Minimizing Mobile Phone Disruption via Smart Profile Management

Amnon Dekel
The Selim and Rachel Benin School of Computer Science and Engineering,
The Hebrew University of Jerusalem
+972-54-813-8160
amnoid@cs.huji.ac.il

Dan Nacht
The Selim and Rachel Benin School of Computer Science and Engineering,
The Hebrew University of Jerusalem
+972-54-734-3534
dan.nacht@mail.huji.ac.il

Scott Kirkpatrick
The Selim and Rachel Benin School of Computer Science and Engineering,
The Hebrew University of Jerusalem
+972-2-658-5838
kirk@cs.huji.ac.il

ABSTRACT
In this paper, we describe the Smart Profile Management application that was designed to help minimize mobile phone disruptions. The system achieves the goal using a machine learning based algorithm that switches the phone profile automatically. A prototype was developed in Python for S60 and an informal usability test was run for a period of 7 days. Results show that a large proportion of profile suggestion changes were accepted by subjects, suggesting that such an application can in fact lower disruptions.

Categories and Subject Descriptors
[H.5.2 User Interfaces]: Interaction styles, User-centered design

General Terms
Algorithms, Design, Human Factors.

Keywords
Mobile phones, profile management, context based computing, smart interfaces, minimizing disruptions.

1. INTRODUCTION
The mobile phone has become the most ubiquitous user operated electronic device in history. With the growing power of small footprint low power processors the functionality of mobile phones has grown to enable people to carry a phone, a full featured PDA, an audio recorder, a music player, a high resolution photo and video camera, a photo and video viewer, a powerful web browser and a computer gaming console in their pockets, all in one small device. Coupled with online contact and calendar management systems that the phone can interface with, people can have access to and manage most of their lives through the small window of their mobile phone. But this explosion of functionality and fusion of services comes at a price – people are finding themselves struggling with more complicated phone interfaces as well as frequent disruptions from incoming phones calls, SMS, MMS and WAP push messages, RSS feeds, email notifications and calendar alarms.

1.1 Mobile Phones as Disruptive Devices
As stated above, since the phone has become capable of being the hub of a person's informational life, it has also become a device that too many a time disrupts a person in the middle of doing something. In the past this disruption came from incoming phone calls and preset alarms. But with all the information services that the mobile phone is interfaced to today, one can be disrupted more than ten times an hour[1]. If the user finds this useful then so be it - but in many cases people find these disruptions to be annoying, and would be happy to minimize the phone's potential for disruption in any way they can.

1.2 Profile Management on mobile phones
One useful feature that has developed over the years is a phone profile manager. This program allows a user to manage the behavior of the phone in specific situations. Standard profiles include "general", "silent", "meeting", "outdoor" and "pager". In each of these, the user can control how loud the phone will ring (from maximum loudness to silent), how many times it will ring (from none, to endless), the ring tone it will use when ringing and whether the phone will vibrate when ringing, in addition to additional second level characteristics. Some systems even allow a person to set a timer for a profile and have it revert to a previous profile when the time has passed. Although profile managers exist on many phones, people tend to use them infrequently or in a very simple fashion - for example manually switching to meeting or silent profiles. The iPhone, which represents a new generation of user experience and usability in mobile phones, has taken this into consideration and has only two profiles (silent or normal) which is activated by a physical switch. But when manually configuring profiles, an altogether too frequent problem is forgetting to change the profile back to "normal", general" or "outdoor", with the subsequent problem of missing incoming phone calls.

1.3 Automatic Profile Changing
Although we have not found any published research into such systems, there are a few commercial applications (Handy Profiles for S60 [5], Photo Contacts Pro 5 [8]) that can automatically

[1] In our study a phone was synched to Google calendar which sent out SMS reminders. This was in addition to other incoming notifications from friends and email.

switch profiles depending on a calendar entry, the time of day, or even the day itself. Although such programs go a long way to solving the manual profile setting problem, they rely on a simple model that matches time to profiles and cannot deal with special cases or learn from continued use by the user.

We identify three methods to develop such systems: *Simple Middleware* applications that switch profiles automatically (as used in the commercial examples above), *Rule Based Systems* where a system of configurable, flexible pre-defined rules changes the active profile, and *Machine Learning Based Systems* - the most sophisticated solution - where a machine learning algorithm is used to select the active profile. The application reported here implements the latter solution: a machine learning module, which is constantly adapting itself to its specific user needs. This application is an enhancement of a previous version of our Smart Profile Management application, where a flexible rule-based system was used to determine the required profile.

