An interface for context-aware retrieval of mobile contacts

Vassilios Stefanis
University of Patras / CTI
Rion, 26504
Patras
Greece
stefanis@ceid.upatras.gr

Athanasios Plessas
University of Patras / CTI
Rion, 26504
Patras
Greece
plessas@ceid.upatras.gr

Andreas Komninos
Glasgow Caledonian University
70 Cowcaddens Rd.
Glasgow G4 0BA
UK
andreas.komninos@gcu.ac.uk

John Garofalakis
University of Patras / CTI
Rion, 26504
Patras
Greece
garofala@ceid.upatras.gr

MobileHCI 2013, Aug 27 – 30, 2013, Munich, Germany.
ACM 978-1-4503-2273-7/13/08.
http://dx.doi.org/10.1145/2493190.2494432

Abstract
Our work discusses a mobile contact retrieval interface which attempts to contextually predict the contacts a user is likely to need access to, in order to facilitate the retrieval process. We compare our prototype implementation with retrieval from traditional applications (contact list and call log) in a preliminary lab experiment and discuss our findings from user behaviour. We conclude with suggestions on how to improve this interface in order to further enhance the retrieval process.

Author Keywords
Mobile contact lists; Context-Awareness; Mobile information retrieval

ACM Classification Keywords
H.5.m. Information interfaces and presentation (e.g., HCI): Miscellaneous.

Introduction
Though retrieving contacts for the purposes of communications is possibly one of the most frequent activities when using a mobile device, surprisingly little research has been carried out on how to improve the user experience by introducing context awareness to such interfaces. In previous work [1] we have shown on a theoretical basis through simulated retrieval tasks,

how mobile users can benefit from mobile social context, especially in the case of mobile PIM (Personal Information Management) items retrieval, such as contacts, multimedia content, emails etc.

As a first step in our research, we have focused on mobile contact lists, which as they are getting bigger [2], among others to incorporate contacts from several sources (e.g. phone, email, Skype, Facebook etc.), pose a significant cognitive load to the mobile user for the task of contact retrieval. Based on the analysis of mobile call log data presented in [3], we identified several context dimensions that play an important role in mobile contacts' communication usage and could be used in a vector context model that could be adopted for predicting the next contact to be called.

A context aware algorithm for predicting outgoing calls

Based on these findings, we proposed a scalable algorithm [4] for predicting the most probable contacts to be called at any time. Using only two context dimensions (frequency and recency), our algorithm was tested against a real mobile dataset [5] with very promising results. Our algorithm considers the context dimensions and provides a list of contacts that replaces the traditional lists available in modern mobile phones (alphabetic list of contacts, list of most frequently used contacts, list of most recently used contacts). The algorithm in [4] works as follows: whenever the user wants to retrieve a contact in order to start a new communication session, a recent portion (called training window – e.g. 15 days long) of the call log (both ingoing and outgoing calls) is analyzed. A score F(c) is assigned to each contact according to the communication frequency with it within the training window. Moreover, a score R(c) within a recency window (e.g. last 6 hours) is also assigned to each contact representing how recent was the last communication with this contact. Finally, the total score for each contact is computed as $T(c) = w_f*F(c) + w_r*R(c)$, where w_f and w_r are the weights of each dimension (e.g. $w_f=0.5$ and $w_r=0.5$ for equal weights). This score is used to sort all contacts in descending order and select the top n to create the proposed contact list.

The Calchas application

We developed a Google Android application, called Calchas (*kal'-huhs*\ – a seer in Greek mythology who predicted the fall of Troy), in order to perform some preliminary lab experiments of our algorithm with real users, instead of simulated runs.

User interface

The interface of the application is simple and adopts a visual style familiar to smart phone users and especially Google Android users. The main screen is predominantly occupied by a suggestion list of the contacts (or numbers) that are deemed more likely to be called (Figure 1). For each contact the application shows the contact's name (if present in the user's contact list) and the contact's phone number. If a contact has more than one phone numbers saved (e.g. home, mobile, work etc) the application presents the contact's last used number. Also, the application presents the date and time of the last communication with the proposed contact. Furthermore, for each suggestion, an Android quick contact badge is available. The quick contact badge is an Android system widget that launches a toolbar with several different actions, such as presenting all the phone

