49-2

OAuth 2.0 授權機制應用於校園開放 API 平台

藍博耀 徐嘉連

輔仁大學資訊工程學系

摘 要

近年從民間到政府對於推動開放資料(Open Data)投入了許多心血，這股「開放」的自由精神持續進行中，然而各大學在校園資訊系統方面，常常無法及時滿足學生或教師的需求，主要原因在於校內的資訊人力有限，為解決此問題本篇論文將提出以OAuth 2.0授權標準來建立校園開放API(Application Programming Interface)平台，此平台將提供有創意之全校師生開發新應用程式，校內資訊人力只要提供開放API資料，期望改善校方自行開發應用程式上的時間與人力成本。

關鍵詞: 系統整合、開放資料、網路服務、網路應用程式介面、開放授權 2.0

Establish campus open API platform with OAuth 2.0

Po-Yao Lan and Jia-Lien Hsu

Department of Computer Science and Information Engineering

Fu Jen Catholic University

Abstract

In recent years, from the public to the government, open government information (Open Data) has become a trend. However, most university campus information systems are often unable to timely meet the information requirements of students or teachers because under limited school's IT technician , in order to solve this problem this paper proposed to establish campus open API(Application Programming Interface) platform with OAuth 2.0 that will provide teachers and students to develop creative new applications, as long as the school's IT technician to provide open API data, expected to improve the school's time and costs on self-development of application.

Keywords: System Integration,Open Data,Web service,Web API,OAuth 2.0

第一章 緒論

1.1 研究背景

開放資料是近年全球熱門的議題，而目前在臺灣也已看到許多政府單位開放資料平台的出現，如台北市政府資料開放平台(http://data.taipei.gov.tw/opendata/)、新北市政府資料開放平台(http://data.ntpc.gov.tw/)、政府開放資料平台(data.gov.tw)等。

然而各大學在校園Open Data方面，僅有少數大專院校舉辦過APP創意競賽，例如：成大的「2014校園瘋雲榜APP創意競賽」[1]、海大的「2014校園APP創意競賽-海洋盃」[2]及高科大「2014年全國大專院校物聯網與行動APP整合創新應用競賽」[3]，這些比賽作品大多是解決學生求學過程中「食」、「衣」、「住」、「行」、「育」、「樂」的各種問題，藉此讓學校未來必須開放更多的資料，才能滿足學生們的各種創新的應用。

本篇論文將以輔仁大學為例，建立一個校園開放API平台，依資料公開屬性，結合OAuth 2.0授權機制，作為學校未來建置Open Data架構參考與實作範例。

1.2 研究動機

目前各大網站Facebook、Google與Amazon皆有各自的開發者平台，在這些平台中提供許多Web API給開發者開發許多創新的應用程式，雖然Web API應用在產業界已發展的非常成熟，但在許多大專院校中的校園資訊系統依然是非常封閉，以致於限制了校園資訊系統許多的創新應用。為了提升原校園資訊系統之應用服務與便利性，將建立一個可以有效管理與授權的API平台，提供給具有開發應用程式之全校師生，來補足原校園資訊系統無法提供的創新應用與服務。基於以上動機，因而研究相關授權機制，OAuth 2.0與WS-Trust 1.4皆是網路常見的開放標準，因OAuth 2.0針對現今各種第三方應用(如：web application、user agent application、native application等)定義出不同的授權流程，在實作方面較為開放與簡單，故本研究中將利用OAuth 2.0裡的標準運作流程，建立一個有授權機制的校園開放API平台，提供有創意之全校師生開發新應用，同時也可減少校方自行開發應用程式上的時間與成本。

1.3 論文架構

接下來的論文內容章節中，第二章會回顧Open Data定義與 OAuth 2.0授權機制，第三章會介紹「校園開放API平台」的系統架構及與OAuth關聯性，第四章會詳細介紹授權伺服器(OAuth Server)與第三方程式(client)的實作，第五章是結論與未來研究方向。

第二章 相關研究

2.1 Open Data 介紹

Open Data**[4]**指的是一種經過挑選與許可的資料，這些資料不受著作權、專利權，以及其他管理機制所限制，可以開放給社會公眾，任何人都可以自由出版使用，不論是要拿來出版或是做其他的運用都不加以限制。

WWW 的發明人之一『李伯納』(Tim Berners-Lee)在 2010 年時曾對 Open Data 發展提出一個五顆星等級的分析**[5]**，其內容為一顆星代表資料放在網路上但無統一格式，二顆星代表可以被機器讀取的結構資料(如：excel 格式)，三顆星代表有前兩者特性外且非特定軟體的資料格式(如：csv 格式)，四顆星代表有前三者特性使用 W3C 的開放標準（RDF 和 SPARQL），讓人或機器可使用 URI 來指定資料，五顆星代表可將你的資料與他人的資料做相互智慧聯結，提供其關聯性。

在本論文中，開放資料將以 4 顆星為設計標準，以利各種第三方應用程式使用。

2.2 OAuth 介紹

2.2.1 歷史沿革

開放授權(OAuth)**[6]**於 2007 年 10 月由 OAuth 討論組發佈標準草案，最後於 2010 年 4 月發表為 OAuth 1.0 協議。

而在在 2012 年 10 月發表 OAuth 2.0 版本，該版本主要簡化上一版在第三方應用的開發程序，同時為 Web 應用、桌面應用、手機及其它電子設備，提供不同的認證流程。

2.2.2 原理概念

OAuth 是一個開放標準，其定義為允許用戶讓第三方應用訪問該用戶在某一網站上存儲的私有的資源(如照片、影片)，而無需將用戶帳號和密碼提供給第三方應用。OAuth 授權機制解決了下列 2 項問題，如下：

- 避免第三方程式濫用資源擁有者(Resource Owner)的帳號與密碼。
- 資源擁有者(Resource Owner)可以有撤銷單一第三方程式的授權。

OAuth 2.0 主要提供程式開發者更簡化的授權流程，以符合各種類型的應用程式完成授權，以下是一個 OAuth 的簡單範例，如圖一

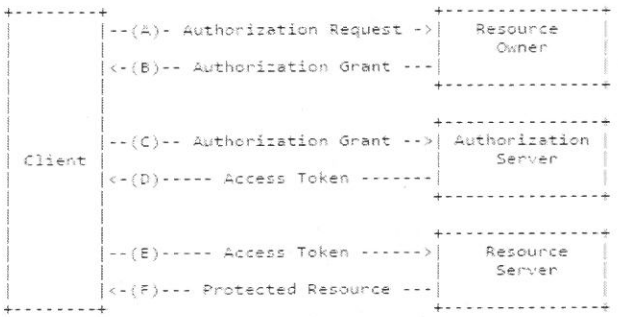
圖一 OAuth 簡單範例

由圖一中可知，授權機制主要由四個角色組成，其分別定義如下：

- 授權伺服器(Authorization Server)- 在資源擁有者(Resource Owner)同意授權後，核發授權令(Access Token)好讓第三戶程式(Client)能暫時存取 Resource Server 上受保護的資源。

