Generic Contact Book

Wei Li
Helsinki University of Technology
weli@cc.hut.fi

Abstract

Social network websites have become more and more popular in both desktop and mobile devices. However, the existing solutions to the mobile contact book do not have strong support to these sites. The traditional way to contact a person is still to find the corresponding information (user account) and then send the message using individual applications. In this paper we propose a solution for generic contact book which enables a user to share any content on his social network sites and combine his social network sites usage into the contact book more efficiently. The idea is to use a push server with XMPP (Extensible Messaging and Presence Protocol) protocol together with the already existing RESTful APIs. By means of such as solution a user can also get new E-mail and calendar pushed into the device while the traffic load is reduced dramatically.

KEYWORDS: Social network websites, Mobile phone contact book, Push services, XMPP, REST

1 Introduction

Social network sites have been growing really fast in the past five years. Many scholars have proposed the definitions to those sites. Aaccording to [5], a social network site can be defined as "web-based services that allow individuals to (1) construct a public or semi-public profile within a bounded system, (2) articulate a list of other users with whom they share a connection, and (3) view and traverse their list of connections and those made by others within the system."

Normally, a social network site has three main characteristics: Firstly, a certain number of people with the same interest form a virtual community. For example, people who have the same hobbies, religious belief and political opinion often form such kind of communities. Also, the people who are in the same company, university or high school can also form such communities. Secondly, members can share information in these communities. Unlike traditional instant messaging software, which forms a temporary group and after the conversation the group does not exist any more, in social networks sites the groups are permanent, meaning that the users only need to join the communities once and after that they will be active in the community whenever they want. The forms of the information are also various, from a one-sentence comment on a news topic to an article or a video clip they see on Internet. Thirdly, social network sites have already become part of people's life instead of only an extension. People are using them not only in their spare time for pleasure, but also in their work and study.

Currently there are thousands of social network sites, among which are some famous ones such as MySpace, Facebook, Google Group and Twitter. These websites have been quite successful[11]. Take Facebook as an example. According to New York Times, Facebook was valued at $6.5 billion in May 2009, which was more than some traditional media such as the CBS television network. Another example is Twitter, a website founded in 2006. Within four years Twitter gained a visitor value of 7 million by February 2009, according to Nielsen.

The growth in the use of smart phones has also generated interest in the use of social network sites. With more computing abilities and a larger screen than traditional mobile phones, smart phones are becoming more and more suitable for accessing the social network sites. According to the report released by Opera [14], the mobile use of Twitter increased during 2800% in 2009. Facebook is also popular on mobile phone, according to the same research by Opera: this social network site was the most visited social network on the mobile Web. However, the current solutions for phone book (either on desktop or on mobile phone) do not have enough support for social network sites. It is not convenient to use these sites from phone book, instead, users still have to go to the web page if he for example, wants to post on one of his friend's page. This inconvenience is the motivation for this paper: we wanted to provide a easier way to share information on social network sites.

The rest of this paper is organized as follows: in section 2 we illustrate the current solutions to contact book, from traditional contact book software which have already been used among millions of mobile phones, to IMS/RCS system which is still under construction. Some solutions with Web2.0 technologies are also introduced. And we also show the problems existing in those solutions. In section 3 we propose my solution to contact book using publish/subscribe mechanism. In subsection 3.1 we explain two current protocols for web services, namely, the RESTful protocol and XMPP. A comparison is made to show which one is suitable in our problem. Subsection 3.2 is about synchronization, also using the pub/sub mechanism to decrease the traffic. Subsection 3.3 is the architecture of my solution, with every existing node such as user terminal, social network site servers and a new push server. Subsection 3.4 is a comparison betweeo our propose and current solutions. In subsection 3.5 we discuss the possibility of using XMPP exclusively. In section 4 we also introduce this solution to Digital TV and other home media centers, with the integration of WLan and Internet connection to Digital TV. And finally in section 5 we draw some conclusions on this paper.

