Building a Smartphone App by Using Service-Oriented Computing and Social Network APIs for Supporting Check-in Activities

Pin-Fan Lee
Institute of Technology Management, National Chung Hsing University
g100026108@mail.nchu.edu.tw

Shuchih Ernest Chang
Institute of Technology Management, National Chung Hsing University
eschang@dragon.nchu.edu.tw

Fei-Fei Cheng
Institute of Technology Management, National Chung Hsing University
ffcheng.ec@dragon.nchu.edu.tw

Abstract

Along with the progress in communication technology and the rapid growth of mobile device, many software companies and individual developers have started to develop a variety of mobile applications. However, the resource that individual developers can get is far less than a software company. Therefore, it is important for individual developers to utilize the free API resources available on the Internet for harvesting the benefit of the cloud computing. This research explores how developers build a check-in application based on Facebook APIs in service-oriented computing (SOC) environment. The design and implementation of a smartphone application using the proposed approach will be described in this article. Moreover, by comparing with some other smartphone applications which also provide check-in functions, the advantages and potential value of our approach to integrating social network APIs and SOC concept will be discussed.

Keywords: Smartphone Application, Mobile Device, Social-Oriented Computing, Check-in, Social Network

I. INTRODUCTION

Innovation in mobile device has pushed ahead the growth of mobile applications with exceedingly rapid speed. Many companies are heading to the mobile application industry either for providing more services to attract more customers via mobile applications or creating more revenue. However, not only software companies are interested in this market

with huge potential value, but more and more independent and freelance developers are getting interested in developing mobile applications (Holzer, and Ondrus 2009).

Naver, one of the biggest search portal sites in Korea, is a good example of how an Internet service company shifted to mobile app industry successfully; it's communication application, LINE, has already been downloaded 20 million times by Feb, 2012, according to the download counts of Android Market. LINE provides real-time exchange messages and voice calls service for free, which becomes a great substitute for SMS message. It seems to be easy for such a big company to develop the service and support such a big amount of users. However, for individual developers, it is hard to develop such an interactive and real time mobile application with limit resource, implying that it is difficult for individual developers to compete with big company.

Hence, in this competitive environment, it is important for individual developers to leverage as much resources as possible from the Internet to support their applications. The service-oriented computing (SOC) appear to be a good solution for individual developers. SOC is the computing model that utilizes services as fundamentals for developing applications/solutions (Papazoglou 2003). Based on the aforementioned reasons, this work expects to present a simple and flexible way to build a smartphone application which benefits from SOC. In this work, Facebook API was used to build connections between mobile applications and social networks. By querying the Facebook API, developers are able to retrieve user's basic information (Name, E-mail, Education, history usage data, friend list, etc.) from Facebook, thus saving them a lot of efforts without building their own services.

The proposed approach of this article was implemented on a smartphone application, Che-Check. Che-Check is an informative check-in application that provides check-in service and location-based service (LBS). By querying the Facebook API, Che-Check users do not need to enter any information for registration. Furthermore, users can leave message, check-in, view check-in history, and send message to friends easily. On the other hand, user can retrieve the place information (location, address, phone number, etc.) and see "Where're the cool places nearby," "Where's the hottest place for check in."

Consequently, in order to explore the advantages and potential value of the proposed approach which integrates Facebook API and SOC concept, a discussion among Che-Check and two other famous Check-in smartphone applications will be conducted.

II. FACEBOOK API

Facebook has been an extremely successful social network service (SNS) since it first launched in 2004. It has already got 80 million users over the world by Feb, 2012, according to the analysis by *checkfacebook.com.* Facebook becomes an easiest way for people to find someone they may already know or get to know from the Internet by searching on Facebook. It has been exclusively acknowledged as a resource for locating people (Scale 2008). Facebook has successfully built strong connections between users and third party web sites.

Social graph is the core concept that Facebook uses to connect different identities, communities, and conversation in the Facebook at first. However, in order to include third party web sites and pages that people liked throughout the web, the open graph concept was then introduced in 2010. Combining the idea of social graph and open graph, not only the strong connections between people and communities are built, but almost every web site on the Internet can be linked to this huge social network in one simple click of "Share" or "Like."

