DailyAlert: A Generic Mobile Persuasion Toolkit for Smartphones

Andong Zhan, Jong Hyun Lim, and Andreas Terzis
Department of Computer Science
Johns Hopkins University
Baltimore, MD 21218
{andong,ljh,terzis}@cs.jhu.edu

Abstract

Mobile persuasion is designed to change attitudes or behaviors of mobile users through persuasion and social influence. With the support of cloud services and sensor devices, mobile phones are an ideal platform for inducing long-term behavior changes. In this paper, we propose DailyAlert, a generic mobile persuasion toolkit for smartphones which generalizes and automates mobile persuasion as a service for mobile applications. We believe this toolkit can help application developers to easily integrate mobile persuasion into their mobile systems.

Keywords

Mobile persuasion, smartphone

1 Introduction

Persuasive technologies are broadly defined as technologies designed to change users' attitudes or behaviors through persuasion and social influence, but not through coercion [10]. Mobile phones are an ideal platform to drive long-term change given their ability to combine low-level, personal sensor data and aggregated, community-scale information [14]. Meanwhile, considering the increasing adoption of smartphones, mobile persuasion technology increases the potential to persuade, intervene, and reward at the right time and place in many application domains, such as healthcare or environmental awareness [14].

At present, multiple mobile persuasive applications have been developed and some have been applied to daily life activities [6, 16, 17]. Their diversity is reflected both in the use of different persuasive formats (e.g., instant messages [17], email [2], games [6, 7, 16], social networks [6]) and data resources (e.g., self-reports [2, 3, 16], mobile sensors [7]).

We argue that the currently booming category of mobile persuasion applications would benefit from a generic toolkit that provides multiple mobile persuasion services and can be easily integrated to third-party applications. However, to the best of our knowledge, such a toolkit does not exist today.

This paper describes the design and the implementation of a mobile persuasion toolkit. The primacy challenge is in generalizing diverse mobile persuasive approaches. Another problem is supporting different data source types, ranging from self-reports and mobile sensor data to online datastores. Moreover, the simplicity and extensibility of this toolkit also need to be considered.

The paper makes the following contributions: **(1)** We summarize the requirements of a generic mobile persuasion toolkit for smartphones. **(2)** We propose DailyAlert which generalizes and automates mobile persuasion. **(3)** We present DailyAlert implementation on Android and demonstrate the advantages of DailyAlert in two sample applications.

The remainder of this paper is as follows. In Section 2 we list the requirements for a generic mobile persuasion toolkit. Section 3 presents the design principles of the DailyAlert system while Section 4 introduces the implementation details of DailyAlert. We describe two example applications in Section 5 and conclude the paper in Section 6.

2 Requirements

Since mobile phones are an ideal platform of persuasion, more and more mobile applications include persuasion functionality. For example, as part of a research collaboration with our colleagues at the JHU School of Nursing, we designed and deployed a custom Wii game to improve the wellness of low-income elders. However, the lack of a feedback channel used to motivate and reward the study's participants resulted in low utilization of the system. In response, we attempted to develop a mobile persuasion system which automatically reminds the elders to exercise using the Wii as well as surveys their daily health status. In a separate project, we published the Daily Nutrient Android application ([8]) to analyze the nutrient intake of FatSecret[1] users and help them to achieve nutrient balance. However, based on the statistics of user behavior, we realized that the low recording rate greatly restricts the accuracy of our nutrient analysis. Our goal is then to integrate persuasion functionality into our application, for example, to trigger users to record important meals on their Android phones and to send weekly personalized emails to users with detailed analysis and suggestions.

We believe that a generic mobile persuasion toolkit will greatly reduce development time for such applications, but only if it meets the following requirements:

Simplicity. A simple and user-friendly design increases persuasion power from the following three perspectives.

[1]FatSecret [3] is a food and diet application which helps users to record their daily meals and exercises on mobile devices or the web.

PhoneSense'11 at ACM SenSys'11,
November 1, 2011, Seattle, WA.

First of all, this toolkit needs to support easy and familiar interaction mechanisms for mobile users, with emphasis on special populations (e.g., elders and patients). Second, it should be easy for persuaders to setup and manage persuasion schedule for their users. Third, a simple persuasion API increases probability of adoption.

Personalization. Personalized timing and context are crucial to influencing attitudes and behaviors [10]. In other words, persuasion success depends on the right time and place. Mobile and sensing technologies create new potential to "know" the users' context, including their health status, locations, activities and even behaviors. A toolkit should persuade users by utilizing this personalized information.

Automation. Automatic persuasion is an essential functionality of the toolkit. It allows persuaders to provide services to larger populations by taking prescribed actions without human intervention. Automation in this context includes periodically persuading users based on pre-set schedules as well as acting based on events. Events can be defined in terms of the users' personal data (e.g., electronic medical records), relevant environmental context (e.g., room temperature), or social networks.

