Web API 之視覺化搜尋與推薦工具

Visualized Search and Recommendation Tool for Web APIs

游筑安、李佳穎、何敏瑜、林軒如、馬尚彬
國立臺灣海洋大學資訊工程學系
Email: erica31129@gmail.com, j961833200717@gmail.com, e55469@gmail.com, mis101bird@gmail.com, albert@ntou.edu.tw

摘　要

近年來 Web API 蓬勃發展，各領域之軟體系統可藉由整合現有之 Web API，提高系統開發的速度。因此，如何協助使用者在數量龐大的 Web API 中，快速並準確地找到符合目標的 API，以整合到其開發之應用系統中，便成為當前一項重要的需求。本研究開發一個網站應用系統：ServRel，用以提供 Web API 之搜尋與推薦的相關功能與使用者介面，搜尋功能部分支援一般的關鍵字搜尋以及輸入輸出搜尋，推薦功能部分則可推薦出合作型 API 與競爭型 API，而搜尋與推薦結果均可透過視覺化的 API Graph 來呈現其結果。同時，ServRel 也提供工作區(workspace)與 API 收藏功能，以提供使用者更便利的 API 管理功能。

關鍵字：API 搜尋、API 推薦、資訊視覺化、個人化服務

一、緒論

近年來 Web API 蓬勃發展，各領域之軟體系統可藉由整合現有之 Web API，提高系統開發的速度，並將開發人力聚焦於其核心功能上；在另一方面，Web API 之運用也讓使用者得到較好的使用者體驗，例如使用者可透過熟悉的 Google 帳號或 Facebook 帳號登入系統，並可以功能完整的 Google Map 或 Bing Map 瀏覽地域相關資訊。Web API 通常視為網際服務(Web Service)的別稱，泛指可在 Internet 上運作的 API (Application Programmable Interface)。

根據知名的 ProgrammableWeb [1]網站分析，REST (REpresentational State Transfer)服務架構為目前最主流的 Web API 架構，其透過標準的 HTTP 協定之 GET、 POST、PUT、DELETE 等操作(method)存取資源(resource)。Web API 可說是目前最為普遍的軟體元件提供模式，其每日均以相當的數量在成長，例如在 ProgrammableWeb 上即超過七千個以 REST 架構提供之 Web API。

因此，如何協助使用者在如此數量龐大的 Web API 中，快速並準確地找到符合目標的 API，以整合到其開發之應用系統中，便成為一項重要的需求。

目前雖然已有數個 Web API 搜尋引擎，如 ProgrammableWeb [1]、APIs.io [2]、Mashape [3]、Bluemix [4]等，提供了以關鍵字或以標籤搜尋 API 之功能。然而，這樣的搜尋並沒有考量 Web API 之特性，使用者不容易在幾次的查詢當中找到符合其需求之 API，API 搜尋容易變成一個耗時且繁瑣的工作[5]。

因此，本研究基於實驗室過去的研究成果：CSD (Contextual Service Discovery)[6]，開發一個視覺化的 Web API 搜尋與關聯式推薦工具，我們稱之為 ServRel。ServRel 工具是基於 MVC (Model-View-Control)模式建構的網站應用系統，用以提供 Web API 之搜尋與推薦的相關功能與使用者介面，搜尋功能部分支援一般的關鍵字搜尋以及輸入輸出搜尋，推薦功能部分則可推薦出合作型 API 與競爭型 API，而搜尋與推薦結果均可透過視覺化的 API Graph 來呈現其結果。同時，ServRel 也提供工作區(workspace)與 API 收藏功能，以提供使用者更便利的 API 管理功能。

二、文獻回顧與探討

底下就針對本研究相關之背景知識以及與目前的 Web API 之搜尋引擎工具進行說明與分析:

1. 資訊檢索(Information Retrieval)與情境導向之服務探索方法(Contextual Service Discovery)