2 Current Solutions for Contact Book

2.1 Traditional mobile contact book

The current contact book is rich in contents[4]. Facing such a popularity of social network sites, it is becoming a trend to integrate the support of these sites into mobile phones so that users can communicate with their contacts via those sites any time they want[15]. Many mobile phone companies have already published their solutions to integrate social network sites[3, 8, 12]. However, problems exist when browsing a web page. The similar problem also exists in the scenario of desktop PC usage. Hence, it requires more considerations ranging from choosing protocols to transfer messages to re-design the current contact book software.

Figure 1 is a typical graphical user interface of a mobile phone.

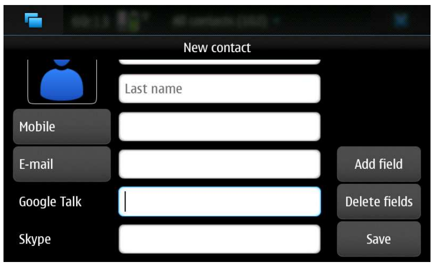

Figure 1: A typical mobile phone contact book

From above we can see that a modern contact book of mobile phone often contains many kinds of entries including mobile phone number, work phone number, E-mail address, and sometimes even some IM numbers such as Skype and GTalk. However, traditional use of such contact information is still to use them via different interfaces. For example, imagine when a user sees a web-page and he wants to share the link with his friend Tom using SMS. The first thing he does is to copy the link address, then select Tom from the contact book and choose to send him a SMS. After that, if he wants to send the same link to another friend, Bob, using E-mail, he needs to choose Bob from contact book and open an E-mail client(assuming there is a E-mail client installed on his phone, otherwise he has to use E-mail services from web pages), write the link and send the mail. Things are more complicated if he wants to share with social network sites' contacts. He has to open a web-page or dedicated application software of the site he wants to use, say Facebook, and write down the link to his friend's page. We can see the problem here: there is not a unified interface to all of these communication media. The contact book only provides the contacts information, not a gateway to those media. The worst case is user has to start a web-page for each social network site at which he wants to leave a message.

Another problem is the synchronization between mobile phones and PCs. Many solutions have already been proposed using for example PC software and USB cable or GPRS, but these solutions also lack the support to social network sites. And many of them still depend on a polling mechanism which requires constant connection establishments and hence is not efficient. For example, Nokia phones provide many synchronization solutions: with Nokia PC Suite a user can synchronize between phones and the PC. However, users have to connect them manually and if the PC is not available, it is not possible to do it. A similar function is also available between phones, but again, it is only limited among specific target phones. Another option is to use the synchronization server provided by Nokia, or even from other companies, for example, Google. By defining the synchronization server and period length, the phone communicates with the server automatically to synchronize contacts, E-mails and calendars. But as mentioned before, it lacks the support to social network sites, and such mechanism depends on sending polling traffic all the time. Recently there are some push notification services using pub/sub mechanism, provided by Nokia and Apple, it is also the method we propose in this paper.

2.2 RCS

The Rich Communication Suite (RCS) is a straightforward solution to such requirement of making one unified interface of different services. It is built on the ground of IP Multimedia Subsystem (IMS), which was proposed as an architectural framework for providing multimedia services[10]. IMS was at first drafted by 3GPP to enhance the mobile networks beyond GSM but in a later version 3GPP stated that supports of other networks than GPRS, such as Wireless LAN and fixed line, are also required.

RCS is proposed by GSM Association (GSMA). The first specification of RCS was finalized in December 2008, with only the support of mobile services. And just like IMS, it also aims to provide the similar functionalities to other networks in its later specification.

The first RCS specification defines one important feature suitable for our problem: the Enhanced Phonebook[9]. The Enhanced Phonebook provides the most functionality we want in this paper. This feature integrates the presence and capabilities information. For example, users can make the chosen contacts see their availability (Online, Offline, Busy, etc.) and published profile. And users will get the update information from their contacts. It is also possible to block this information from chosen contacts. By creating contacts with their social network sites information, it is possible to choose the media to share information.