Extensibility. A generic toolkit must support extensible persuasion actions across the following three dimensions: timing, format, and condition. Extensible timing should support multiple options for designers, e.g., instant/periodic, conditional/unconditional, and schedule/event-driven. Extensible formats are necessary for designers to introduce new methods of persuasion. For example, some applications would prefer ads and games while others may prefer to persuade through social network interactions; some would prefer to interrupt users immediately while others prefer to exert an imperceptible influence on users' behaviors. In particular, a toolkit should preferably support not only smartphones but also PCs, tables, game consoles, and other user devices. Furthermore, extensibility helps designers to utilize third-party data resources to decide whether to trigger a persuasion action. For example, an application could invite users to schedule a medical check based on changes in their health status, recorded through their electronic medical records.

3 Design Overview

To fulfill the aforementioned requirements, we present the design principles behind the DailyAlert generic toolkit in this section. We introduce the concept of *alerts* to model mobile persuasion actions in a way that generalizes and automates mobile persuasion. We then present the DailyAlert architecture which is extensible and easy to integrate into mobile applications. Finally, we present a mechanism for expressing event-driven alerts that utilize third-party data resources.

3.1 Alert Model

In our mobile persuasion model, we use an *alert* to define the basic unit of persuasion actions, for example motivating a group of elders to take their medicine everyday at 5:00 PM via instant messages. An alert consists of three parts — *AlertTrigger*, *AlertFormat*, and *Customers* — which define when to trigger an alert (e.g., at 5:00 PM everyday), how to persuade the customer (e.g., via instant messages), and who the customer is, respectively.

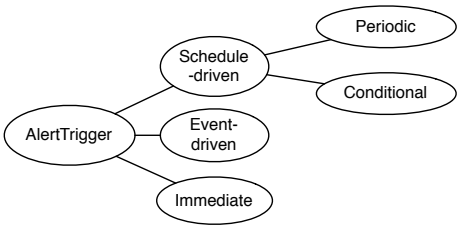

Figure 1. An AlertTrigger defines when to trigger an alert and has three basic types: immediate, schedule-driven, and event-driven.

As Figure 1 suggests, we classify AlertTriggers into three types: *immediate*, *schedule-driven*, and *event-driven*. An immediate alert will be triggered right after it has been activated. DailyAlert triggers *periodic* alerts based on their schedule. Finally, DailyAlert triggers conditional alerts based on their schedule and external conditions. For instance, a trainer motivates his students to weigh themselves via smartphones only when they fail to do so by 9 PM every Monday, Wednesday, and Saturday. In this case, DailyAlert needs to check the students' weight database and decide whether to trigger the alert. The third type of alert is event-driven, triggered when a given event occurs. An event-driven alert could be triggered by a single event, e.g., `weight > 200` lbs, or a combination of events, e.g., `weight > 200 lbs AND takenPill == false`. The end of this section describes the DailyAlert expressions used to define events and conditions.

Figure 2 illustrates the AlertFormat entity that DailyAlert defines to alert mobile users. For example, since mobile devices have limited input and output capabilities, we introduce two simple yet powerful AlertFormats — *message* and *form*. Message is used to display text messages, such as SMS, on mobile devices. Form is a data collection format, which is used to create customized questionnaires on mobile phones. To utilize the properties of smartphones, forms not only support text and single- or multiple-choice questions but also provide functions to collect GPS locations, images, audio, video, and barcodes. Furthermore, DailyAlert also enables application designers to customize new AlertFormats in XML, HTML, JSON, etc. In addition, AlertFormat is not limited to mobile persuasion but supports other networked devices and platforms. For example, we can introduce email into DailyAlert to show non-urgent but rich content alerts on multiple platforms.

3.2 Architecture

Figure 3 presents the architecture of DailyAlert in which thin clients interact with a server. The DailyAlert server provides web service interfaces to create, manage, and activate alerts. The server's scheduler is responsible for maintaining and triggering activated alerts. When an alert triggers, the scheduler pushes AlertFormat data to the DailyAlert clients that are the customers of the alert. A service that runs in the background at the mobile device parses the received alert and triggers the application based on the specific alert format. The advantages of this design are three-fold: first, a

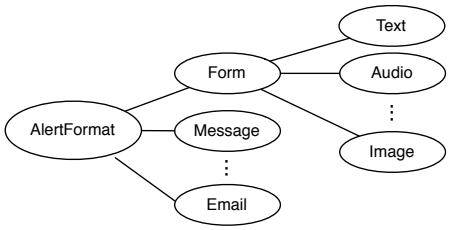

Figure 2. AlertFormat defines how to persuade mobile users. It includes two simple yet powerful formats: message and form. Taking email as an example, we also show DailyAlert can be extended to non-mobile persuasion.