資訊檢索(Information Retrieval，簡寫為 IR)[7]之搜尋機制是基於一般文件或是資源媒介文件建

立出的索引文件，以查詢需求內容(例如關鍵字、標題、段落)的相關資訊。現今 IR 系統最常用的模型為向量空間模型(VSM: vector space model)[8]，此模型會將需求文件向量化，然後從索引中檢索出與需求字詞有交集的文件向量，算出需求文件與資料庫文件的相似度。CSD (Contextual Service Discovery)[6]則基於 IR 方法，並加入 WordNet 語意資訊，提出一個以字彙擴充(term expansion)與綁定涵蓋度分析(binding coverage analysis)技術為基礎的服務探索方法，用以確保探索出之服務能符合需求之綁定情境(binding context)，而實驗證實 CSD 相較於一般以 IR 為基礎之服務探索方法，更能確實提升探索之準確度(precision)。本研究將以 CSD 方法為基礎，發展 Web API 之搜尋與推薦工具。

2. Swagger 文件格式

Swagger [9]由 Wordnik 公司在 2011 年釋出，採用比 XML (eXtensible Markup Language)更輕量、解析速度更快的 JSON (JavaScript Object Notation)格式撰寫，為開源社群中最被廣泛運用的 Web API 描述格式，Swagger 2.0 是目前開源社群內最受歡迎的 Web API 描述撰寫格式，有許多 Web API 之開發者均已提供其 Swagger 描述文件，以供客戶端與其他開發者運用。本研究以 Swagger 作為預設的 Web API 描述格式，用以進行 Web API 之發布、測試、搜尋與調用。

3. D3.js (Data-Driven Documents)

D3 [10]是一種可將資料及網頁文件互相連結，並驅動整個連結流程、產生一連串流暢動畫呈現的工具。D3 實現資料視覺化主要過程為以下四項：(1)載入資料(loading)進網頁記憶體；(2)將資料與網頁元素進行綁定(binding)，可依需求建立網頁元素；(3)將網頁元素變形(transforming)以詮釋資料；以及(4)針對使用者輸入進行過場轉移(transitioning)，以呈現不同狀態之效果。本研究之前端網頁介面即使用 D3.js，以達成較好視覺化之效果。

4. 目前之 Web API 搜尋引擎

目前較常見的 API 搜尋引擎，仍以傳統文字搜尋方式為主。以下就簡要介紹較知名的幾個 API 搜尋引擎：

- ProgrammableWeb：

  是最龐大也是歷史最悠久的 Web API 資源網站，其上所發布的 Web API 數量是最多的，目前已發布超過七千個 RESTful 服務。ProgrammableWeb 各 API 的資訊頁面中會推薦相關 API，也提供標籤(tag)協助使用者搜尋需要的 API。ProgrammableWeb 之搜尋方式仍是傳統的文字相關度比對。

- APIs.io：

  提供透過 APIs.json 格式搜尋 API 之方式，會在整個網路上搜尋含有 APIs.json 格式的文檔，達成自動化 API 發布的功能。APIs.json 是一個簡單的 API 描述文件，主要分三個部分：API 基本資料、API 集合與 API 屬性集合，是一個相對簡單的描述格式。然而，APIs.io 這樣的 API 發布模式目前尚未普及。

- Mashape：

  新興的 API 統一管理服務，其作法是將市面上各方 API 整理為統一入口，以市集的概念去列出 API，讓搜尋 API 就像在 App Store 瀏覽 APP 一樣方便。在開發者相關功能部分，Mashape 提供開發者帳號一個專屬的 key 做為類似身分證明的條碼，所有 API 都只需要用這個單一 key 即可呼叫服務，Mashape 的搜尋機制仍為傳統的文字相關度比對。

- Bluemix：

  IBM Bluemix 是 IBM 的 PaaS (Platform as a Service)公有雲平臺，其服務主打減少管理與控制雲端基礎設施的時間，讓開發者可以更專注於以 API 開發應用程式。Bluemix 可允許 Java、Ruby on Rails、RubySintra、Node.JS 等語法，支援混合式開發環境，並提供 SQL DB、NoSQL、快取等多種應用服務。

整體而言，目前知名的 API 搜尋引擎都僅提供傳統的關鍵字搜尋，仍以文字相關度為基礎，並未考慮到輸入或輸出資料等介面資訊，導致其搜尋準確度不佳，也未提供便利的使用者介面，使用者必須自行比對其服務是否符合所需，且必須自行填寫多次表單或進行多次搜尋方可找到其所需之 API，這些問題即為本研究開發 ServRel 之需求。