A crucial problem of RCS and IMS, however, is that the system is so complicated and hence the cost is high. Figure 2 shows a simplified RCS architecture [2]. From this figure we can see that it requires so many hardware and software components to work together to provide the expected services. Operators have to purchase a whole IMS solution to replace some existing parts in their networks. With the fact that it is possible to provide similar services using soft switches and SIP (Session Initiation Protocol) technology combined with existing access networks, IMS has not gained many contracts worldwide[13]. And with the emerging Web 2.0 technology, people are starting to doubt if IMS is actually going to be deployed even in the future.

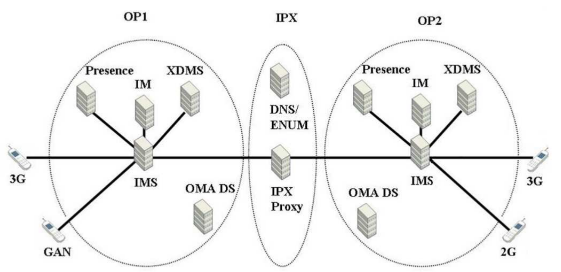

Figure 2: Simplified RCS Architecture
[2]

2.3 Other Solutions

There are some other solutions which can make the information sharing easier provided by mobile phone producers. For example, Motorola has published their application named Motoblur which combines many social network site portals on Android mobile operating system. Users need to create a Motoblur account when first use it into which the social network sites information is imported. Such information is stored on Motorola's server for further use and user can use the same account in any Motoblur supporting phone.

Nokia has also published their solution: on the new N900 phone, which is using the Maemo/Linux operating system, there is a combination of various IM tools for example GTalk, Facebook chat. So they will have a unified interface and use can chat with their friends from different IM account. The only problem is it only has IM support. Users can not for example comment on friend's wall or share videos.

3 Generic Contact Book

3.1 Application Protocol

Many social network sites have published their APIs which enable users to access services without going to their main pages. Hence, it is possible to design a generic phone book which has a unified interface and uses these APIs to communication with contacts on those social network sites. Most of these APIs are based on REST (Representational state transfer).

REST is defined by Roy Fielding in 2000 in his doctoral dissertation [6]. It is a software architecture rather than a protocol. Being a client-server architecture, a RESTful system deployment always involves the request from client and response from server like others, only in REST it is the representation of resources that is being transferred. A RESTful system explores maximally of pre-defined protocols and builds functions upon those protocols. For example, with the existing authenticating and caching mechanisms of application layer protocol chosen to use, designers would not have to worry about these things but should focus on other requirements. Hypertext Transfer Protocol (HTTP) is probably the most commonly used protocol for web-based REST systems. The methods used are as simple as GET, PUT, POST and DELETE, each of them is responsible for manipulating specific resources in client or server.

Facebook's RESTful APIs provide the flexibility of using almost any programming language which can manipulate HTTP messages. Those APIs have included the most functionality such as login/logout and publishing message and photos. RESTful architecture is so popular that many Web 2.0 sites have chosen it to build their APIs. Twitter and MySpace are also using it. Hence it would be convenient to build our unified application upon it.

However, there are still some problems for RESTful systems. The most important one is that RESTful solution is much dependent on a polling mechanism. If a client wants to keep itself informed of the information updates it is interested, for example, a contact's availability or profile, it must keep sending requests to the server every for example 30 seconds, although most of them will end up of getting no update responses. And it is only one piece of resources the client is interested in. We can imagine there are many of them for one contact, many contact for one social network site and many sites. The traffic amount of the mobile phone will be really huge. It is a crucial drawback in our problem since the typical clients are installed on hand-hold terminals which have limitation of battery time. With such high number attempts to connect to the server, the battery time is going to be reduced dramatically.

Some other protocols have been proposed to deal with such problem. Extensible Messaging and Presence Protocol (XMPP, formerly named Jabber) is a XML-based protocol developed by Jabber community. Its original target is to provide a near-real-time and extensible instant messaging (IM) protocol. And in fact it has become a popular protocol in IM area with the applications such as GTalk from Google. But the usage of XMPP is not restricted to IM. It is possible for the designer to define original extension to XMPP to make other functionalities than IM. Our solution is based on such extension.