- ▶ 資源伺服器(Resource Server)-資源擁有者(Resource Owner)存放受保護資源的伺服器，可以根據授權令(Access Token)來接受請求。
- ▶ 第三方程式(Client)- 代表資源擁有者(Resource Owner)去存取受保護資源的應用程式。
- ▶ 資源擁有者(Resource Owner)- 可以授權第三方程式去存取受保護資源的人。

在了解了四個主要的角色後，我們將介紹這四種角色如何互相溝通，如圖二，說明了OAuht2.0 中各種角色的互動:



圖二 OAuth 2.0 概念協定流程圖 [7]

(A)第三方程式端向資源擁有者請求授權，這個授權請求可以直接向資源擁有者發送，或是間接由授權伺服器來請求。

(B)第三方程式端收到資源擁有者的授權同意憑證(Authorization Grant)，此憑證可至授權伺服器端換取授權令牌(Access Token)。

(C)第三方程式經資源擁有者授權同意後以授權同意憑證(Authorization Grant)至授權伺服器換取授權令牌(Access Token)。

(D)授權伺服器端驗證授權同意憑證(Authorization Grant)，通過驗證就發行授權令牌給第三方程式。

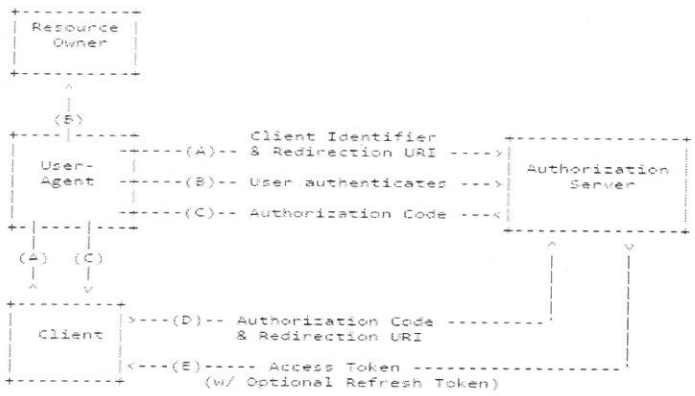
(E)第三方程式端以此授權令牌(Access Token)向資源伺服器要求受保護的資源。

(F)資源伺服器確認授權令牌(Access Token)是否有效，如果有效則讓第三方程式訪問受保護的資源。

2.2.3 四種授權流程介紹

- Authorization Code Grant Flow

  此授權流程以授權同意憑證(Authorization Code)來換取授權令牌(Access Token)，可避免授權令牌(Access Token)曝露在 User-Agent 端點，是四種流程中最安全的授權方式，適用於 web server 類型的第三方程式(Client)，以下是此流程細部說明，如下圖三。



圖三 Authorization Code Grant Flow [8]

(A) 第三方程式(Client)透過資源擁有者的使用者代理(User-Agent)向認證伺服器發出授權請求，並以 Client Identifier(通常是 Client ID) 與 Redirection URI 來認證請求。

(B) 授權伺服器(Authorization Server)收到授權請求，立即透過 User-Agent 詢問是否同意授權。

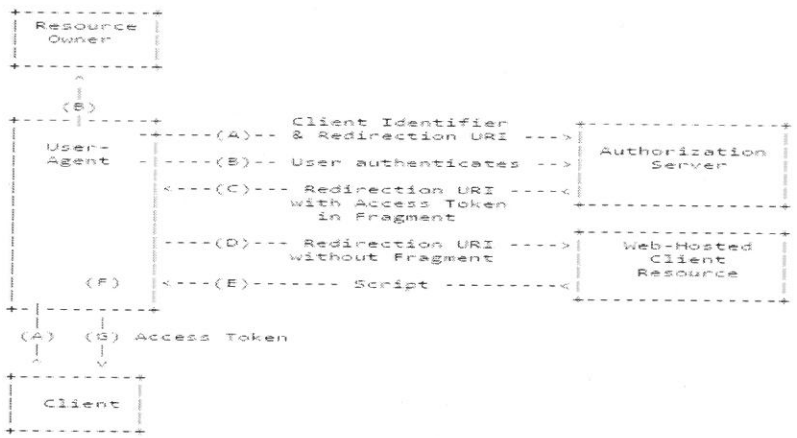
(C) 授權伺器(Authorization Server)收到同意授權後，產生授權同意憑證(Authorization Code)並傳回至第三方程式。

(D) 第三方程式收到授權同意憑證(Authorization Code)後，再以此 Code 與 Redirection URI 向授權伺器(Authorization Server)請求授權令牌(Access Token)。

(E) 授權伺器(Authorization Server)驗證授權同意憑證(Authorization Code)與 Redirection URI 無誤後發送授權令牌(Access Token) 與 Refresh Token 給第三方程式(Client)。

- Implicit Grant Flow

  此種流程與 Authorization code grant flow 最大的不同就是不用授權同意憑證(Authorization Code)去換授權令牌(Access Token)且不發送 Refresh Token，也因此在 User-Agent 端點恐有授權令牌(Access Token)曝露的問題，所以適用於 Public 類型的第三方程式(Client)如：Client Side browser-based，以下是此流程細部說明，如下圖四。



圖四 Implicit Grant Flow [9]

(A) 第三方程式(Client)透過資源擁有者的使用者代理(User-Agent)向認證伺服器發出授權請求，並以 Client Identifier(通常是 Client ID) 與 Redirection URI 來認證請求。

(B) 授權伺服器(Authorization Server)收到授權請求，立即透過 User-Agent 詢問是否同意授權。

(C) 授權伺器(Authorization Server)收到同意授權後，產生授權令牌(Access Token)放在 Fragment 裡，再傳回至 User-Agent。