三、研究方法與架構

本章節將說明本研究方法之核心概念、系統架構、系統運作流程以及使用者介面設計，本研究方法之成果：ServRel 系統即根據這些分析與設計結果進行後續之開發。

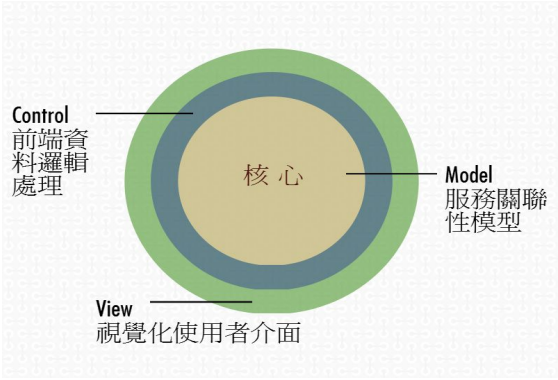

圖 1 計畫概念圖

表 1 搜尋功能

編號	功能說明
0-1	提供以關鍵字或以 API 輸入及輸出搜尋，並可瀏覽 API 詳細資料及相關 API。
0-2	於搜尋結果頁可查看搜尋結果 API 之前置合作型、後置合作型、競爭型 API 及 API 詳細資訊，API 詳細資訊區可試用 API 及下載程式碼。

表 2 API 視覺化功能

編號	功能說明
1-1	使用者可在搜尋後，將列表中想查看的 API 拖曳至畫布區，即可在畫布區看見視覺化顯示之 API，同時也可在畫布區查看此 API 之前置合作型、後置合作型、競爭型 API 及 API 詳細資訊。
1-2	使用者在畫布上點選兩個 API，即可出現關聯線，顯示兩 API 之相似度。
1-3	使用者可將畫布上之 API 移除。

表 3 API 收藏功能

編號	功能說明
2-1	使用者登入後即可將搜尋到之 API 加入我的最愛。
2-2	使用者可在我的最愛區內，依自己喜好分類。
2-3	我的最愛區內之 API 若有更新會提醒使用者。

表 4 工作區功能

編號	功能說明
3-1	使用者登入後，可將同一專案所需使用之 API 都拖曳至畫布區，一併存進工作區的專案內。
3-2	在工作區的專案內，使用者可直接下載此專案內所有 API 之程式碼。
3-3	工作區內之 API 若有更新會提醒使用者。

表 5 提供 API 功能

編號	功能說明
4-1	使用者登入後，即可上傳自己開發之 API，供其他使用者使用。
4-2	使用者可刪除、更新自己提供之 API。

本研究以 MVC (Model-View-Control) [11]模式為基礎，其核心概念圖如圖 1 所示，ServRel 系統是以服務關聯性模型(Service Relationship Model，簡稱 SRM)作為 MVC 模式之 Model，為系統之核心，在外圍則包覆前端的視覺化使用者介面(Visualized User Interface，簡稱 VUI)，扮演 MVC 模式中的 View，以提供給使用者進行相關功能之操作，最後則以前端資料邏輯處理層(Data Logic Handler，簡稱 DLH)進行中介輔助，銜接 SRM 與 VUI。此三個子系統之主要負責工作描述如下：

1. **服務關聯性模型(SRM)：**SRM 之主要責任為提供後端服務功能，可從 DLH 接收請求，並將服務功能之執行結果透過 DLH 交由 VUI 資料進行呈現。SRM 所提供之後端服務功能包含兩類：(1)提供給使用者 Web API 之搜尋與推薦資訊，以及(2)使用者之個人儲存資料。
2. **前端資料邏輯處理(DLH)：**DLH 用來處理從 SRM 回傳的資料，將之轉換成前端介面方便使用的資料集，再提供給視覺化使用者介面(View)。
3. **視覺化使用者介面(VUI)：**VUI 主要功能包含：(1)將使用者發送的服務請求傳送至 DLH，以取得後端服務執行結果；以及(2)把執行結果資料呈現給使用者：本研究之使用者介面規劃呈現於 Web 端，因此本研究規劃運用 D3.js 達成服務間關聯性的視覺化呈現，以提供較好的使用者體驗。