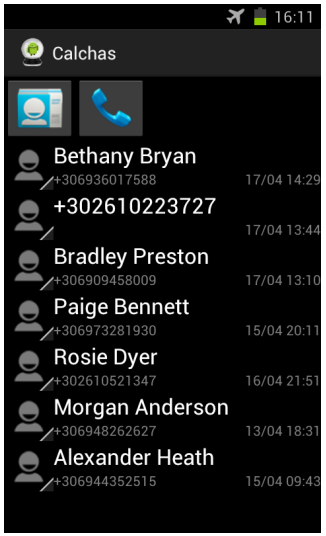

Figure 1. Calchas main screen

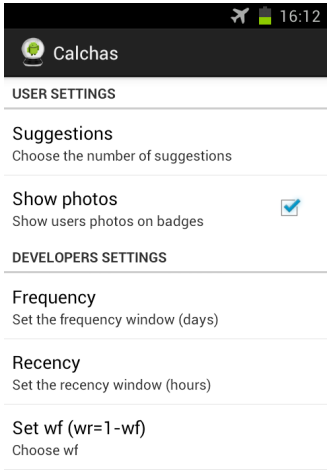

Figure 2. Calchas settings screen

numbers of the contact, making a call, writing a new SMS, emailing the contact if an email address is available etc. Finally, if the suggestion is a number that is not assigned to a contact, by clicking the quick contact badge the user can save this number as a new contact or update an existing contact with it. A further section in the main interface assists the user in the case that the desired contact is not in the suggestion list. In this instance, the user is given two options via an icon that launches the native contacts application and a further icon to access the device's call log.

The application is configurable through a settings menu (Figure 2). Therein, the user can choose the number of suggestions that will appear in the main screen (5, 6, 7 or 8 suggestions) and enable an option for contact photos to replace the default quick contact badge widget icon (if available). Finally, the user can set the frequency window (in days), the recency window (in hours) and the weights for the recency and frequency dimensions. These are adjusted manually for experimentation, though we are aiming for an automated adaptation of these parameters based on profiling the user by monitoring their behaviour.

Application architecture

The Calchas application consists of three modules. The first is the UI module responsible for creating the interface described in the previous section. The UI of the application is refreshed with a new prediction set which is calculated when the application enters the Running state of the Android activity lifecycle. This assures that the user is provided with up to date proposals, every time that the application's main screen becomes active. The second module is the coded implementation of our algorithm, as described in the previous sections. The third module is the usage statistics module. We save anonymous statistics in an SQLite database in the user's device and we also provide the option for sending those statistics to a remote server via HTTP protocol. For these logging purposes, we consider each refresh of the application's user interface as a new usage session. For each session we record the time started (as a UNIX timestamp) and the events that occurred. We define the following events: a) The user selected one of the suggestions b) the user launched the contact list c) the user launched the call log. When an event has occurred or the user just closed the application (e.g. she wanted just to see the proposals) we consider that the session is over and we record the end time. Moreover, for the event (a) we record the full list of suggestions (contact id or the hashed number if the suggestion is not a contact), the actual order of the suggestions, the chosen suggestion and the frequency and recency scores of the chosen suggestion.

Description of the experiment

The purpose of our experiment was to obtain preliminary results on the effect of Calchas application on the retrieval process (speed, path) of a contact to be called. For this purpose we designed and conducted a lab experiment that involved several retrieval tasks representing different retrieval scenarios.

In [6] a retrieval experiment was designed using the participants own contact lists and call logs. However, in this case we wanted to set all participants on the same baseline and to exclude any potential learning or memory effects that might influence their performance. For this reason and because we only had a few participants, all experiments sessions were carried out

on the same device (Samsung Galaxy S2, Android 4) using the same call log and contact list. The call log and contact list were taken from an actual participant in a previous experiment [3] who was close to the average characteristics of user behavior (size of contact list, density of communications) [3]. Specifically, this call log had 500 entries (Android's maximum call log size) representing over 55 days. The contact list had 172 contacts and we replaced every name with a unique randomly generated English name and surname.