The XMPP is working like this: a client makes a subscribe request to the server or through the server to another client, wanting to get the update information. So whenever the peer client makes an information update, it publishes it to the server and the server notifies every other clients that have subscribed this information. Such mechanism is called publish/subscribe (pub/sub). It is obvious that compared with RESTful systems, the ones using XMPP have much less number of attempts to connect to servers. Of courses, XMPP also has its drawbacks. Since it has more structured features, the redundancy in the payload is high and hence it is a heavy weighted protocol. But in our deployment scenario, it is still more desirable to use XMPP than use RESTful solutions. Of course, since some social network sites only have their APIs using REST, we need to have some transformation between these two architectures.

3.2 Synchronization

As mentioned before, the synchronization is another important issue in our case since it requires the consistency of contact book between mobile phone and desktop PC. Some current synchronization solutions have already been being used in recent years and they can be divided into two categories. The solutions belong to the first category will connect to the synchronization server at an user-defined interval. Any changes made between two intervals will not get updated until next synchronization connection. For example, Google published its synchronization, Google Sync, to synchronize E-mails, contacts and calendars. It uses a proprietary protocol called Exchange ActiveSync (EAS) developed by Microsoft. An EAS enabled terminal will try to connect to a server provided by device vendors or Internet Service Provider, send its changes since last synchronization and get changes from the server. It is noted that it is also possible for operators to use some already existing EAS server instead of building their own. For example, Nokia's mobile users can also use Google Sync servers to synchronize their contents since there is an EAS client called Mail for Exchange installed in the phones. The mail problem of such solution is the date is not really synchronous. The data change between two synchronization intervals might be important and users will not get them right after they are sent. Another example of such solution is RSS, which is also based on pull technology.

The second category is based on Pub/Sub mechanism, and my solution also belongs to it. For example, Nokia published its mail synchronization technology called Nokia Messaging. The E-mail transmission is using Internet Message Access Protocol (IMAP), and there is a state called IMAP IDLE to make the push mail possible. When a connection to the E-mail server is established but no new mails available, the connection is set at IMAP IDLE state, waiting for any new information updated. Whenever there is any new mail, the state is changed to normal IMAP state and the server pushes the mail to client.

However, my solution is not using IMAP idle, instead, it uses XMPP as well. The main difference between those two solutions is, it does not require a constant connection between the client and the server for XMPP. It might not be feasible to transfer E-mail using XMPP since there will be some specific contents such as large size file and images which are not convenient to be transferred with XMPP. However, since our application only requires data update of contacts information, it will not be a problem with XMPP. Another advantage of such solution is it can use the same server as the messaging traffic. So it is worth mentioning that in next subsection of architecture, although there are two servers for synchronization and contents pushing listed on the figure, it only means they have different functions, and in actual implementation they can be combined as one server. There are already some solutions using XMPP. It is more preferred in the sense that a client does not need to keep trying connecting to the synchronization server to check if there is any change. Instead, whenever a client makes a change to the contact book, it sends a notification to the synchronization server which publishes this change to the clients who sharing the same contact book. For example, the Push Notification and MobileMe services provided by Apple are using XMPP.

3.3 Architecture

Figure 3 is the architecture of my solution. There are four key components: terminal application software, synchronization server (SS), push server (PS) between mobile user terminals (UT) and social network sites servers (SNSS).

Terminal application software could consist of two components: a plug-in for web browser and an individual application software. The first one works like other plug-in: when a user finds a page or a video clip when browsing, he could right-click the mouse and a context menu pops up with an option named "Share". Going into this menu it will show all contacts. The user can then choose the person he wants to share the link with and choose a media he wants to use. If he chooses to use SMS or E-mail, it will use the already-existing solutions to send it. If he wants to send the message to contact's Facebook or Twitter, the message will be sent to the PS which will then send it to Facebook or Twitter's server. The individual application works like IM software, but it is able to receive and show message from all kinds of media: from SMS to Facebook and Twitter's chat.