底下將進一步說明此三子系統之細部結構與功能：

1. **服務關聯性模型(SRM)**：此子系統透過伺服器端的 Tomcat 和 Java Servlet 建置，提供 User Resource 與 API Resource 兩種類型之資料：

   (1) User Resource：包含帳戶的驗證與註冊，以及個人資料之存取，如我的最愛(即 API 收藏)、我的工作區、我上傳的 Web API 等資料。

   (2) API Resource：提供給前端使用者 Web API 查詢與推薦功能。Web API 推薦功能包含競爭型(competition)與合作型(cooperation)兩種類型，以下將分別介紹此兩種推薦類型之核心概念：

   A. 競爭型(competent)

   關聯 API 推薦：競爭型 API 代表其功能與目標 API 相近，因此，要能找出目標服務之競爭型 API，主要關鍵在於透過 query-by-example 之概念，先將目標 API 設定為需求，根據此需求透過 CSD 找出待選 API 的相似度分數後再進行排序，高於內部門檻分數的 API 將視為合乎條件之 API，納入 API 集合後再推薦給使用者。

   B. 合作型(cooperative)

   關聯 API 推薦：合作型 API 可銜接目標 API 之輸出資料，或產生目標 API 可接收之輸入資料，其概念如圖 2 所示，分為可產生目標 API 可接收之輸入資料的前置合作型 API (pre-cooperative API)，以及可銜接目標 API 之輸出資料的後置合作型 API (post-cooperative API)。

   換句話說，前置合作型 API (pre-cooperative API) 的待選服務 (candidate API) 之輸出參數 (output parameter)必須多於或等於目標 API (target API) 的輸入參數 (input parameter)；而後置合作型 API (post-cooperative API) 的待選 API (candidate API) 之輸入參數 (input parameter)必須少於或等於目標 API (target API) 的輸出參數 (output parameter)。

   一樣透過 query-by-example 之概念，先將目標 API 的輸入或輸出設定為需求，根據此需求透過 RSM 求出待選 API 的分數後再進行排序，計算完成後，我們即可找出 PRE 或 POST 分數高於門檻分數之服務，再將合乎條件之 API 納入 API 集合，分別推薦給使用者。

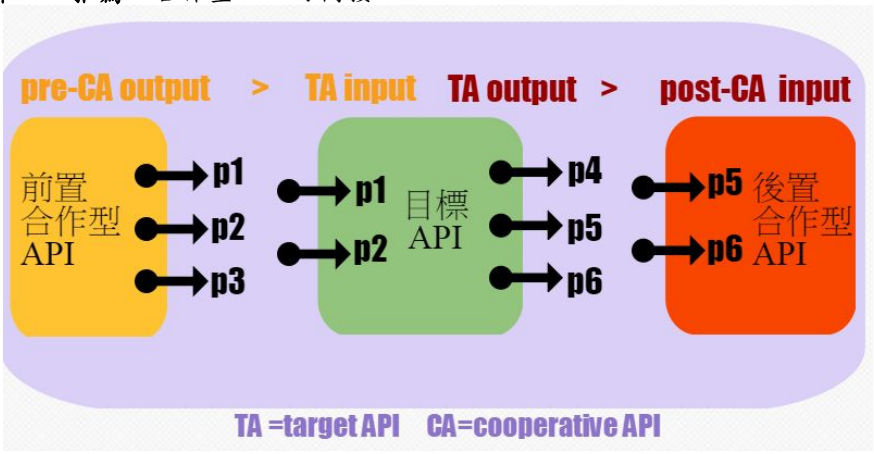

圖 2 合作型服務之概念圖

2. **前端資料邏輯處理(DLH)**：此部分透過 JavaScript 技術建置，包含 Personal Storage、User Authentication 和 Service Invoker 三個主要功能服務：

(1) Personal Storage：管理登入後的個人資料，包含我的最愛(即 API 收藏功能) 我的工作區 我上傳的 Web API。

(2) User Authentication：負責使用者註冊與認證。

(3) Service Invoker：前端與後端溝通的橋梁，負責接收後端資料和發送前端請求。

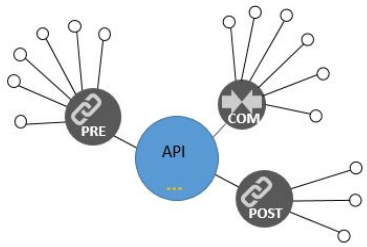

圖 3 API Graph 之視覺化呈現示意圖

3. **視覺化使用者介面(VUI)**：此元件透過 HTML5、CSS3 與 JavaScript 等技術建置，並規劃將使用 D3.js 函式庫和 Swagger Editor 之開放原始碼。