Facebook released Facebook application programming interface (API) in 2007. Since Facebook has huge user base, it has become a good resource for third party developers (Kourtellis, Finnis, Anderson, Blackburn, Borcea, and Iamnitchi 2010). Many third party developers have approached to connect their applications and Facebook API in order to get user's information quickly in a single call. They are allowed to use API methods/functions to retrieve a variety of information about users, such as full user profiles, list of friends, networks, groups, and events that users are attending (Pietiläinen, Oliver, LeBrun, Varghese, and Diot 2009; Ko, Cheek, Shehab, and Sandhu 2010).

2.1 FQL (Facebook Query Language)

There are two methods that developers can use to query Facebook API, and these two methods are Graph API and FQL (Facebook Query Language). The Graph API provides a consistent and simple view of the Facebook social graph, evenly representing objects in the graph, while FQL enables developers to query data via SQL-style interface.

In this research, FQL is applied because FQL provides advanced features which are not provided by Graph API. For instance, developers are able to do multiple queries in just one FQL call, thus allowing developers to fetch data from the first query and use it in the second query and so on. User profile information that developers can access by using FQL is shown in Table 1. Other tables (such as user's comment, check-in, friends, family, photo, stream, etc.) that can be queried by FQL are listed on Facebook developer page.

Table 1. Elements of the user profile table that can be accessed by Facebook API.

Name	Type	Description
uid	int	The user ID of the user being queried.
username	string	The username of the user being queried.
first_name	string	The first name of the user being queried.
middle_name	string	The middle name of the user being queried.
last_name	string	The last name of the user being queried.
sex	string	The gender of the user being queried.
email	string	A string containing the user's email address
education	array	A list of user's education history
languages	array	The user's languages.

2.2 Permission

Social network users always have their personal information (including information related to contact information, user identification number, E-mail address, and wall posting)

on social network websites (Lewis, Kaufman, Gonzalez, Wimmer, and Christakis 2008). Although there are privacy policies to protect personal information, attackers might still be able to break this barrier by controlling other compromised account (Rabkin 2008). In order to enhance the security protection, whenever a user logs on to an application at the first time, the user will go through an authentication process that asks whether the user wants to grant specific permission requested by the application (shown in Figure 1). The amount of user information available to the application depends on the permissions granted by users.

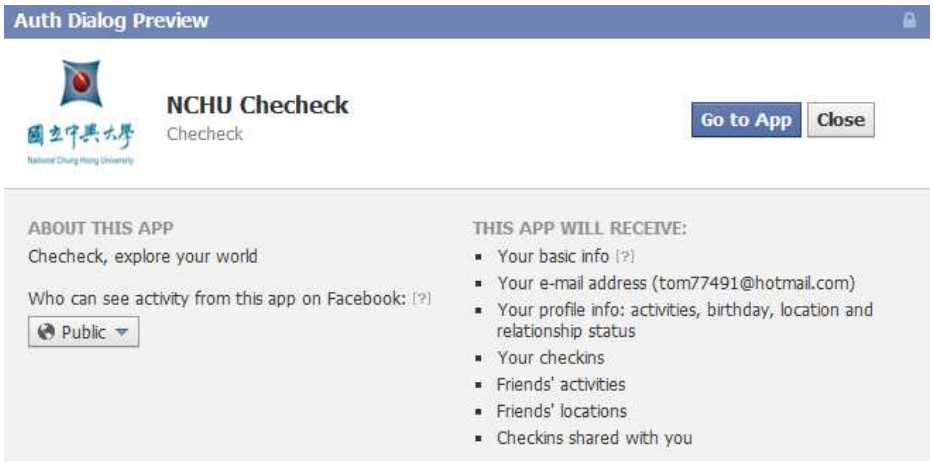

Figure1. Dialog for granting permissions by a user to an application.

III. ARCHITECTURE AND DESIGN

Facebook API provides an always up-to-date data pool for application developers. This huge data pool is available for web application developers and mobile application developers, and the way to fetch data is getting easier. Obviously, it becomes a good choice for developers who have only limited resource. In most cases, application developers use Facebook API in order to fetch user's basic information, such as profile, contact information, and photos. However, there is still a lot of valuable information that developers can get from this big data pool, and there are more potential applications that are worthy to discover.