As described above, the SS is utilizing XMPP to make the contact book consistent among UTs such as mobile phone, PC, net notebook and digital TV. The PS is communicating with SNSS using polling mechanism all the time via SNSS's RESTful APIs. Whenever any information which the user has subscribed is updated, the IS sends notification to mobile UT using XMPP. One thing worth mentioning is the PS can also push other contents into user's terminal: E-mail, calendar, RSS. For example, it can also keep polling user's E-mail server, and whenever there is any new E-mail, it will push it to user's terminal. In such scenario, the traffic load between PS and mobile UT is reduced dramatically and hence the battery time is prolonged. For PCs, however, there might not be the problem of energy supply, and user might not care if the traffic load is high (provided they have unlimited Internet access). So if operator wants to reduce the burden of PS, it could let PC communicate with SNSS directly. Of course, communication via PS is also possible. The operator can encourage its desired method by providing differentiated charges.

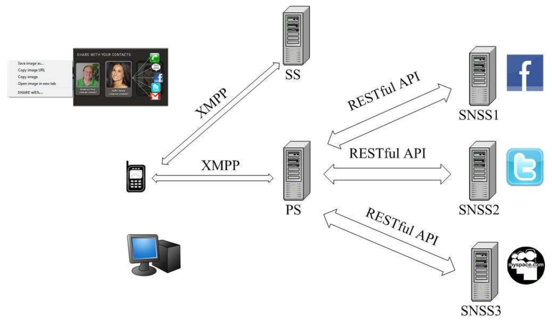

Figure 3: Architecture of Generic Contact Book

3.4 Comparison to Current Solutions

As discussed before, the most conspicuous advantage of the solution in this paper is that it will save energy from keeping polling servers. The energy has become one crucial restriction to mobile terminal. In addition, current mobile broadband charging model depends on actual traffic used. By reducing the polling connections, the phone will have longer battery time and reduced Internet traffic load.

Another advantage compared to current solution, especially to RCS solution, is that it does not require big change to the current architecture. As mentioned above, the most drawback of IMS/RCS is that it needs to add many components, both hardware and software, to current core network. However, it is becoming an acknowledge that many of the functions of IMS can be provided by some Web2.0 technologies.

However, my solution also has some drawbacks. For example, an intermediary server (for synchronization and actual contents push) must be added, whereas, solutions based on polling and RESTful mechanism do not need it. But we believe that if operators or ISP provide such servers to save energy and reduce Internet traffic for users, they will attract more consumers and hence the expense on such servers will be paid off.

3.5 Full XMPP Possibility

From the description we can see that XMPP has advantages of reduced traffic and hence making more scalability. It is a valid question that if a full XMPP architecture can be achieved. One possibility is to use the IM semantic of XMPP to express more behaviors of social network sites such as login. For example, there is a XMPP gateway for Twitter called tweet.IM which in charge of transferring chatting messages of Twitter RESTful format to XMPP format. If such gateway is installed on the terminal, the intermediate server will not be necessary. However, it will require more work since the IM semantic of XMPP is too limited to express all the behaviors. For example, it is hard to find equivalent IM actions for some Twitter behaviors such as reply and favorite. In another word, it requires to build the whole new semantic on top of XMPP as expressive as RESTful methods, instead of reusing some existing and relatively simple IM semantic.

Of course we can expect that social network sites provide their APIs using XMPP instead of letting us build the whole semantic transfer work. However, since the RESTful APIs have simplicity and light weighted features, they are still going to be used without switching to XMPP until it is beyond the servers' processing ability.

4 Future Work of Extension on TV

It is also possible extend such a generic contact book to Digital TV and other home media consoles. Although the traditional digital TV technique is to use dedicated cables and protocols to transfer contents [7, 1], there is a trend to integrate Internet and WLan connection into digital TV and other consoles (for example, on the video consoles XBOX360, Play Station 3 and the Sony Bravia TV). And such digital TV will also use many existing Internet protocols to view the contents on Internet. With such integration, it will be easier than ever to use our solution based on HTTP protocol.