2. THE SMART PROFILE MANAGER

Our smart profile manager application is part of the Mobile-Smarts project at the School of Engineering and Computer Science at the Hebrew University Jerusalem. The Smart Profile Manager is a machine learning based application for Symbian S60 that constantly learns its user profile selection pattern online, and offers to change profiles depending on the various features learnt (as specified in section 2.1.2). The system uses the K nearest neighbor algorithm, and supports all standard profiles ("general", "silent", "meeting", "outdoor" and "pager").

2.1 Technical Description

The program was developed with Python for S60 [7] on a Nokia N95. We decided to use Python to develop the application because it enables real rapid prototyping of mobile applications. Having said that, there are also disadvantages to developing with Python S60, such as a limited set of OS access commands, which forced us to rely on external libraries. An example of such a library that we used is Xprofile, a module that was developed by Cyke64 [1] (a Nokia forum champion), which enables switching the active phone profile. Another serious disadvantage of developing with Python is the lack of event notification (i.e. our software cannot be triggered by OS events when a user adds appointments, changes the active profile, etc).

2.1.1 Previous Work

The previous version of our application implemented a rule-based system which changed the active phone profile according to various parameters as described in Table 1. The system was flexible and adjustable (allowing fine-tuning of the various parameters), however it suffered from the disadvantages mentioned in section 1.3.

Table 1: Summary of Previous System Rule Parameters

Context	Description	Use case example
Time of the Day	Changes related to the time of the day	Reverting from *Silent* to *General* mode if the user appears to have forgotten switching back after N hours
Phone's Database	Events appearing in the phone's calendar, such as appointments	Switching to *Meeting* when a meeting starts, and back to the previously active profile when it ends
Usage History	Following user behavioral patterns	Switching to a certain profile which the user constantly selected over a period of time (e.g. *Outdoor* every Monday at 10:00AM, and Pager everyday between 4:00PM and 4:30 PM)

2.1.2 kNN Machine Learning Algorithm:

The algorithm we use in the current application version for choosing the appropriate active profile is *k-nearest neighbor* (kNN), which is one of the most fundamental and simple classification methods. The k-nearest-neighbor classifier is commonly based on the Euclidean distance between a test sample and the specified training examples. The training examples are vectors in a multidimensional feature space. The training phase of the algorithm consists only of storing the feature vectors and class labels of the training samples. In the actual classification phase, the test sample (whose class is not known) is represented as a vector in the feature space. Distances from the new vector to all stored vectors are computed and *k* closest samples are selected. The actual classification of the object is then determined according to the most common label amongst its *k* nearest neighbors.

2.1.3 Characterizing the Vector

The current version of the application uses the following information for the example vectors:

- Day of week
- Time of day
- Current calendar record
- Active profile

The reason for this limited feature set is our desire to "keep it simple" for the proof of concept; however, the above feature set can later be extended to contain additional information such as geo location, cell information and even accelerometer data about how the phone is being moved (or not moved).

2.1.4 Fuzzy Classification

For analyzing the phone's event database, we converted an objectively measurable parameter into a subjective "category membership", which is used for classification. That is, the software reads the content of the current calendar record (if it exists), and looks for pre-defined words, which help associate that event to a certain category: "jogging", for example, will associate the event for the "Outdoor" category, whereas "lecture" will associate it to the "Silent" category. It is important to stress that these categories are not "true categories" that we wish to classify, rather merely ranges of feature values. The reason we use this kind of classification is the fact that a more accurate NLP (Natural Language Processing) system would have created a larger group of possible categories – which would be better in general, but require a longer time to test, as well as a much bigger group of subjects for our initial experiments. Future implementations should certainly include some sort of NLP module in order to deliver more accurate results.