(1) D3.js 函式庫之運用：使用此函式庫繪製如圖 3 的 API Graph，透過 API Graph 之形式呈現不同 API 之間的關聯性。當使用者選了某個 Web API，在本系統的視覺化使用者介面上，將呈現此 API 擴展後之 API Graph，分別呈現其競爭型服務、前置合作型服務、與後置合作型服務，使用者又可點選任一個競爭型或合作型服務，再展開此節點之競爭型服務與合作型服務，透過此視覺化之 API Graph 介面達成較好的服務探索(API exploration)效果。

(2) Swagger Editor 開放原始碼之運用：透過整合 Swagger Editor 開放原始碼及 Swagger 編輯器之介面，讓使用者可便利地撰寫 Swagger 文件，同時提供 Web API 上傳、檢查與發布功能。

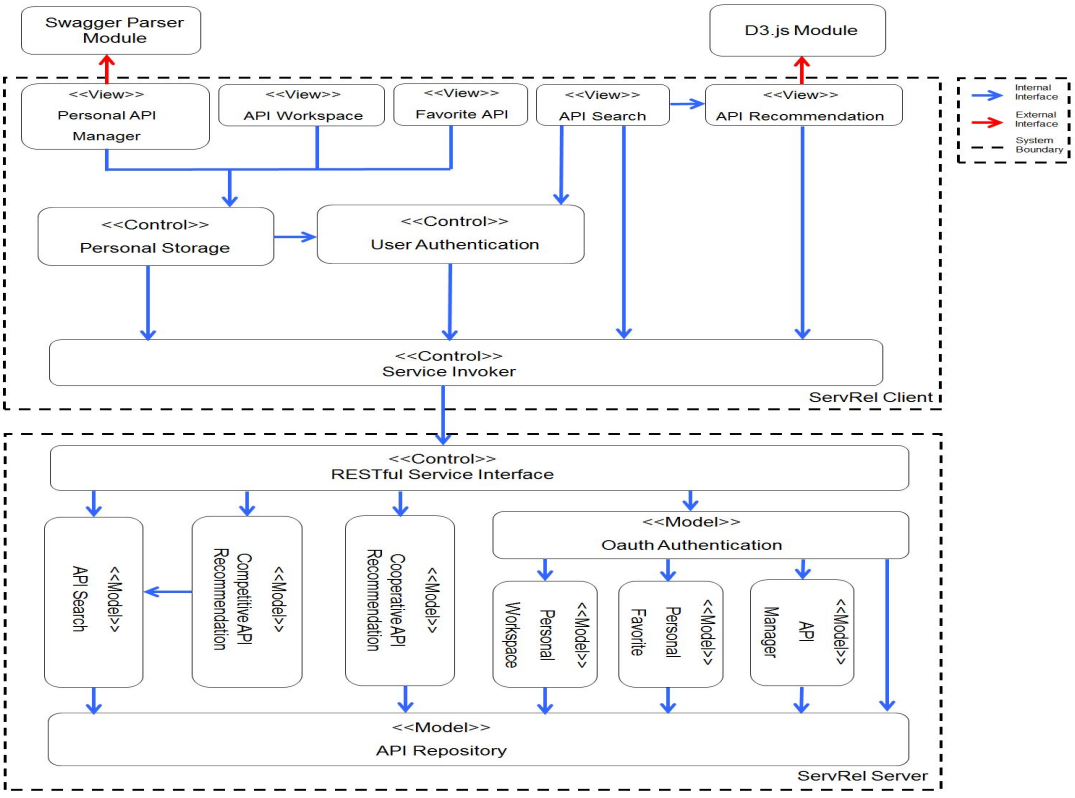

圖 4 系統架構圖

根據上述之核心概念，ServRel 系統之系統架構如圖 4 所示，主要包含五種使用者介面：個人 API 管理介面(Personal API Manager)、專案工作區介面(API Workspace)、我的最愛 API 介面(Favorite API)、API 搜尋介面(API Search)及 API 推薦介面(API Recommendation)，以及三種控制模組：個人資料儲存模組(Personal Storage)、用戶認證模組(User Authentication)、服務調用模組(Service Invoker)。