In this work, taking the advantages of SOC, our architecture design aims to utilize Facebook API more efficiently and more comprehensively, consequently providing individual developers a simple and flexible way to build an informative smartphone application without building needed local supporting utilities. As shown in Figure 2, the client application is able to get a variety of information by sending FQL request to Facebook API. After receiving queries from client side, the API will then fetch the specific table that is described in the received query message and send response back to client. Therefore, application developers do not need to build up a database for storing all the information about users. Developers can choose to query FQL and fetch the "user" table in Facebook database whenever they need to get specific user's information such as E-mail or Name. Moreover, to query the user's data

whenever the client needs it keeps the received information always up-to-date. There is no need to check whether the user's data stored in his/her own database is out of date or not.

However, some developers may like to store extra information about users or the client application, such as user register date, preference, and credit earned. Developers just need to create one or a few tables in their own database for storing such extra information, and the number of table stored in the database varies based on different purposes of developers. Compared with the large amount of tables stored in Facebook database, only one or a few tables is needed in developer's database. As shown in Table 2, Facebook API provides more than 60 tables which are available to developers. In addition, developers do not need to maintain huge database, saving plenty of time, money, network traffic, and effort for them.

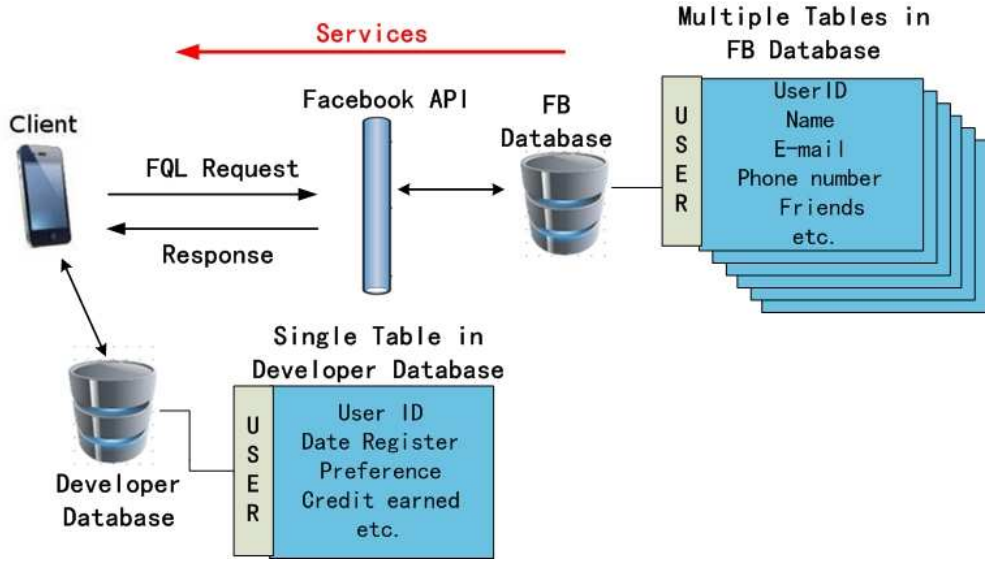

Figure 2. Architecture design with service-oriented computing (SOC) concept.

Table 2. Comparison of tables stored in developer database and Facebook database.

Name	Developer's database	Facebook API database
Number of Tables in database	one or a few (Alterable)	63 (Unalterable)
Updatedness	low	high (always up-to-date)
Maintenance	need maintenance	no maintenance needed
Query Language	SQL	FQL
Permission	no need	need

IV. IMPLEMENTATION OF CHE-CHECK

In this section, the process of implementing the proposed approach on the mobile application, Che-Check, will be illustrated in detail. The first step is to build the connection between Che-Check and Facebook by using Facebook API. After the connection is made, then start to develop Che-Check application based on the Facebook mobile application SDK. There are two versions (iOS and Android) of Facebook SDK released on the Facebook Developer page. The official Android version of Facebook SDK is used in this research.

4.1 Connect mobile application with Facebook

Before being able to connect to Facebook API, a Facebook application, named NCHU Che-Check in this case, must be created on the Facebook developer page. NCHU Che-Check will be used to integrate with the Che-Check smartphone application. NCHU Che-Check is the bridge application between Facebook and Che-Check, it will be used to record the basic information and identification of the Che-Check. Figure 3 shows the information need to be entered in Facebook application page.