The main task for each participant was to find and call a contact using three different applications: the build in contact list, the call log and finally with Calchas. The user's settings in Calchas were 7 suggestions, 10 days for the frequency window and 6 hours for the recency window. Also, w_f and w_r were equally weighted at 0.5. Each participant had to retrieve six contacts, each one with all three means. The first two contacts were a worst-case for Calchas (not presented in the suggestion list), the next two were an average case (presented but low in the suggestion list) and the final two a best case (presented and high in the suggestion list). All participants used the applications with the same order as we wanted to see how previous experience in the retrieval process affected the retrieval path within Calchas.

We gathered 20 participants (9 female) from 18 to 45 years old (m=30.1, stdev=7.5) with different educational backgrounds, 12 of them Android users. We asked each participant to perform the six retrieval tasks by presenting a card with the contact name to be found one at a time. We measured the time from the launch of each application until the participant initiated a call to that contact. We should mention that Calchas averages less than 0.3 seconds to perform the predictions and launch on the device used. We also set the device to flight mode so that call log should not be affected during the experiments. Finally, before the start of the experiment with each participant we made a short demonstration of the Calchas's functionality and set the device's time and date to a preset value.

Results

An ANOVA of results (Figure 3) shows that the retrieval times are not statistically significantly different in the average and best case scenarios for Calchas (*p=0.044)*. This means that the presentation order for suggestions does not seem to play an important role. In our application we present the suggestions ordered by prediction score, but an interface could equally well present suggestions in alphabetical order. This verifies that alphabetically ordered interfaces such as the one presented in [6] may be optimal as they can be mixed within existing interfaces familiar to users. We also note that in the worst-case scenarios, the application poses a penalty on the retrieval process as the users first examine the list of suggestions only to find that the desired contact is not provided, however, this penalty averages approximately 2-3 seconds, which we do not consider to be prohibitive.

In the worst-case scenarios, the user has to choose between proceeding with the call log or the contact list. In this case there are two distinct situations. For contact "Elizabeth", 7 users chose the call log as a second step and 13 chose the contact list. Those users had a better performance using the call log in the preceding retrieval tasks, while those who chose the contact list, took on average the same time on both preceding tasks. Hence we note that in their case, the

choice was to "play it safe" and go for the contact list, where finding the contact is a guaranteed result. For contact "Isabella", users are precisely split for the second step. Those that chose to use the call log, surprisingly did better with the contact list in the preceding tasks. While this finding was unexpected, we noted that two of these users had an exceedingly long retrieval time using the call log, which improved dramatically when they used the call log again as a second step. Taking these two users out of the picture, the results fall back to expected behaviors as the remaining 8 users fared better with the call log than the contact list in preceding tasks. Moving on to the 10 users that chose the contact list as a second step, we note that for them the average time using the call log was significantly longer than using the contact list, hence their choice is justified.

It appears that the users' previous experience of using each tool plays a role in deciding which retrieval path to follow when Calchas does not show the desired contact in the suggestion list. By extension, it can be assumed that the users' predicted experience, i.e. how likely they believe it will be to find a contact using one or the other tool, plays a role in selecting a second step. Here, there is a trade-off between using the contact list and the call log. Using the contact list always takes about the same time (unless of course the desired contact is on the first displayed screen) but retrieval success is more or less guaranteed. The call log is advantageous