(D) User-Agent 依 Redirection URI 重新導向 Web-Hosted Client Resource，但不包含 Fragment 內容。

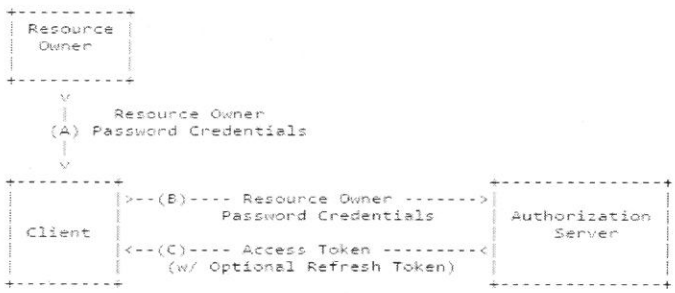
(E) Web-hosted Client Resource 回應 User-Agent 一段 Script。

(F) User-Agent 透過此 Script 來 Decode 出 Fragment 裡的授權令牌(Access Token)。

(G) 將授權令牌(Access Token)傳給第三方程式(Client)。

- Resource Owner Password Credentials Grant Flow

  直接使用 Resource owner 的帳號與密碼去換取授權令牌(Access Token)，此種流程必須是高度信認的第三方程式(Client)採用(如：官方應用程式)，以下是此流程細部說明，如下圖五。



圖五 Resource Owner Password Credentials Grant Flow [10]

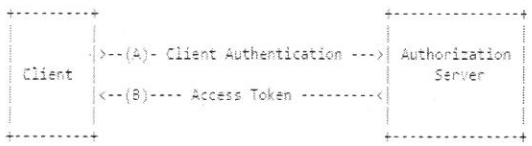
(A) 第三方程式(Client)要求資源擁有者(Resource Owner)先提供帳號與密碼以便授權認證使用。

(B) 第三方程式(Client)以資源擁有者(Resource Owner)的帳號與密碼去授權伺服器(Authorization Server)認證並請求授權令牌(Access Token)。

(C) 通過認證後，產生授權令牌(Access Token)並傳回至第三方程式。

- Client Credentials Grant Flow

  此流程中通常是第三方程式(Client)與資源擁有者(Resource Owner)為同一人，且僅適合在第三方程式是 Server 的類型，以下是此流程細部說明，如下圖六。



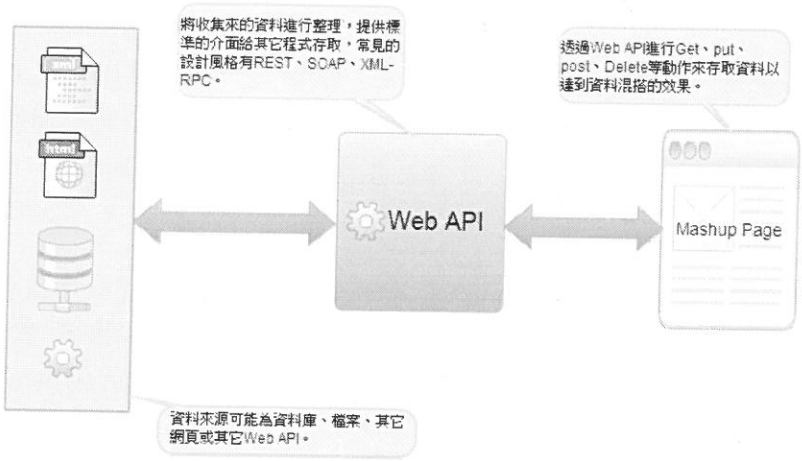
圖六 Client Credentials Grant Flow [11]

(A) 第三方程式(Client)必須先向授權伺服器(Authorization Server)註冊，取得一組 Client ID 與 Secret，此為 Client credential，第三方程式(Client)再以此 credential 向授權伺服器(Authorization Server)請求授權。

(B) 授權伺服器(Authorization Server)收到授權請求，驗證通過後即核發授權令牌(Access Token)。

2.3 OAuth 2.0 與校園開放 API 平台關聯性

2.3.1 傳統 Web API

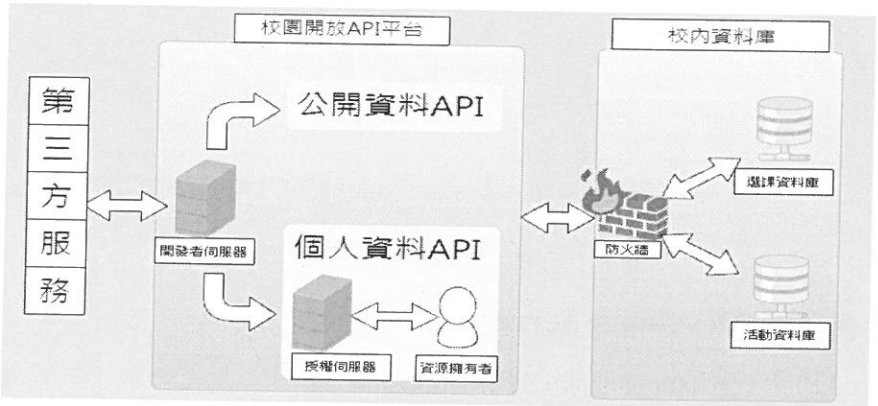


圖七 傳統 Web API 存取架構圖

由圖七可知傳統的 Web API 架構是無法對存取來源做身份控管的，若要達到身份管理必須將存取來源建立身份識別機制，故篇論文將建立一台開發者伺服器來管理所有存取來源身份，提供校方管理人員即時監控 Web API 存取狀況與權限管理。

2.3.2 具授權管理機制的校園開放 API 平台

在校園開放 API 平台中包括「開發者伺服器」、「授權伺服器」與「WebAPI」，其中開發者伺服器提供全校師生註冊自己的開發的應用程式，也提供存取監控與管理；授權伺服器主要負責 OAuth 2.0 流程執行如：token 建立、資料擁有者的授權確認等；Web API 主要分為公開資料 API 與個人資料 API，其中存取個人資料 API 時會結合認證伺服器做 OAuth 2.0 授權動作，所有 Web API 也存放在授權伺服器中，此部份未來可獨立建置，本篇論文中參考校內現行的活動管理系統、選課系統與場地預約系統中資料定義(schema)來模擬建立資料庫以供 Web API 做資料存取。平台身份認證採輔大校內現有單一帳號簽入機制(LDAP)來實作之，故僅供校內師生註冊第三方應用，未來其它校若也想建立該平台僅須修改成校內現有的身份認證機制即可，以下是校園開放 API 平台架構，如圖八。