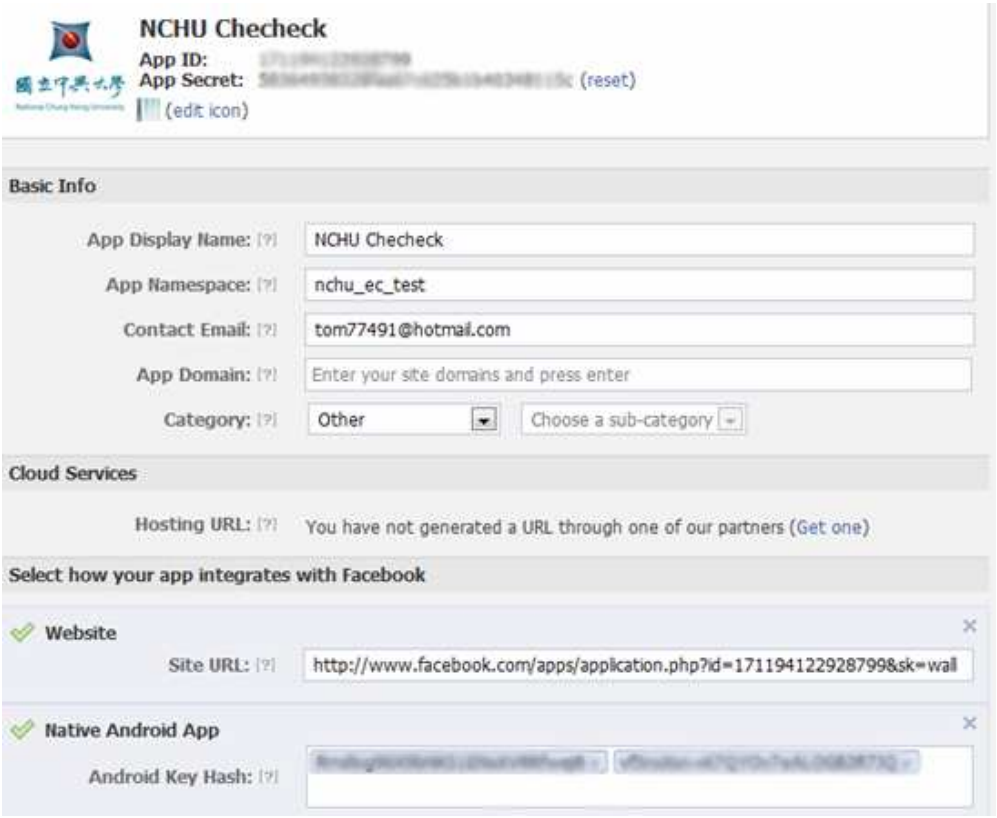

Figure 3. The information required in Facebook application page.

In the Facebook application page, the APP ID represent the Facebook application created on Facebook, this APP ID is needed when integrating Facebook Android SDK into the code. In this case, it specifies the bridge application, NCHU Che-Check. On the other hand, in order to enhance the security of mobile application, Facebook requires developers to fill in the Android Key Hash, which is the signature used to sign the mobile application. Android Key Hash can be generated by running the keytool which comes with the Java JDK.

4.2 Authentication process

After the connection setting is done, Che-Check users will be able to log in with Facebook account in an official Facebook login page. After Che-Check users login with Facebook account, users will be protected under the security policies of Facebook. The authentication process for first time users is needed. In the Che-Check application, whenever a new user logs in, a list of permissions needs to be specified by the user in an authentication

dialog (shown in Figure 4). After the first time authentication process, the Che-Check application will be able to access the user's data using query through Facebook API. The permissions needed and used by Che-Check are listed in Table 3.

Figure 4. Authentication requested at the first time login.

Table 3. Permissions required and used in the Che-Check application.