2.1.5 User Interface:

Before switching profiles, the application asks permission to do so. In order to draw the use's attention, two short vibrations occur

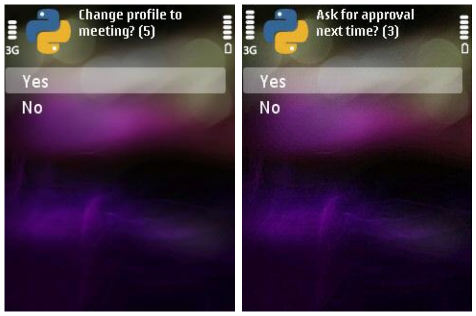
Change profile to meeting? (5)
Yes
No
Ask for approval next time? (3)
Yes
No

Figure 1: Program Dialogs to change profile

along with a screen dialog, which waits for 5 seconds (countdown is displayed). The reason for this is the fact that the user might not be attentive to respond, and therefore the system should not hang. The user's response (accepting, rejecting or even ignoring) is saved to a log file for later analysis. An additional dialog enables the user to release the system from asking for approval in the future (see figure 1) and thus not disrupt the user further.

2.1.6 Data acquisition:

As mentioned above, the program keeps track of the user's behavior via the file system: a log file tracks the user's responses to the profile change suggestions. This log contains the suggestion timestamp, the suggested profile and the user's reaction. This data, along with the user's calendar data, allows us to analyze the algorithm's behavior and how it affected the actual user experience.

3. USABILITY STUDY

We ran an informal usability study in order to gauge initial user impressions and find usability problems. We configured 6 Nokia N95 phones with the application, where the K parameter was set to 3 and the learning set minimal size was set to 1000 vectors, and asked the subjects to continuously use it for a period of 7 days. We also asked the subjects to move all their calendar management to Google calendar [3] which was then synched to the phone calendar using the free Goosync service [4]. We did this in order to simulate the use of office based meeting management services with the mobile phone user. After the test we analyzed the usage logs and ran informal interview with the subjects

3.1 Results

The following table summarizes the results of our usability study. It contains the following columns:

- **Subject** – unique ID for each subject
- **Total** – total number of potential suggestions
- **Right** – correct suggestions (e.g. "change to *Meeting*" when *Meeting* was indeed required)
- **Wrong** – wrong suggestions (e.g. "change to *Silent*" when *Outdoor* was actually required)
- **Missing** – no suggestion occurred, where such was actually needed
- **Not Required** – suggestion occurred where it was actually unnecessary

The red figures show the percentage of correct suggestions.

In general, the application performed as planned and changed profiles according to schedule, later changing them back when the meeting time was done. Looking at the raw application log data across subjects shows that most of them accepted the profile change suggestions by the application: three of the subjects enjoyed a rate higher than 90% of correct suggestions, one of them experienced a rate of 77%, and the other two did not get any correct suggestions (however a deeper analysis of their logs indicated that they had very low numbers of potential profile changes, which was insufficient for such a learning algorithm).

Table 2. Application Suggestion Performance

Subject	Total	Right	Wrong	Missing	Not Required
1	14	14 100%	0	0	0
2	13	10 76.9%	0	3	0
3	2	0 0%	2	0	0
4	11	10 90.9%	1	0	0
5	4	0 0%	0	4	0
6	13	12 92.3%	1	0	0

3.2 Cross Validation

In order to assess how the results of this statistical analysis will generalize to an independent data set, we did a small cross-validation test with subjects 3 and 6 (since they represented a bad result and a good one, respectively). For this test, we pre-populated their training set, each with the combined training set of all other subjects (excluding him/her). We then asked subjects 3 and 6 to test the application for an additional week. The results appear in Table 3.

Table 3. Application Suggestion Performance – 2nd run

Subject	Total	Right	Wrong	Missing	Not Required
3	25	16 64%	7	2	0
6	35	30 85.7%	3	1	1

While we see that subject 6 experienced worse performance comparing to the initial test where no data from other subjects was used for the training set of subject 6, (mostly since many wrong suggestions were made because of the mixed training set), subject 3 enjoyed a significant improvement in performance comparing to the initial test. Additionally, for the first time in our experiment, we witnessed a case where the system suggested a profile change when no change was actually necessary (for subject 6). We think that this too was caused by the mixed training set and will need to scrutinize the implications of using global training sets in our future work.

4. DISCUSSION

A seen by the results, a good majority of the application generated profile change suggestions were accepted by the users. This means that they found these suggestions useful and are open to allowing a program to make changes in the operation of their phone. Had the data shown a clear disagreement with the suggestions, such a program could not be useful.

It is clear that a more formal and robust usability study must be run in order to be able to feel more confident about the results and the issues that are brought up. We plan on doing so with a future more capable iteration of the application. Within these constraints, we think that the data shows that users were open to using our application and believe it to be helpful. Notwithstanding the above, the trial raised a number of issues which need to be dealt with before such an application will be really useful.