個人 API 管理介面(Personal API Manager)、專案工作區介面(API Workspace)及我的最愛 API 介面(Favorite API)會使用到個人資料儲存模組(Personal Storage)進行個人資料的儲存，需要經過用戶認證模組(User Authentication)確認是會員後，才能繼續使用服務調用模組(Service Invoker)呼叫後端功能。

後端的 ServRel API (即 RESTful Service interface)提供服務調用模組(Service Invoker)所需要的資料(API 規格請參見附件)，並運用服務搜尋模組(API Search)、競爭型 API 推薦模組(Competitive API Recommendation)、合作型 API 推薦模組(Cooperative API Recommendation)、Oauth 認證模組(Oauth Authentication)、個人專案工作區模組(Personal Workspace)、我的最愛模組(Personal Favorite)、服務管理模組(API Manager)等，從 API 儲存庫(API Repository)取得相關資料，並透過這些後端模組執行相關功能，以回應前端提出之請求。

而 ServRel 之系統運作流程圖設計如圖 5，使用者進入 ServRel 系統後，可依照所需選擇先登入或直接開始使用，系統首頁即是搜尋頁面及空白的畫布區，使用者輸入查詢條件後，便會顯示符合條件的 API 列表，使用者可將 API 拖曳至畫布區，即可在畫布上看見視覺化的 API 資訊。在列表區及畫布區，使用者皆可查看 API 的詳細資訊、試用 API 及下載程式碼，也可查看和某 API 有競爭、輸入合作或輸出合作關係的 API (以 API Graph 的方式呈現)。

在查詢過程中，若使用者看到欲收藏的 API，可將此 API 存入我的最愛(需登入，未登入則跳至登入畫面)。查詢完後使用者將需要的 API 都放進畫布裡，移除多餘的 API 後，再儲存畫布上所有 API 至工作區的專案中(需登入，未登入則跳至登入畫面)，在工作區的專案裡，使用者可一次下載此專案內所有 API 的程式碼。