Name	Description
user_checkins	Permission to get user's check-in history
friends_checkins	Permission to get friend's check-in history
user_location	Permission to get user's current location
publish_checkins	Permission to publish check-in on Facebook
photo_upload	Permission to upload photo on Facebook

4.3 Sending FQL query from mobile application

To send a query from Che-Check to Facebook API, the query messages need to be placed in the code in advance. When the related application function is triggered, the query message will be placed in the URL format and sent to the Facebook API as an URL call. The format of FQL query is similar to SQL language, so that the SQL style interface can be used to query Facebook API. For instance, one of the functions in Che-Check is to query the user's check-in history, including the check-in place and check-in date. The specific queries look like:

```
Query1= SELECT name FROM page WHERE page_id IN
                    (SELECT page_id FROM checkin WHERE author_uid= me())
Query2=SELECT timestamp FROM checkin WHERE author_uid= me()
```

However, in some cases, sending a single query at a time is not enough, the application has to send out a series of queries in the multi-query format in order to get more precise or complicated result. For instance, one of the functions in Che-Check is to query the friend's check-in history, including the check-in place, check-in date, and the person who published the check-in information. The specific queries look like:

```
Query1= SELECT author_uid, page_id, timestamp FROM checkin
            WHERE author_uid IN (SELECT uid2 FROM friend WHERE uid1 = me())
Query2=   SELECT name, uid, pic_square FROM user
            WHERE uid IN (SELECT author_uid FROM #query1)
Query3= SELECT name FROM page WHERE page_id IN (SELECT page_id FROM #query1)"
```

Three of these queries were casted to a JSON object and sent to the Facebook API. Query 1 is processed at first, and then the result of query 1 is used in Query 2 and Query 3. This multi-query feature enables clients to make a series of queries in a single call.

After these queries were sent to Facebook API, a corresponding response will be sent back as a JSON object. The JSON object looks like:


```
[{"uid":605698608,"current_location":{"id":110922325599480,"country":"Taiwan","city":"Taichung"},
"name":"Aaron Lee"}]
```


4.4 Storing extra information in developer's database

As mentioned in previous section, with the SOC concept, most services of Che-Check come from the Facebook API. The only reason to build database is to store the extra information about Che-Check users. Therefore, a simple database is built using MySQL for storing the user's information such as date register, credit earned, and user preference. In addition, PHP is used to connect the smartphone applications with My SQL database.

V. APPLICATION OF CHE-CHECK

The main applications of Che-Check are described in this section. Che-Check is an informative Check-in application. By fetching a variety of up-to-date data from Facebook API, such as users' check-in, friends' check-in, and location-based information, Che-Check is able to collect a large amount of data valuable to users. Moreover, combining the idea of LBS, Che-Check is designed to organize these data to meet users' need based on users' current locations. On the other hand, a website is built to recruit vendors who are interested in providing check-in discount. Vendors can join this website and place the latest promotions or discount for free. Users can not only can get useful information, but also view the check-in discount provided by vendors geologically nearby. Some valuable functions are listed below:

- User's check-in history, which shows the user's check-in history in a list, including check-in place and check-in date.
- Friends check-in history, which shows friend's check-in history in a list, including friend's picture, check-in place, and check-in date.
- The hottest spot nearby, which shows the check-in places nearby, sorting by the check-in counts of those places (Figure 5).
- Friend's check-in spots (on map), that show friends' check-in locations on the map according to the coordinate information of their check-ins (Figure 6).
- Latest check-in discount, which shows the latest discount information provided by vendors nearby.
- Message left for friends, for leaving messages on friends' Facebook walls or recommending something friends might be interested in (Figure 7).
- Check in function, for publishing check-in information via Che-Check and displaying information on the user's Facebook wall (Figure 8).

In addition, users can enjoy these functions in a game-like environment. For example, whenever a user publishes check-in via Che-Check, the user earns 10 credits in Chec-Check. Users can use those credits to open some bonus functions or to get a special title on his/her profile.

Figure 5. The hottest spot nearby.

Figure 6. Friend's Check-ins on the map.

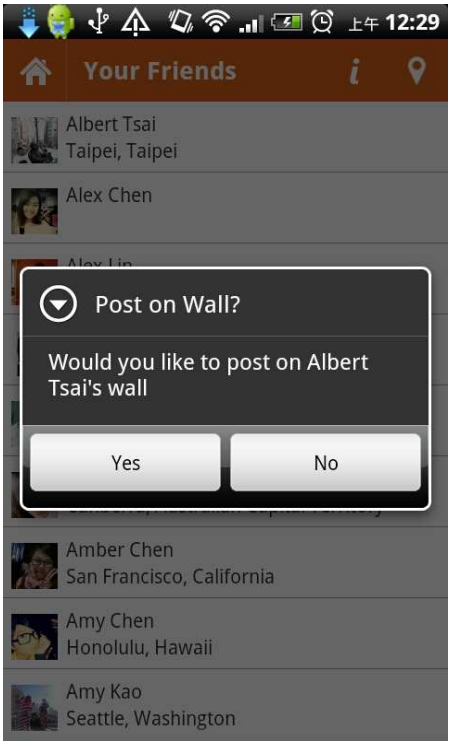

Figure 7. Leave messages to Friends.