4.1.1 Adjusting Parameters

There is no doubt that taking arbitrary parameters for K, as well as the minimal training set, does not necessarily yield the best results. In order to fine-tune the system, we must perform many more iterations, checking the effect of adjusting those parameters, eventually optimizing the system classification.

4.1.2 NLP Module

As specified in section 2.1.4, the initial version of the machine learning algorithm was merely a proof of concept. In order to achieve more accurate classification, an NLP module should be added, so that the vectors' event feature would contain a broader range of values that can be compared to each other.

4.1.3 User Interface

As described in section 2.1.5, our system needs initial user input to help gather enough data for generating good quality classifications. From the user interface perspective it is clear to us that this is not a good solution, since a program that is meant to help minimize disruption should not create disruptions itself! Our hope is that the development of robust global training sets will enable us to shorten or even cancel this need.

4.1.4 Dealing with special cases

The problem of missing phone calls because of automatic profile changes raised the issue of being able to learn and deal with special cases. For example, the system should be able to learn that if a user accepts a call from someone consistently, no matter what profile the user is in, then subsequent phone calls from that contact should cause the phone to “put them through” even if it is in meeting or silent mode. This will not solve all problems relating to missed calls when in these profiles, but will help to ensure that the ones seen as important will raise attention. Note that such special case treatment does not only have to occur through learning - a user can manually add a special case flag to a contact which will do the same thing. This is important since learning takes time.

5. SUMMARY

The results hint that our program can help in reducing mobile phone disruptions since the suggestions it made were accepted in a wide majority of cases. The current implementation is indeed a giant leap from the initial version, which contains a rule-based system. A number of problems were identified. The major problem seems to be the need for a more accurate classification scheme, where calendar records are analyzed using NLP. Moreover, additional iterations are required in order to fine-tune the algorithm parameters in order to find their optimal values. We also need to test the application for longer time periods with more subjects, in order to monitor the algorithm’s behavior along time.

6. FUTURE WORK

As stated earlier, a number of improvements are needed in order to feel more confident that our system can in fact become a useful tool for minimizing disturbance. We identify the following:

- More testing: Performing additional experiments for longer time periods, in order to fine tune the algorithm parameters and conclude how much time is needed in order to reach a reasonable level of accuracy
- Deeper profile usage analysis: Checking what profiles people actually use (we believe many individuals only switch between "general" and "silent", for example)
- Data validation: Cross validating subjects on a larger scale (potentially creating a global initial training set which can be distributed with the application)
- Dealing with special cases: the system will learn to handle exceptional scenarios, such as bypassing silent mode when a certain contact calls
- Natural Language Processing: Adding an NLP module for more accurate phone database analysis
- More context data: Extending the feature set to contain additional information, such as GPS location or cell ID

7. ACKNOWLEDGMENTS

We would like to thank Paul Wisner and Nokia for providing us with a generous equipment grant.

8. REFERENCES

[1] Cyke64. (2007). Retrieved from http://cyke64.googlepages.com/

[2] Davidyuk, O., Riekki, J., & Ville-Mikko, R. Context-Aware Middleware for Mobile Multimedia Applications. *MUM 2004*. College Park, Maryland, USA.

[3] Google-Calendar. Retrieved from http://www.google.com/calendar/

[4] Goosync. Retrieved from http://www.goosync.com/

[5] *Handy Profiles for S60*. Retrieved 2 2008, from http://nokiae70-software.epocware.com/Handy_Profiles.html

[6] Mynatt, E., & Tullio, J. (2001). Inferring Calendar Event Attendance. IUI’01. Santa Fe, New Mexico, USA

[7] Nokia. (2007). Python for S60. Retrieved from http://opensource.nokia.com/projects/pythonfors60/

[8] Photo Contacts Pro 5. Retrieved 2 2008, from http://microsoft.handango.com/PlatformProductDetail.jsp?siteId=75&jid=432431334B12EEB287FE33DD1D7FD3FB&platformId=2&productType=2&catalog=0&sectionId=0&productId=166074

Infrastructure Requirements

The infrastructural as described in the MobileHCI 2009 web site are acceptable for us:

- Wireless Internet connectivity
- A power source
- A table and 2 chairs
- A board to pin up a poster