷，才能將相關資料轉換成數字型態的代碼，以便輸入到工作媒合系統中。儘管如此，Nathanson (1975)的評鑑結果認為工作媒合系統有其功效，媒合分數越高的就業人員，其停留在工作職位上的時間也越久。

Botterbusch (1986)的報告更顯示當時全美有 15 個工作媒合系統在運作，所有的系統都是以美國的職業分類典 DOT 與相關的標準文件，作為資料輸入與媒合的基礎。系統媒合的項目包括教育發展、職業準備、一般能力、態度與特殊技能等。和現在的工作銀行或人力銀行的媒合項目相比較，顯然上一代的工作媒合系統考慮的面向比較周全，例如一般能力的媒合、與一些標準化測驗的媒合等。不但要考慮應徵者的能力是否符合工作職位的需求，也就是能不能執行該項工作，也要考量應徵者對該工作的尊重程度，也就是有沒有意願做該項工作。

在工作媒合方式上，大致可以分成基於語法的關鍵字媒合、與基於語意的相似度媒合。前者的求職與求才資料，大都以關聯式資料庫系統來儲存，又可以分成單一欄位的關鍵字媒合，或是將所有欄位的資料攤平後，視為一個欄位來媒合，媒合的技術通常都可以藉由 SQL 的查詢命令來完成。在相似度媒合方面，除了資料的儲存可以是資料

庫系統之外，也可以是其他模式的儲存方式，但和關鍵字媒合不同的地方在於領域本體的支援與相似度計算的方法。

在語意相似度的工作媒合方法上，本章探討了多種不同的方法，其中 Pan 與 Farrell (2007)所提出的能力陳述之語意相似度計算，係根據兩個能力陳述所擁有的共同資訊量，作為語意相似度的衡量基準。又可以分成兩個部份，一是 18 個能力動詞之間的語意相似度，另外一個是語意角色之間的語意相似度。這個方法和其他方法不同的地方，在於能力的表示是以完整的能力陳述來表示，也就是「能力動詞+能力受詞」的方式，其他大部份的相似度的計算，都只限於能力受詞，也就是技能實例之間的語意相似度。

Wicaksana 和 Yetongnon (2006)的工作媒合是基於 WordNet 的語意相似度計算，但是因為 WordNet 是一種英文自然語言的詞典，可以稱之為一種通用性的本體，而非特定領域的本體。因此，在計算兩個概念之間的相似度時，可能會碰到一詞多義的狀況，也可能因為許多領域的實例不會出現在 WordNet 之中，可能會造成無法計算語意相似度或計算上的偏差。

Yi, Allan, 與 Croft (2007)所提出的工作媒合系統，係利用過去成功配對的經驗資料，作為延伸查詢的依據。雖然本質上還是屬於關鍵字查詢的範疇，但確實可以最大化查詢的結果。因此，其語意相似度是指與過去的配對經驗有相似。

Sure, Maedche, 與 Staab (2000)所提出的工作媒合系統 ProPer，

採用補償媒合的概念，允許在多重技能之間設定不同的加權，也允許能力達不到要求的申請者成為候選人。但是，在計算匹配的相似度時，申請者所具備的技能與雇主所要求的技能之間，必須是一對一的對應，不允許用其他技能來取代。在實務應用上，如果缺乏不同技能名稱的相似度轉移機制，也就是只利用表面的名稱作媒合，而不以內涵來媒合，該工作媒合系統仍然不完整。

Mohaghegh 與 Reza Razzazi (2004)利用本體來計算不同技能名稱之間的語意相似度，其重要的概念係由本體的 isa 架構具有繼承的特性而來。當兩個節點之間有 isa 的繼承關係時，子節點與父節點的語意相似度就是父節點對子節點的貢獻度，也就是繼承邊的加權值。而父節點與子節點的語意相似度為子節點的加權乘以繼承邊的加權。因此，可以計算例如 C++與 C、或 Delphi 與 Pascal 之間的語義相似度。但是加權值的決定必須依賴領域的專家，想要推廣到所有領域的工作媒合似乎是不容易達成。

Lv 和 Zhu (2006)的工作媒合方法也需要先建立能力本體，和 Mohaghegh 與 Reza Razzazi (2004)的能力本體類似，但除了 isa 架構外，還包括 partof 關係。前者的相似度可以達到 1，而後者小於 1。以相似度的定義而言，isa 關係的相似度為 1 似乎是過於高估。在相似度的計算方面，是將整個技能本體當作是一個圖形結構，利用 Dijkstra 的最短途徑演算法來求出此圖形結構的最短途徑矩陣，其節點之間的最短距離就是相似度。除此之外，還可以對不同的能力項目設定加權值。

Biesalski、Breiter、與 Abecker (2005)的工作媒合系統利用四種媒合：正確媒合、分類相似度媒合、比例式媒合與補償式媒合。其中的分類相似度媒合極可能會造成偏差，因為只要是 isa 架構的繼承關係，其相似度就是 1。另外，補償式媒合也允許超過 1 的相似度，也可能造成整體相似度值的誤導。

Yahiaoui、Boufaïda、與 Prié (2006)提出寬廣能力模式的工作媒合，此能力模式包括特殊技能、一般能力與行為能力等，也是一種能力本位的工作媒合。相對而言，Bizer 等人 (2005)的語意相似度計算就較為簡單與可行，採用稱為里程碑 (milestone)的概念。Das, Chong, Eadon, 與 Sriniasan (2004) 則是利用 Oracle 關聯式資料庫管理系統(RDBMS)的延伸架構，定義了數個資料表，並且發展了應用程式介面，可以和標準的 SQL 一起使用，其語意相似度的計算是以兩個名詞之間的距離來衡量。