Figure 8. Publish Check-ins.

VI. DISCUSSION

With the rapid growth of social network service (SNS), the need of enabling SNS services on mobile devices is increasing (Kim and Kim 2011; Wu, Ho, and Chen 2011). The implementation of combining SNS and mobile applications has been emerging in many application domains. Many popular smartphone applications supporting the check-in function also let users register by connecting to Facebook, such as Foursquare (Figure 9) and Jiepang (Figure 10). However, different mobile applications apply social network API differently. Typically, those applications' API connections are built for fetching the basic information which is used in the user registration process. In order to provide location-based service, those applications still need to build their own services and databases due to the fact that LBS applications usually need to handle huge volumes of data efficiently (Pelekis, Frentzos, Giatrakos, and Theodoridis 2008). In contrast, in our proposed SOC based approach, most of the services needed in Che-Check come from Facebook API. Without building local services and huge database, only a table is needed in our local MySQL database.

A comparison among Foursquare, Jiepang, and Che-Check is conducted in our research by observing the data fetched in applications and the permissions requested to users (shown in Table 4). For example, compared with Che-Check, Foursquare and Jiepang do not request for users' permission which allows them to fetch friends' check-in data. As a matter of fact, Che-Check utilizes Facebook API for supporting in more functions than Foursquare and Jiepang (i.e., Che-Check applies Facebook API more efficiently and more widely than others).

Table 4. Comparison among Three Mobile Applications.

Function \ Application	Foursquare	Jiepang	Che-Check
Register with FB account (Fetch user's data)	✓	✓	✓
Publish check-in to FB	✓	✓	✓
Load user's Friend list (Fetch user's friend data)	✓	✓	✓
Publish on FB friend's wall	✓	✓	✓
View friend's Check-in data (Fetch friend's check-in data)	○	○	✓
View the hottest check-in place nearby, LBS (Fetch place data)	○	○	✓

○: Fetch data not from Facebook API **✓: Fetch data from Facebook API**

Figure 9. Login page of Foursquare.

Figure 10. Login page of Jiepang.

VII. CONCLUSION

In this work, a simple and flexible way of building a check-in application is presented. Therefore, individual developers who have limited resources are able to build an informative mobile application without building local services by applying our proposed approach which

benefits from the social network API and SOC. Furthermore, this paper has discovered the potential values of the proposed approach by conducting a comparison with relevant discussions. In addition to fetching only the basic information of users, more data stored in Facebook is fetched and turned into useful information. It is expectable that an increasing number of individual developers will benefit from our research findings and implications by utilizing free social network APIs in building mobile applications as demonstrated in this article.

VIII. LIMITATION AND ONGOING WORK

Although it is convenient to get a variety of information from the huge data pool of Facebook API, not all of the tables and information are available. Whenever a query goes through the Facebook API, the server first checks whether the client user has granted permissions to make related information available to client applications. If not, the query will not be able to get the expected result. Therefore, the actual amount of tables in Facebook API that are available to client applications will depend on the permission agreements between the applications and the user. On the other hand, there are still some limitations on the FQL version of Facebook API. For example, the amount of data that a client is able to fetch at a time is limited. Therefore, some of the functions are limited as well.

In this paper, only the Facbook API was used in the proposed architecture. Nevertheless, in order to enhance the concept of SOC, there are still many other free APIs (such as APIs of Twitter, YouTube, Google Maps, and Flickr) available on the Internet that can be utilized in building service-based mobile applications. There are other potential and valuable applications which can be created by adding more free APIs into our proposed approach and architecture. In particular, people would like to use the applications which mash up several services they are already familiar with, such as Google maps and YouTube. In the future, we would attempt to add more free APIs into our proposed approach, consequently making the proposed service-based mobile application architecture more complete.

ACKNOWLEDGEMENTS

This work was supported by the National Science Council, Taiwan, under contract number NSC-99-2221-E-005-058-MY3.