Take XBOX360 as an example. It has a network interface which can be used to connect to the Internet. Besides playing online games, it is also possible to use it to join a social community called XBox LIVE to communicate with other game players (in Play Station 3 it is even possible to browse any web page). And now Microsoft has integrated Facebook and Twitter into XBox LIVE so people can chat with their friends via those web sites, browse friends' pages, comment on the wall and check videos from there.

5 Conclusion

Social network sites have already been playing a key role in people's daily life. It is a common requirement to integrate the support of those sites into current contact book on both mobile phones and PCs. However, current usage is still to go to the webpage of those sites, either using PC or mobile phone, and to share information on them. We proposed a solution to integrate the social network sites into the phone book. A user can easily communicate with his friends on those sites by choosing relevant function.

The APIs published by social network sites make our idea possible. By deploying them, users do not need to go to the webpages to use their functions. However, since those APIs are mainly using REST architecture and the existing RESTful solutions have the disadvantage of heavy pulling traffic load and hence reduced terminal battery time, they are not suitable for mobile devices. XMPP is more suitable for such a scenario. But since some social network sites only provide RESTful APIs, a push server is needed to communicate with social network site servers using REST and only send update notification to user terminals using XMPP. The push server can also be in charge of pushing other contents such as E-mail and calendar. A unified architecture for both mobile phones and PCs has been proposed in this paper. It can provide multiple methods for a user to share the link he is browsing.

References

[1] L. Ardissono, A. Kobsa, and M. Maybury. *Personalized digital television: targeting programs to individual viewers*. Kluwer Academic Pub, 2004.

[2] G. Association. RCS Release 1 Technical Realization v1.2, February 2010.

[3] A. Bhamidipaty and P. Deepak. Symab: Symbol-based address book for the semi-literate mobile user. *Lecture Notes in Computer Science*, 4662:389, 2007.

[4] J. Blom, J. Chipchase, and J. Lehikoinen. Contextual and cultural challenges for user mobility research. 2005.

[5] D. Boyd and N. Ellison. Social network sites: Definition, history, and scholarship. *Journal of computer mediated communication-electronic edition*, 13(1):210, 2007.

[6] R. Fielding. *Architectural styles and the design of network-based software architectures*. PhD thesis, Citeseer, 2000.

[7] B. Fox. Digital TV comes down to earth. *IEEE Spectrum*, 35(10):23–29, 1998.

[8] S. Gaur. Mobile phone contact book, the social approach, 2008.

[9] K. Henry, Q. Liu, and S. Pasquereau. Rich communication suite. In *Intelligence in Next Generation Networks, 2009. ICIN 2009. 13th International Conference on*, pages 1–6, Oct. 2009.

[10] M. Jain and M. Prokopi. The IMS 2.0 Service Architecture. In *Next Generation Mobile Applications, Services and Technologies, 2008. NGMAST '08. The Second International Conference on*, pages 3–9, Sept. 2008.

[11] A. Lenhart and M. Madden. Social networking websites and teens: An overview. *Pew Internet & American Life Project*, 3, 2007.

[12] G. Lugano. Mobile social software: definition, scope and applications. In *Proceedings of eChallenges*, pages 1434–1441, 2007.

[13] T. Magedanz. IMS vs. P2P and Web 2.0 - Understanding the Role of the IP Multimedia System (IMS) in Face of a Converging Telco and Internet Service World. In *Applications and Services in Wireless Networks, 2008. ASWN '08. Eighth International Workshop on*, pages 3–3, 2008.

[14] Opera Software . http://www.opera.com/smw/2009/11/, November 2009.

[15] A. Oulasvirta, M. Raento, and S. Tiitta. ContextContacts: Re-designing SmartPhone's contact book to support mobile awareness and collaboration. In *Proceedings of the 7th international conference on Human computer interaction with mobile devices & services*, page 174. ACM, 2005.