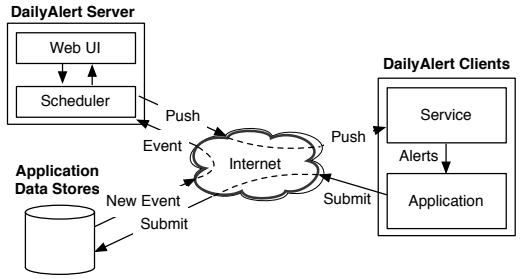

Figure 3. DailyAlert Architecture enables mobile developers to easily integrate mobile persuasion into their applications.

thin client conserves energy on mobile devices; second, a simple DailyAlert client is easy to integrate into mobile applications; third, it enables DailyAlert to support non-mobile persuasion by centralizing alert maintenance to the server. For example, it is easy to extend DailyAlert to push alerts to other online services including email, game systems, and social networks.

In addition, this architecture simplifies the definition and generation of conditional and event-driven alerts that leverage external data sources. To realize conditional schedule-driven alerts, DailyAlert actively pulls data from applications through RESTful APIs when checking conditions. This approach is comparable to most commercial datastores, such as Google Health [11], DailyBurn [2], and FatSecret. Furthermore, applications can push alerts to DailyAlert in order to trigger relevant alerts. The combination of push and pull collection methods maximizes the types of data sources that can trigger events.

3.3 Expressions

DailyAlert allows applications to express custom conditional schedule-driven alerts and event-driven alerts through *condition expressions*. A condition expression is a Boolean expression whose elements are application-specific data entities, e.g., `weight > 200`.

Furthermore, since most conditions combine several sub-conditions, e.g.,`weight > 200 AND unit == lbs`, we use a tree structure to combine these conditions. In such a tree, a union is an `AND` node with two children, an intersection is an `OR` node with two children, while an inversion is a `NOT` node while only one child.

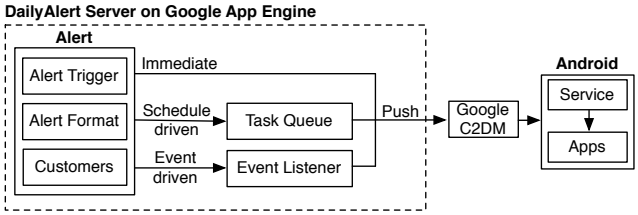

Figure 4. DailyAlert implementation.

4 Implementation Details

Next, we present the details of a prototype DailyAlert implementation. We choose Android [5] as our target mobile platform and build the DailyAlert server on the Google App Engine [12]. Android has several application components that match the requirements of our design well. Specifically, Android allows for long running background processes (known as *services*) and provides Cloud to Device Messaging (C2DM) [1] which allow the device to receive alerts from the server and forward them to related mobile applications. In addition, Google App Engine enables us to easily build, deploy, and administer the server. It also provides useful APIs, such as task queues, a datastore, etc.

As Figure 4 suggests, the implementation consists of two parts: DailyAlert server and Android service. The DailyAlert server can trigger alerts on specific Android mobile phones through Google's C2DM. To minimize the energy consumption on the users' mobile devices, we adopt a thin-client design — Android devices only need to receive alerts and forward them to a given application, while the DailyAlert server running on the Google App Engine is responsible for the maintenance of activated alerts. Another advantage of this design is that DailyAlert can also easily support other non-mobile formats, for example, emails, message to social networks (e.g., tweets on Twitter), or operations for other networked embedded devices.

4.1 DailyAlert Server

The DailyAlert server is responsible for maintaining and pushing alerts to target devices. First of all, it provides a web UI for persuaders to create, activate, or delete alerts from multiple platforms. Second, the DailyAlert server executes activated alerts based on their `AlertTrigger` types. For immediate alerts, the server pushes them to target devices as soon as they are activated. For schedule-driven alerts, the server uses the Task Queue API available from the Google App Engine to periodically trigger the alerts or check their conditions. Finally, the server registers active event-driven alerts to Event Listeners. Each event listener configures RESTful APIs to receive events pushed by devices and datastores. Third, the DailyAlert server pushes triggered alerts to their target devices based on their alert formats. For Android devices, we implement two typical formats: message and form. Messages display text on the users' devices while forms present questionnaires used to collect information from users. We utilize Open Data Kid (ODK) [13] to realize this format in DailyAlert. We also implemented a non-

mobile alert format, whereby persuaders can send emails to the users' Gmail inboxes.