REFERENCE

1. Android Developer. “Facebook Query Language (FQL),” March 8, 2012 (available online at http://developers.facebook.com/ docs/reference/fql/).
2. Android Market. March 6, 2012 (available online at http://market.android.com).
3. CheckFacebook.com. “Global Audience of Facebook,” March 6, 2012 (available online at http://checkfacebook.com).

4. Wu, C.-J., Ho, J.-M., and Chen, M.-S. "A Scalable Server Architecture for Mobile Presence Services in Social Network Applications," *IEEE Transactions on Mobile Computing*, online version available on Dec. 8, 2011.
5. Adrian Holzer and Jan Ondrus, "Trends in Mobile Application Development," *Proc. Mobile Wireless Middleware, Operating Systems, and Applications Workshops*, Oct., 2009, pp. 55-64.
6. Lewis, K., Kaufman, J., Gonzalez, M., Wimmer, A., and Christakis, N. "Tastes, Ties, and Time: A New Social Network Dataset Using Facebook.com," *Social Networks* (30:4) 2008, pp: 330-342.
7. Paul Kim and Sangwook Kim, "A Model of Close-Relationship among Mobile Users on Mobile Social Network," *2011 Proc. Ninth IEEE International Conference on Dependable, Autonomic and Secure Computing*, Dec., 2011, pp. 1103-1109.
8. Ko, M.N., Cheek, G.P., Shehab, M., and Sandhu, R. "Social-Networks Connect Services," *Computer* (43:8) 2010, pp: 37-43.
9. Nicolas Kourtellis, Joshua Finnis, Paul Anderson, Jeremy Blackburn, Cristian Borcea, and Adriana Iamnitchi, "Prometheus: User-Controlled P2P Social Data Management for Socially-Aware Applications," *Proc. ACM/IFIP/USENIX 11th International Conference on Middleware,* Nov., 2010, pp. 212-231.
10. Mike P. Papazoglou, "Service-Oriented Computing: Concepts, Characteristics and Directions," *Proc. Fourth International Conference on Web Information Systems Engineering*, Dec., 2003, pp. 3-12.
11. Nikos Pelekis, Elias Frentzos, Nikos Giatrakos, and Yannis Theodoridis, "HERMES: Aggregative LBS via a Trajectory DB Engine," *Proc. 2008 ACM SIGMOD International Conference on Management of Data*, Jun., 2008, pp. 1255-1258.
12. Anna-Kaisa Pietilainen, Earl Oliver, Jason LeBrun, George Varghese, and Christophe Diot, "MobiClique: Middleware for Mobile Social Networking," *Proc. 2nd Workshop on Online Social Networks*, Aug., 2009, pp. 49-54.
13. Rabkin, A. "Personal Knowledge Questions for Fallback Authentication: Security Questions in the Era of Facebook," *Proc. 4th Symposium on Usable Privacy and Security*, Jul., 2008, pp. 13-23.
14. Scale, M.S. "Facebook as a Social Search Engine and the Implications for Libraries in the Twenty-First Century," *Library Hi Tech* (26:4) 2008, pp: 540-556.

利用社群網路 API 及服務導向運算概念建置

打卡型智慧型手機應用軟體

李品範
Pin-Fan Lee
國立中興大學科技管理研究所
g100026108@mail.nchu.edu.tw

張樹之
Shuchih Ernest Chang
國立中興大學科技管理研究所
eschang@dragon.nchu.edu.tw

鄭菲菲
Fei-Fei Cheng
國立中興大學科技管理研究所
ffcheng.ec@dragon.nchu.edu.tw

摘要

隨著智慧型手機以及行動裝置的普遍程度提高，許多軟體公司以及電腦遊戲場商紛紛投入開發行動裝置應用軟體的行列。然而現今開發具互動性及即時通訊性之行動應用軟體對於一般開發者具有相當的進入門檻，因此，如何善用網路上免費的 API 資源是開發者需要思考的議題；本研究將以打卡功能之手機應用軟體為例，探討如何利用 Facebook 所提供的 API 資源結合服務導向運算之概念，以降低開發的門檻。另外，本研究將以上述的概念實作開發手機應用軟體 - Che-Check，並藉由討論其與其他知名的打卡型手機應用軟體使用 Facebook API 程度上之不同，以探討本設計架構潛在的應用價值。

關鍵字：智慧型手機、行動裝置、服務導向運算、打卡、社群網路