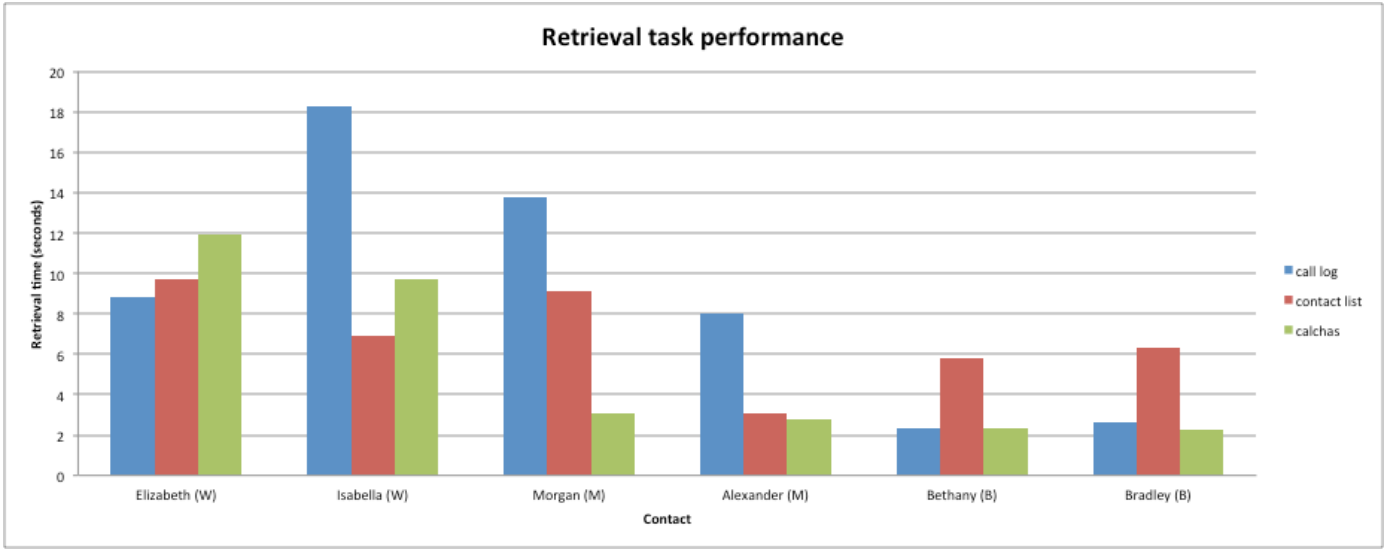

Figure 3. Tasks for the worst (W), average (M) and best (B) case scenarios

as interaction consists solely of scrolling, instead of typing or hitting index targets, where according to Fitt's law the required interaction time is longer. Further, the call log is advantageous if the desired contact has been communicated with recently or if it is a frequently used contact (hence more chances to spot it while scrolling), though in both these cases, Calchas could have picked up the contact with a variation in the algorithm settings (longer frequency and recency windows). In [4], we found that even for very "social" users who make many calls to many contacts and are hard to predict for, our algorithm yields an 82% hit rate compared to 55% (call log) and 75% (frequently called list). However, for those "miss" cases where a user makes a bad choice in selecting a second step tool due to inability to accurately judge which tool might be best (e.g. elderly people with limited memory abilities or very social people who communicate with many contacts), the interaction penalty can be quite significant. We see two potential improvements that can help channel user behaviour towards minimizing this penalty: First, Calchas only displays a single screen-ful of suggestions. Further suggestions could be loaded by expanding the frequeny and recency windows on demand (e.g. pressing a "load more" button). A second option would be to dynamically adapt the call log that is used as a second step, by showing a version where the suggestions presented in Calchas would be removed, hence leaving a significantly shorter list that contains only the more "rare" contacts. Finally by shortening the contact list to exclude Calchas' suggestions and all entries in the shortened call log, we could provide a third, and final step for retrieval. A longitudinal field study with more participants, this time using their own data, would help us test some of these concepts and solidify our understanding.

References

[1] Komninos, A., Plessas, A., Stefanis, V. and Garofalakis, J. Application of dimensionality reduction techniques for mobile social context. In *Proc. of Ubicomp 2011*, ACM Press (2011), 583-584.

[2] Komninos, A. and Liarokapis, D. The use of mobile contact list applications and a context-oriented framework to support their design. In *Proc. of MobileHCI '09,* ACM Press (2009).

[3] Stefanis, V., Plessas, A., Komninos, A. and Garofalakis, J. Patterns of usage and context in interaction with communication support applications in mobile devices. In *Proc. of MobileHCI '12*, ACM Press (2012), 25-34.

[4] Plessas, A., Stefanis, V., Komninos, A. and Garofalakis, J. Using communication frequency and recency context to facilitate mobile contact list retrieval. *International Journal of Handheld Computing Research (IJHCR)* (to appear).

[5] Laurila, J.K., Gatica-Perez, D., Aad, I., Jan Blom, T.-M.-T. D., Bornet, O., Dousse, O., Eberle, J. and Miettinen, M. The mobile data challenge: Big data for mobile computing research. In *Mobile Data Challenge by NOKIA Workshop, in conjunction with Int. Conf. on Pervasive Computing*, 2012.

[6] Bergman, O., Komninos, A., Liarokapis, D., & Clarke, J. You never call: Demoting unused contacts on mobile phones using DMTR. *Personal and Ubiquitous Computing,* 16(6), 757-766, Springer (2012).