圖八 OAuth 2.0 與校園開放 API 平台概念圖

第三章 校園開放 API 平台系統設計架構

3.1 設計動機

秉持 Open Data 的公開與分享之精神，建立一個輔大開放 API 平台架構，提供本校資訊中心實作參考，期未來全校教職員工生可以透過此平台下載資料或開發新服務，且應用於學業、研究或校內行政工作上，來滿足各種不同的資訊需求。

3.2 實作工具簡介

為快速實作校園開放 API 平台(Campus Open API Platform)中複雜的授權機制，且希望相容於 Asp.net 開發環境並為 Open Source 類別庫，經以上考量故決定使用 DotNetOpenAuth 類別庫**[12]**發之，此類別庫提供完整的第三方程式(Client)、認證伺服器(OAuth Server)與資源存取端的實作範例，解決了開發上許多的困難，以下是主要套件清單如表一，下載 OAuth 範例原始碼，如圖九。

表一 套件清單

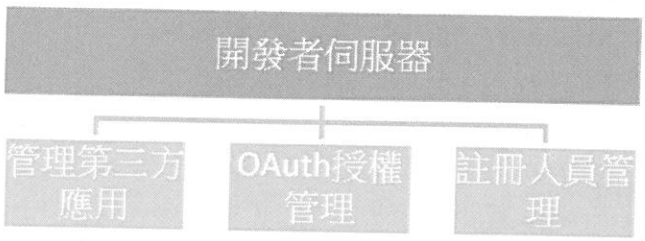
套件名稱	說明
DotNetOpenAuth.OAuth2 Core	DotNetOpenAuth 必要套件
DotNetOpenAuth.OAuth2.AuthorizationServer	Authorization Grant 與 Access Token 操作
DotNetOpenAuth.OAuth2.Client	建立 Client 請求
DotNetOpenAuth.OAuth2.ResourceServer	提供 ResourceServer 驗證操作
備註：所有套件請至 https://www.nuget.org/ 下載	

圖九 下載 OAuth 範例原始碼

3.3 開發者伺服器(Developer Server)

校園開放API平台(Campus Open API Platform)結合輔大已有的單一帳號簽入的機制(LDAP)，提供全校教職員工生註冊自己的第三方程式，統一存放在開發者伺服器中，以下是開發者伺服器設計參考說明。

3.3.1 伺服器功能



圖十 開發者伺服器功能

- 管理第三方應用

  提供「系統管理者」管理已註冊之第三方程式，若為「一般使用者」僅能管理自己的第三方應用程式。

- OAuth 授權管理

  提供「系統管理者」管理 Access Token 授權狀況。

- 註冊人員管理

  提供系統管理者維護人員權權，在本伺服器中使用者分為「一般使用者」與「系統管理者」兩種角色。

3.3.2 伺服器環境需求

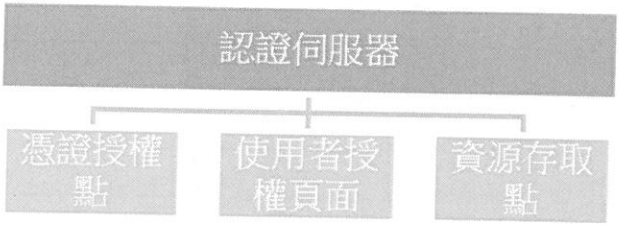
表二 環境需求

項目名稱	規格說明
作業系統	Windows Server 2008
Web Server	IIS 7.0
資料庫	Sql Server 2008 R2
開發工具	Visual Studio 2010
類別庫	DotNetOpenAuth
編譯器版本	.NetFrameWork 4

3.4 認證伺服器(OAuth Server)

主要負責發行 Access Token、Resource Owner 授權認證與存放 Web API 三大功能，以下是認證伺服器設計參考說明。

3.4.1 伺服器功能



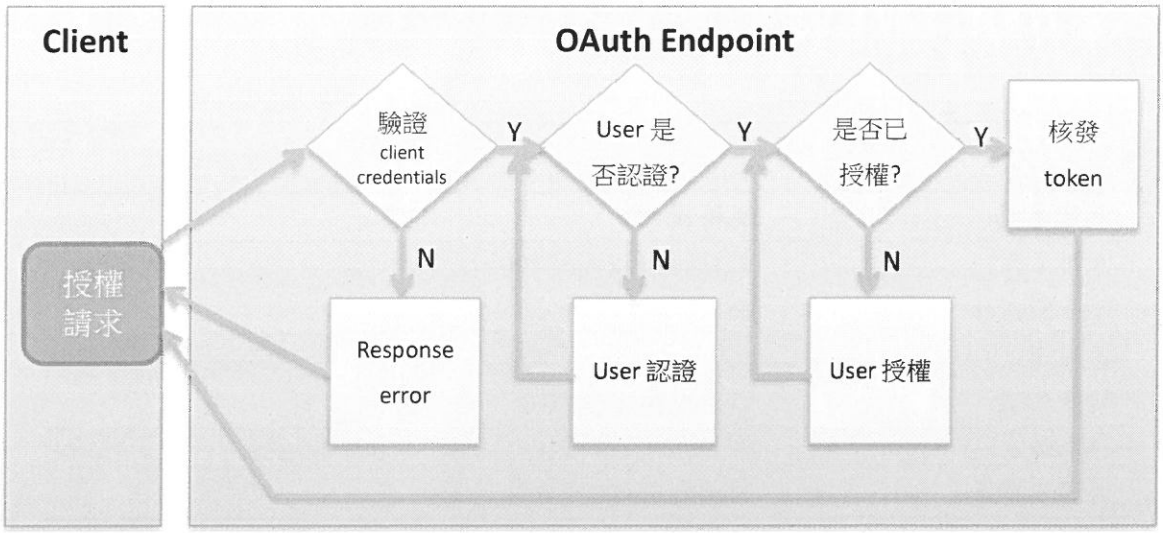
圖十一 認證伺服器功能

- 憑證授權點(Authorization Endpoint)：接受 Client 端 Access Token 請求，該程式虛擬碼，如表三。

表三 憑證授權點虛擬碼

```
START
Input(Object)：client request(clientID、APIKey、CallBackURI、Scope)
Ouput(String)：Access Token
Authorization Endpoint (client request)
If clientID && APIKey && CallBackURI exist in Developer Server THEN
   initialize Token
   IF (Token Authorized) THEN
      Return Access Token to CallBackURI
   ELSE
      Redirect to User Authorization Page
Else
   Return Exception Illegal request
END
```