若使用者想提供自己開發之 API 給其他使用者使用，登入後可直接進入個人 API 功能頁面，上傳與發布自己開發的 API，之後其他使用者即可看到此 API。

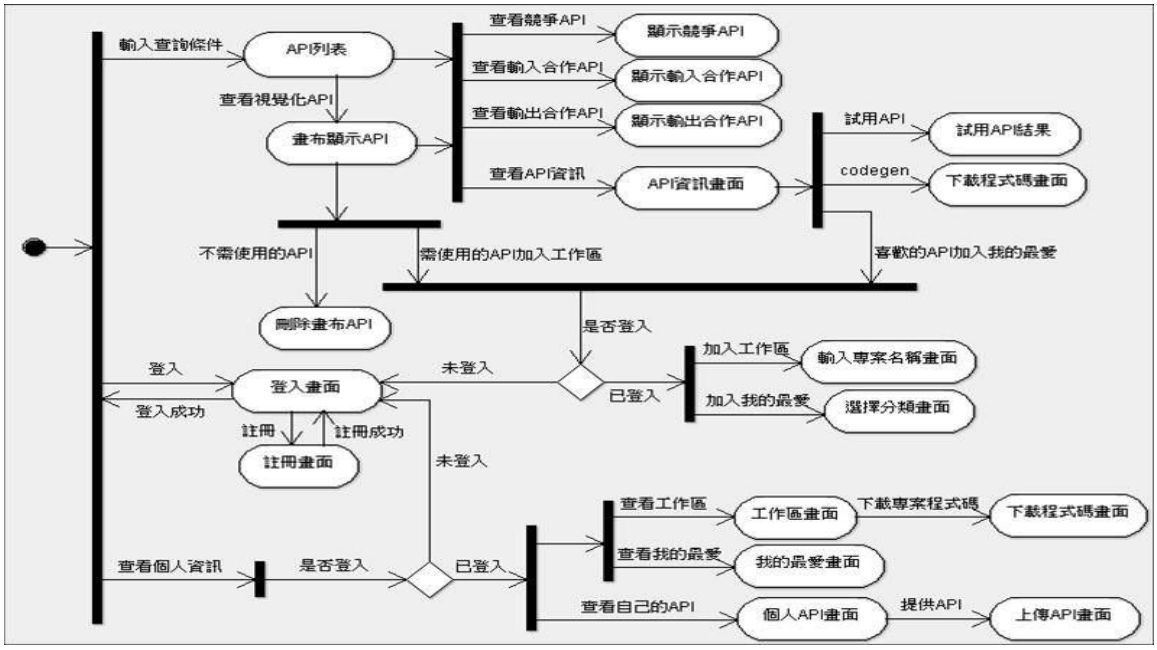

圖 5 系統使用者流程圖

四、實作成果

根據上述之系統架構與設計概念，本研究之實作成果如下：

1. 查詢功能

有一位開發者，想搜尋一 API 是由經緯度尋找活動，他可以使用 API 查詢功能，輸入關鍵字(latitude,longitude)或輸入所需 API 的 input 型態(latitude,longitude)及 output 型態(event)，方框旁邊的加號可以增加型態（如圖 6）。

按下搜尋後，查詢結果（如圖 7）由左至右包含了 API 的 icon、API 名稱、將此 API 加入工作區的使用者人數、以及追蹤此 API 放入我的最愛的使用者人數。點選 API 可觀看 API 的細部資訊，上排選項由左至右分別是 API 內部資訊、前置合作型 API、後置合作型 API 以及競爭型 API。

圖 6 API 查詢頁面(網頁左半)

2. API 視覺化功能

使用者可將有興趣的 API 之 icon 拖曳進入右邊的畫布區（如圖 8）。點擊 icon 可瀏覽 API 詳細資料、拖曳進畫布區的 API 將會顯示三條分別連向不同圓形的連接線，按下右上角的連接線連到具 pre 字樣的圖形，可以顯示出此 API 的前置合作型 API。如範例所示，開發者可看到排名前三的「由地名搜尋經緯度」API。而左上角連接具 post 字樣的圖形則是顯示後置的合作型 API。在此範例中，開發者會看見排名前三的「由活動尋找影片」API。按下左下角的第三條連接線所連到的雙箭頭圖形，會顯示競爭型 API。

以此方式推薦兩種合作型 API，使得開發者在製作專案時能方便運用，且提供開發者更多樣的選擇及想法。

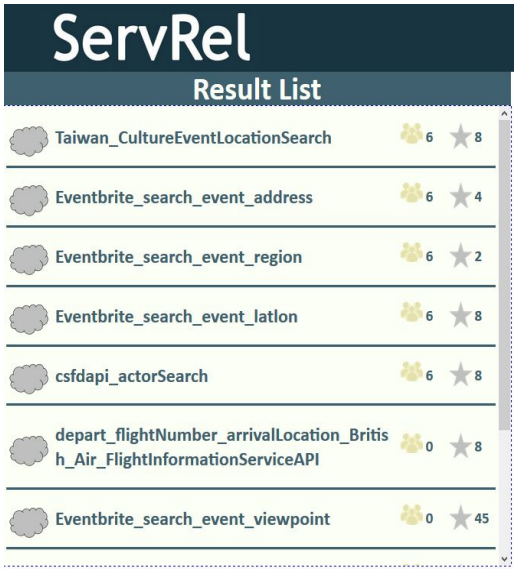

圖 7 搜尋結果(網頁左半)

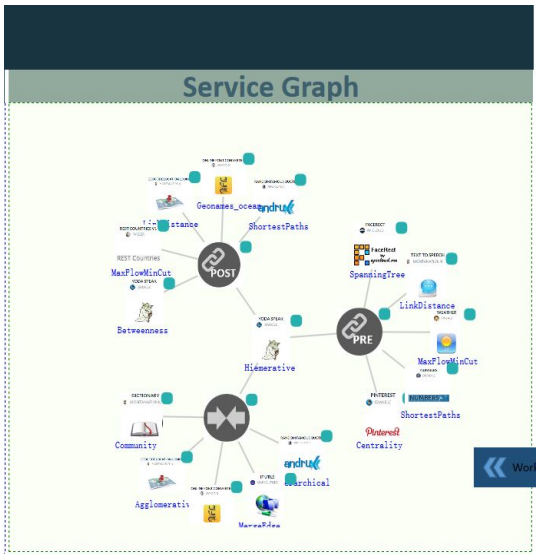

圖 8 畫布區點選畫面(網頁右半)