在 OAuth Endpoint 中驗證流程中主要有三個判斷，判斷一為驗證 client credentials，主要是驗證請求授權的 client 是合法性；判斷二為 User 是否認證，主要是取得 Resource Owner 的授權；判斷三為是否已授權，主要是檢查 token 的狀態是否已取得授權，其驗證流程如圖十二，每一判斷所需參數如表四。



圖十二 授權判斷流程

表四 授權判斷流程驗證參數表

重要判斷 / 參數名稱(說明)	驗證 client credentials	User 是否認證	TOKEN 是否已授權
userID(LDAP 帳號)		V	V
userPWD(LDAP 密碼)		V	
redirect uri(Callback rui)	V		
clientID(Client ID)	V		V
apikey(存取 Web API 使用的 API Key，由校園開放 API 註冊取得)	V		
State(授權狀態)			V
issueDate(Token 建立時間)			V

Token(token 字串)			V
Scope(存取範圍)	V	V	V

- 使用者授權頁面：提供 User 認證與授權，該程式虛擬碼如表五。

表五 使用者授權頁面虛擬碼

```
START
Input：none
Ouput：none
User Authorization Page
IF User is logined Then
  IF authorize the client THEN
     UPDATE state=1 and redirect to Authorization Endpoint
  ELSE
     UPDATE state=2 and redirect to Authorization Endpoint
ELSE
  Redirect to login page
END
```

- 資源存取點(API Endpoint)

接受 Client 端依 Access Token 請求資料，通過驗證後，以 JSON[13]資料格式回應 Client 端，該程式虛擬碼，如表六。

表六 資源存取點虛擬碼

```
START
Input(Object)：client request (clientID、APIKey、Access Token)
Output(JSON)：Web API data
Web API Endpoint(client request)
IF clientID && APIKey && Access Token exist in developer server THEN
  IF Access Token of issueDate <10 min THEN
     RETURN Web API data
  ELSE
     Return Exception code(Access Token time out)
ELSE
  Return Exception code(Illegal request)
END
```

3.4.2 伺服器環境需求

表七 環境需求

項目名稱	規格說明
作業系統	Windows Server 2008
Web Server	IIS 7.0
資料庫	Sql Server 2008 R2
開發工具	Visual Studio 2010
類別庫	DotNetOpenAuth
編譯器版本	.NetFrameWork 4

3.5 校園開放 API 平台 API 種類

在參考了本校既有的「活動管理系統」、「場地借用系統」與「獎學金申請系統」資料庫設計，本篇論文將 Web API 分成「公開資料」與「個人資料」兩類，總共設計了 3

支 Web API，提供未來設計之參考，以下是 Web API 設計說明，如表八。

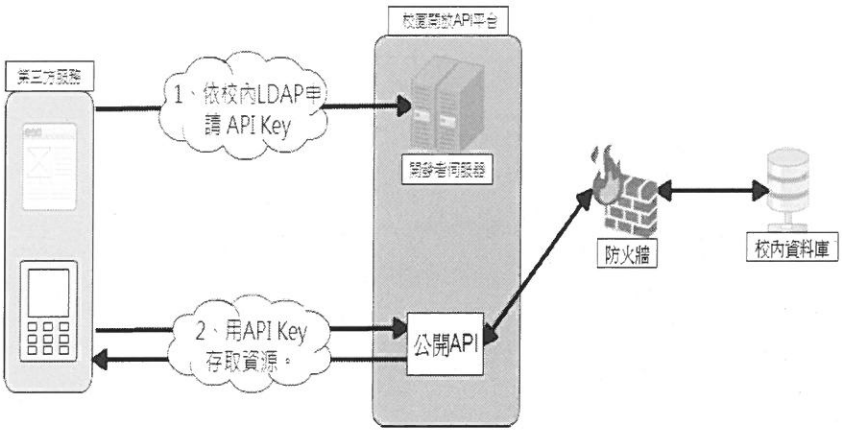
表八 Web API 設計說明

Web API 種類	必要輸入參數	說明	Web API 設計範例
公開資料	1.clientID 2.apikey	第三方程式須先到開發者平台註冊取得 apikey 才可至 Web API 存取資料。	1.活動資訊(詳如表 9)。 2.場地借用資訊(詳如表 10)。
個人資料	1.clientID 2.apikey 3.oauth_token	第三方程式須先到開發者平台註冊取得 apikey，而 oauth_token 參數為認證伺服器自動產生。	3.個人當學期選課資訊。

共 3 支

3.5.1 公開資料 API

- 存取流程



圖十三 公開資料 API 存取流程圖

由圖十三可知，存取公開資料僅需至校園開放 API 平台註冊即可存取，此部份資料以無涉及個人資料或機密為主，以下是存取步驟：

1、透過 LDAP 認證登入至輔大開放 API 平台申請 API Key。

2、使用 Client ID 與 API Key 至 Fju APIs 訪問資源。

本篇論文依校內原系統實作出「活動資訊」、「場地借用資訊」、「獎學金資訊」與「學生社團資訊」，共 4 支公開資料 Web API，以下詳細規格。

表九 課外活動 Web API

活動資訊					
參考連結：http://slme.dsa.fju.edu.tw/fjuoauthsrv/fjusrv/fjuActivity.ashx					
功能描述：提供即將舉辦之前 30 筆活動資訊。					
輸入參數					
	欄位名稱	欄位說明	型態	長度	備註

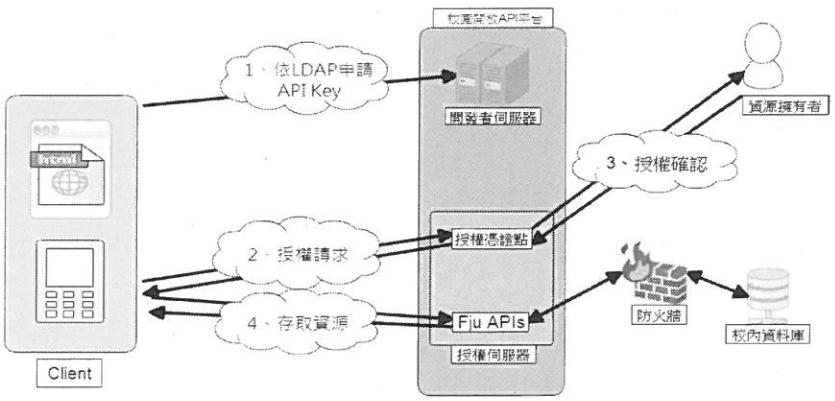
1.	clientID	第三方程式 ID	String		LDAP 登入帳號
2.	apikey	存取金鑰	String		由開發者伺服器產生
輸出欄位					
	欄位名稱	欄位說明	型態	長度	備註
1.	ActivityID	活動 ID	int		例：2
2.	ActivityName	活動名稱	nvarchar	100	例：期末成果發表
3.	LessonID	場次 ID	int		例：1
4.	LessonName	場次名稱	nvarchar	100	例：103 學年第一學期
5.	ActivityStr	報名起始時間	datetime		例：2015-02-10T08:00:00
6.	ActivityEnd	報名結束時間	datetime		例：2015-02-10T16:00:00
7.	LessonStr	場次起始時間	datetime		例：2015-02-10T08:00:00
8.	LessonEnd	場次結束時間	datetime		例：2015-02-10T16:00:00
9.	location	地點	nvarchar	50	例：聖言樓 1 樓
10.	DepCN	主辦單位	nvarchar	50	例：資訊工程學系
11.	TypeName	活動類型	nvarchar	30	例：學術研討

表十 場地借用資訊 Web API

場地借用資訊					
參考連結：http://slme.dsa.fju.edu.tw/fjuoauthsrv/fjusrv/fjuSpace.ashx					
功能描述：提供場地借用情況資訊。					
輸入參數					
	欄位名稱	欄位說明	型態	長度	備註
1.	spc_id	場地代碼	String		例：1,2,3,4…
2.	clientID	第三方程式 ID	String		LDAP 登入帳號
3.	apikey	存取金鑰	String		由開發者伺服器產生
輸出欄位					
	欄位名稱	欄位說明	型態	長度	備註
1.	term	學年學期	char	4	例：1001
2.	spc_name	場地名稱	nvarchar	50	例：谷欣廳
3.	spc_info	位置資訊	nvarchar	100	例：野聲樓 1 樓
4.	spc_type	場地類型	char	2	01➔一般教室 02➔電腦教室 03➔會議室/演講廳 04➔室內運動場
5	spc_quota	場地容納人數	int		例：120
6.	spc_adminDpt	權責單位	nvarchar	20	例：XX 單位
7.	spc_adminTel	權責單位電話	nvarchar	10	例：2905XXXX
8.	spc_user	借用單位	nvarchar	20	例：XX 單位
10.	spc_StartTime	借用起始時間	DateTime		例：2015-04-10T08:00:00
11.	spc_EndTime	借用結束時間	DateTime		例：2015-04-10T10:00:00
12.	spc_Fee	場地費用	INT		例：0
13.	spc_Cond	借用條件	nvarchar	50	例：限校內單位
14.	spc_memo	備註	nvarchar	100	例：XXXX

3.5.2 個人資料 API

- 存取流程



圖十四 個人資料 API 授權存取流程圖

由圖十四可知，存取個人資料需至校園開放 API 平台註冊且經資源使用者同意授權才可存取，本設計以 OAuth 2.0 中 Client Credentials Grant Flow 授權標準，以下是授權存取步驟：

1、透過 LDAP 認證登入至輔大開放 API 平台申請 API Key。

2、使用 Client ID 與 Secret 至 OAuth 伺服器請求授權，此階段 OAuth Server 會同時驗證請求來源 uri 的合法性。

3、OAuth Server 轉址授權頁給 Resource Owner 授權。

4、經 Resource Owner 同意授權後以 Access Token 至 Fju APIs 訪問資源。

本篇論文依校內原系統實作出「個人當學期選課」，共 1 支個人資料 Web API，以下是 WEB API 詳細規格。

表十一 個人當學期選課 Web API

個人當學期選課					
參考連結：http://slme.dsa.fju.edu.tw/fjuoauthsrv/fjusrv/fjuScore.ashx					
功能描述：提供查詢個人在校當學期選課資料。					
輸入參數					
	欄位名稱	欄位說明	型態	長度	備註
1.	clientID	第三方程式 ID	String		LDAP 登入帳號
2.	apikey	存取金鑰	String		由開發者伺服器產生
3.	oauth_token	授權金鑰	String		由認證伺服器產生
輸出欄位(JSON 格式)					
	欄位名稱	欄位說明	型態	長度	備註
1.	term	學年學期	char	4	例：1031
2.	stuno	學號	String	9	例：498515xxxx
3.	sbjName	科目名稱	nvarchar	100	例：計算機概論
4.	sbjNo	科目代碼	nvarchar	10	例：1234567890
5.	SessionStr	起始節次	Char	2	D0➔07:10~08:00 | D3➔11:10-12:00 D4➔12:40-13:30 | E1➔18:40-19:30 E2➔19:35-20:20 E3➔20:30-21:20 E4➔21:25-22:10
6.	SessionEnd	結束節次	Char	2	
7.	location	上課地點	nvarchar	50	例：SF650
8.	tchCN	老師姓名	String	20	例：XX 老師