五、結論

近年來 Web API 蓬勃發展，各領域之軟體系統可藉由整合現有之 Web API，提高系統開發的速度，因此，如何協助使用者在數量龐大的 Web API 中，快速並準確地找到符合目標的 API。目前雖然已有數個 Web API 搜尋引擎，但都僅提供以關鍵字或以標籤搜尋 API 之功能，這樣的搜尋並沒有考量 Web API 之特性，使用者不容易在幾次的查詢當中找到符合其需求之 API。因此，為解決上述問題，本研究建構了一個 API 搜尋網站，其特性包含：

(1)提供 API 輸入輸出型態搜尋。(2)提供搜尋結果之畫布式圖像視覺化呈現。(3)提供 API 之前後關聯性推薦。

參考文獻

1. ProgrammableWeb. Available from: http://www.programmableweb.com/.
2. APIs.io. Available from: http://apis.io/.
3. mashape. Available from: https://www.mashape.com/.
4. IBM. Bluemix. Available from: https://www.ng.bluemix.net.
5. Torres, R., B. Tapia, and H. Astudillo. Improving Web API Discovery by Leveraging Social Information. in 2011 IEEE International Conference on Web Services (ICWS). 2011.
6. Ma, S.-P., C.-W. Lan, and C.-H. Li, Contextual service discovery using term expansion and binding coverage analysis. Future Generation Computer Systems, 2015. 48: p. 73-81.
7. Chien, L.-F. Information Retrieval Techniques for Spoken Language Processing. in International Symposium on Chinese Spoken Language Processing. 2002.
8. Salton, G., A. Wong, and C.-S. Yang, A vector space model for automatic indexing. Communications of the ACM, 1975. 18(11): p. 613-620.
9. Swagger. Available from: http://swagger.io/.
10. D3: Data-Driven Documents. Available from: https://d3js.org/.
11. Fowler, M. GUI Architectures. 2006; Available from: http://martinfowler.com/eaaDev/uiArchs.html.

附件、 API 規格說明

1. Search(主頁左半)

按鍵:搜尋格式

輸入	輸入	輸出	輸出
Keyword	/search/keyword?text=”XXX”	API 資料:名稱、rank、logo、工作區人數、我的最愛人數	[{"id": "...","name": "...","rank":,"...","favorpeople":"...","workspacepeople":"...","pic":"url"},{…}]
Description	/search/io?		
Input output	inputs=[name, place] &outputs=[latitude, longitude] &description=” XXX”		

按鍵:搜尋後

輸入	輸入	輸出	輸出
搜尋後 1-1. pre 1-2. post 1-3. Comp	/apis/cooperative/pre?id=”2” /apis/cooperative/post?id=”2” /apis/competitive?id=”2”	API 資料:名稱、logo	[{"id":"...","name":"...","rank":,"...","favorpeople":"...","workspacepeople":"...","pic":"url"},{…}]

2. Graph(主頁右半)

輸入	輸入	輸出	輸出
1. pre 選項	/apis/cooperative/pre?id=”2”	API 名稱、logo	[{"id":"2", "name": "API","rank": "1"}, {…}]
2. post 選項	/apis/cooperative/post?id=”2”		
3. comp 選項	/apis/competitive?id=”2”		
4.info 選項 (pop window)	/api/info?id=”2”	API 資訊: test、我的最愛人數、工作區人數、日期、敘述、URL	{"Info": { "name": "API", "introduction":"...","hostUrl":"…", "path": "/call", "httpMethod":"…","contentType": "…", "apiEndpoint": "…","favorpeople":"…", "workspacepeople": "45"},"requests: ["name": "…", "description":"....","parameterpath": "…", "testData": {"type":"AAA"}, {…}]}
5.兩個 API 間的相似度	/apis/similarity?id1=”1”&id2=”3”	細部分數	{"IOMappingScore":"...","UnitTestingScore":"...","TextBasedSimilarityScore":"...","AcceptanceTestingScore": "...","FinalScore": "..."}