第四章 第三方程式實作

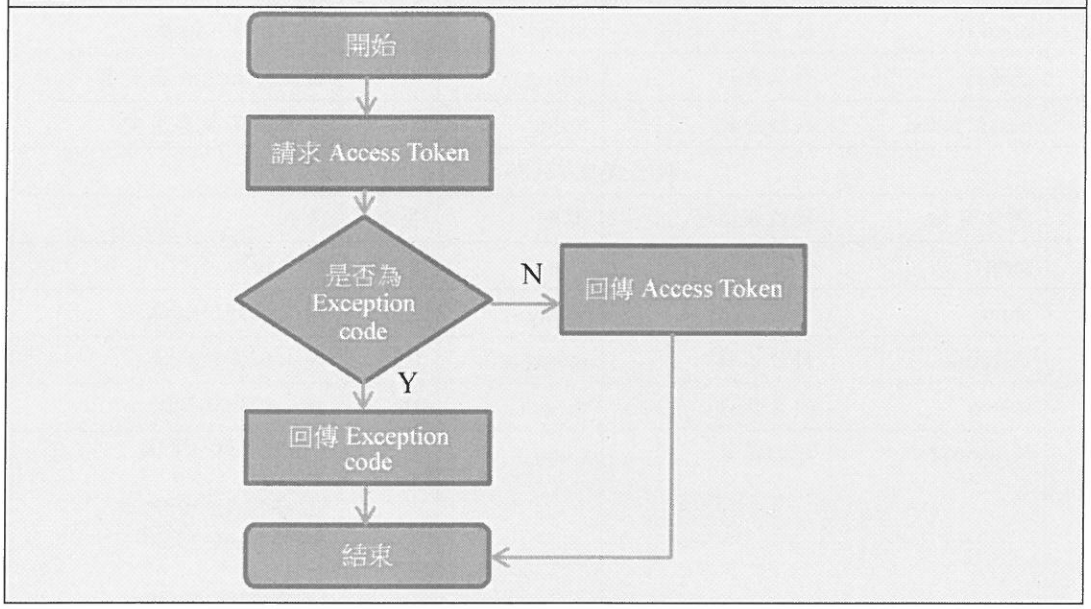
4.1 活動暨個人選課檢視程式

本程式使用到 JQuery FullCalendar 套件**[14]**呈現前端的行事曆，以學校「活動資訊」與「個人當學期選課」API 做為資料來源，整合於自行開發的行事曆網頁上，透過月曆的呈現方式，提供使用者方便、快速判斷是否參加活動。

4.1.1 程式流程與虛擬碼

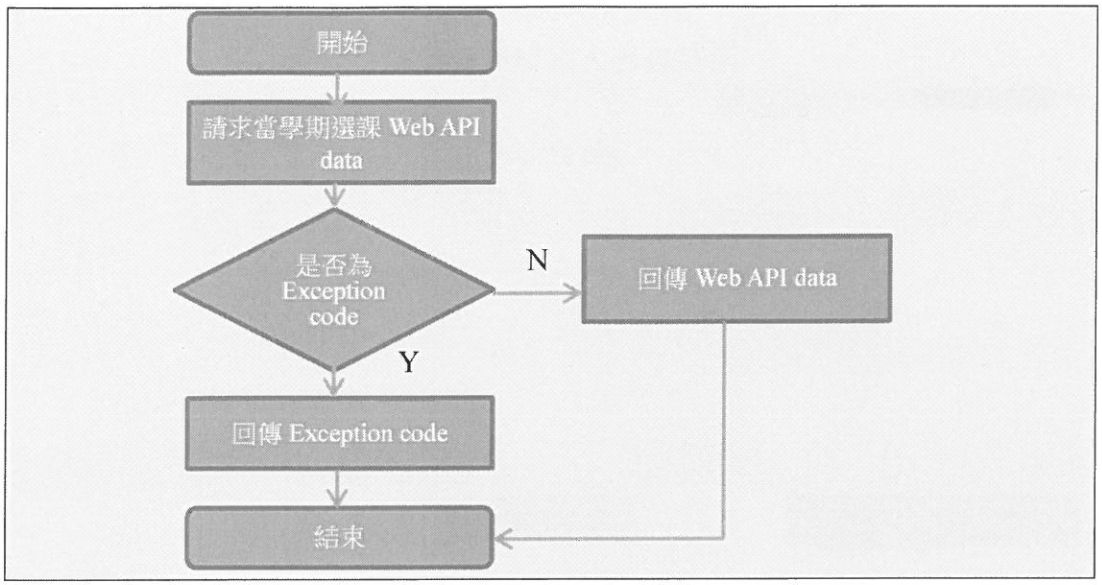
表十二 client 端請求 Access Token 虛擬碼與流程

```
START
INPUT: clientID、APIKey、CallBackURI、Scope
OupPut:Access Token/Exception code
Token Request()
   Result=Request to Authorization Endpoint with clientID、APIKey、CallBackURI、Scope
   If (Result <> Exception code) THEN
      Return Access Token
   ELSE
      Return Exception code
END
```



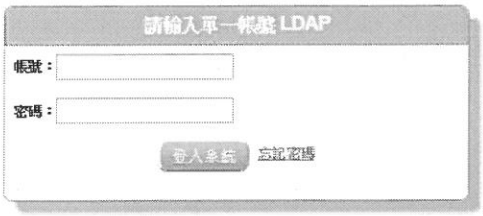
表十三 client 端請求 Web API(當學期選課) 虛擬碼與流程

```
START
INPUT: clientID、APIKey、Access Token
OupPut:Web API data
Web API Request()
   Result=Request to Web API Endpoint with clientID、APIKey、CallBackURI、Scope
   If (Result<> Exception code) THEN
      Return Web API data
   ELSE
      Return Exception code
END
```

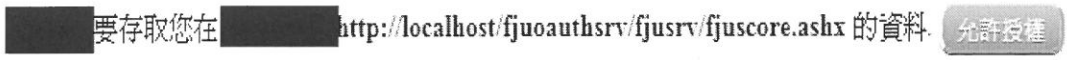


4.1.2 程式畫面

圖十五 未授權活動暨個人選課



圖十六 授權登入



圖十七 授權確認

圖十八 已授權活動暨個人選課

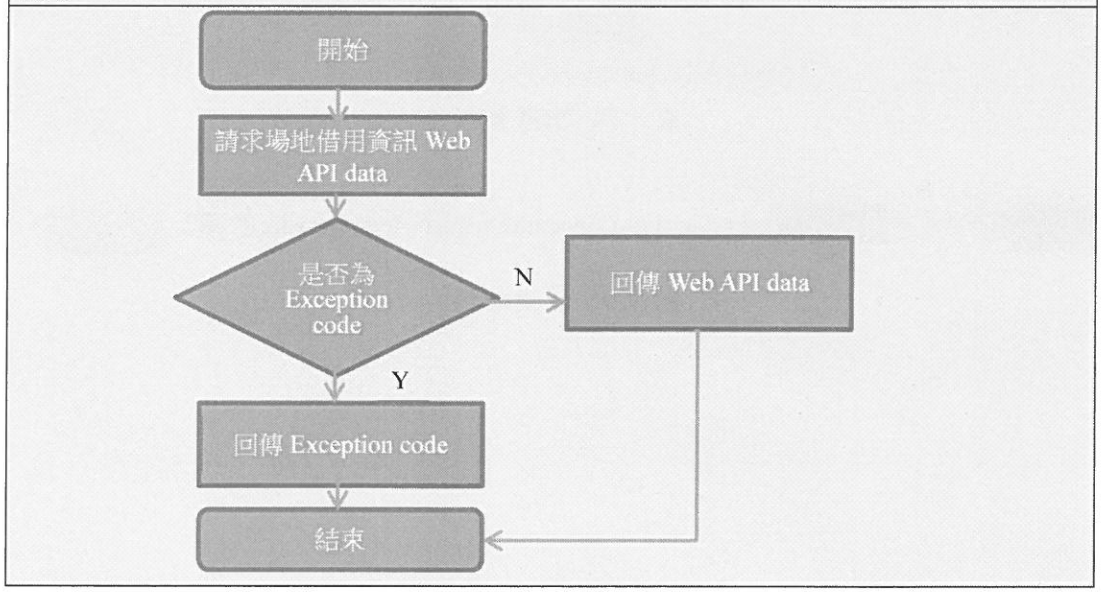
4.2 校內場地借用狀況程式

本程式以「場地借用資料」API 做為資料來源，以地圖的方式呈現於網頁上，提供欲借用場地之師生檢視全校各場地借用情形，間接增加閒置場地之使用率。

4.2.1 程式流程與虛擬碼

表十四 client 請求 Web API(場地借用資訊) 虛擬碼與流程

```
START
INPUT: clientID、APIKey、spc_id
OupPut:Web API data
Web API Request()
Result=Request to Web API Endpoint with clientID、APIKey、spc_id
  If (Result<> Exception code) THEN
    Return Web API data
  ELSE
    Return Exception code
END
```



4.2.2 程式畫面

輔仁大學-場地使用程式

圖十九 校內場地借用首頁

圖二十 校內場地借用週次表

第五章 結論

5.1 研究成果

目前在各學校中教務、學務或行政事務大多已有內部的資訊系統，但這些系統大多是內部的封閉系統，若要做系統升級或修改功能時常常花費學校許多資訊人力與時間成本，故我們提出結合 Open Data、Web API 與 OAuth 2.0 等技術建立一個「校園開放 API 平台」架構，讓校內非核心的資訊應用給校內有開發能力的教師或學生，資訊人員僅須維護好 Web API 的資料品質，以此來減輕各校資訊人力不足的困擾。

5.2 未來展望

本篇論文結合OAuth 2.0標準建立一個可管理的「校園開放API平台」，而未來還有許多深入的細節值得研究，以下是我們整理出以後可再深入討論的主題方向

- OAuth Server 安全性問題(如：Covert Redirect**[15]**、denial-of-service attack**[16]**)。
- 在現有的OAuth 2.0授權機制僅以scope來控制存取範圍，但實務上常常一個校園Web API必須依不同身份權限的人回應不同的資料，此部份須再擴充OAuth 2.0 可針對角色授權才能比較符合實務需求。(如：畢業生使用權限、有擔任行政職的教師或有擔任幹部的學生)。
- 「校園開放API平台」實現跨校Web API的資訊整合(如：跨校選課系統)，以下兩點方向再做進一步研究。
  - 可於「開發者伺服器」中Web API擴充「校學屬性」提供OAuth授權流程控制，達到資訊整合的效果。
  - 針對各校身份認證方面(開發者、使用者或管理者)可整合教育部成立的TANet 無線網路漫遊機制，來達到跨校認證的效果。

參考文獻

[1] 國立成功大學計算機與網路中心，2014 校園瘋雲榜 APP 創意競賽，103 年 12 月 16 日，取自 http://app-fun.web2.ncku.edu.tw/files/11-1040-599.php?Lang=zh-tw。

[2] 國立臺灣海洋大學，2014 校園 APP 創意競賽-海洋盃，2014-04-30，取自：http://app2014.ntou.edu.tw/bin/home.php。

[3] 國立高雄第一科技大學電資學院電子工程系、資通訊服務創新產業碩士班，2014 年全國大專院校物聯網與行動 APP 整合創新應用競賽，103 年 9 月 01 日，取自：http://www1.ord.nkfust.edu.tw/files/13-1004-33517-1.php?Lang=zh-tw。

[4] Open Knowledge,http://opendatahandbook.org/guide/zh_TW/what-is-open-data/.

[5] Tim Berners-Lee, Linked Data, 2006-07-27，取自：http://www.w3.org/DesignIssues/LinkedData.html.

[6] Internet Engineering Task Force,IETF,April 2010, from: http://zh.wikipedia.org/wiki/OAuth.

[7] OAuth, Internet Engineering Task Force (IETF) Request for Comments: 6749, October 2012, from: https://tools.ietf.org/html/rfc6749#section-1.2.

[8] OAuth, Internet Engineering Task Force (IETF) Request for Comments: 6749, October 2012, from: https://tools.ietf.org/html/rfc6749#page-24.

[9] OAuth, Internet Engineering Task Force (IETF) Request for Comments: 6749, October 2012, from: https://tools.ietf.org/html/rfc6749#page-31.

[10] OAuth, Internet Engineering Task Force (IETF) Request for Comments: 6749, October 2012, from: https://tools.ietf.org/html/rfc6749#page-37.

[11] OAuth, Internet Engineering Task Force (IETF) Request for Comments: 6749, October 2012, from: https://tools.ietf.org/html/rfc6749#page-40。

[12] Jan Fajfr, DotNetOpenAuth and Url rewriting (in Azure), Wednesday, August 17, 2011, from: http://hoonzis.blogspot.tw/search/label/OAuth.

[13] Crockford, Network Working Group ，Request for Comments: 4627, July 2006, from: http://www.ietf.org/rfc/rfc4627.txt.

[14] Adam Shaw, jQuery Project，April 2009, from: http://fullcalendar.io/.

[15] Wang Jing, OAuth Security Advisory: 2014.1 "Covert Redirect", 04-05-2014, from: http://oauth.net/advisories/2014-1-covert-redirect/.

[16] Jelena Mirkovic, Sven Dietrich, David Dittrich, and Peter Reiher " Internet Denial of Service: Attack and Defense Mechanisms (Radia Perlman Computer Networking and Security)," Publisher Prentice Hall PTR Upper Saddle River, NJ, USA ©2004，ISBN：0131475738.