4.2 DailyAlert Android Service

The DailyAlert Android service is a background process running on an Android device that receives data pushed by C2DM and passes the received messages to specific Android applications after parsing them. Although these functions can be directly implemented into an Android application activity, we choose to include them in an Android service that is independent from individual applications. The benefits are two fold: first, it is easy for Android developers to integrate DailyAlert into their applications; second, since the service is a background process, it is easy to add new functions without the users' involvement, e.g., snooze, alert acknowledgment, and statistics collection for DailyAlert.

To estimate the power consumption of DailyAlert Android service, we created a stress test which pushes 30 alerts to a Motorola Droid smartphone [9] in 30 minutes, i.e., an average of one alert per minute, including text messages and forms. We use PowerTutor [4] to estimate the power consumption in application level within 5% of actual values. The average power consumption is only 2 mW.

5 Applications

Next, we present two example applications of DailyAlert: a Wii-based home health monitoring system for elders with chronic illnesses and a mobile dietary analysis and persuasion application. DailyAlert allows these applications to quickly integrate mobile persuasion into their logic.

5.1 Home Health Monitoring for the Elderly

To monitor the daily weight and balance capability of elders with chronic illnesses, we initiated a pilot study with our colleagues in the JHU School of Nursing to deploy a customized Wii game to the residencies of six elder adults around Baltimore city for a period of two weeks. The participants were shown how to use the game in the first day of the study and were otherwise unattended. At the end of the study we conducted a survey to gauge the difficulty of using the game and all participants agreed that the game was generally easy to use. However, as Figure 5 shows, most participants practiced the game only on an average of 3.3 times out of 14 expected trials, leaving us with only a small amount of data. This experience suggests that a powerful persuasion tool is required to motivate the participants. Such a tool would notify the participants to practice daily if they forget to and ask targeted questions when abnormal events occur (e.g., weight is larger than expected).

As part of a followup study we plan to introduce DailyAlert to this application. In this use case, the nurse activates remotely a schedule-driven alert, namely DailyWeight, to check whether the participants practice every day before 6:00 PM. So every day at 6:00 PM DailyAlert pulls and checks the current day's weight and balance data from HealthOS [15], i.e., the datastore we use in our system. If the participants miss practice, DailyAlert sends a specific message to their Android phones. We also setup an event-driven alert to take actions when the weight and balance capability deteriorate. When new weight and balance data arrive from the Wii devices, HealthOS pushes the newest data to

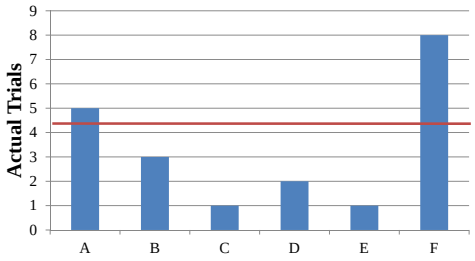

Figure 5. Low utilization of Wii game consoles. The average number of trials for six participants is 3.3 out of 14 possible trials.

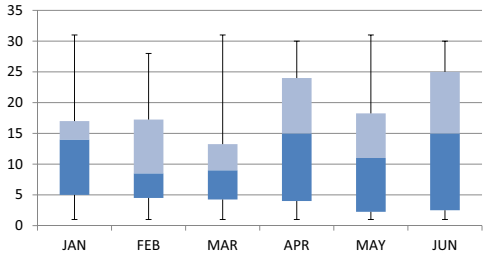

Figure 6. Box plot of active days between January and June 2011 among 102 users. An active day is a day with dietary inputs.

DailyAlert's event listener. The event listener then runs the expression engine to calculate the Boolean result. If the result is positive, DailyAlert immediately asks the participant to fill a form on her Android phone to know her physical status and potential problems.

5.2 Mobile Dietary Analysis and Persuasion

The second application for DailyAlert is Daily Nutrient, which is an Android application providing daily nutrition analysis for people using the FatSecret application to self-report and record their daily food intake and exercise. The purpose of Daily Nutrient is to help users to monitor their daily nutrient intake and finally achieve nutrient balance. We have published Daily Nutrient to the Android Market for three months where it has been downloaded by more than 100 users. However, we found that incomplete self reports complicate the accurate analysis of nutrient intake.

As Figure 6 shows, Daily Nutrient users only report on average fewer than 15 days in a month. Although using Daily Nutrient helps them increase their self reports in a month (the number of active days in the later three months are higher than the former three months), the low active days in a month still impact the accuracy of nutrient analysis. If we only consider the active months (i.e., months with more than 15 active days), as shown in Figure 7, we find that users are more likely to miss reporting their dietary intake during weekends. However, dietary intakes in weekends are significantly important for nutrient analysis. We also sample missed meals

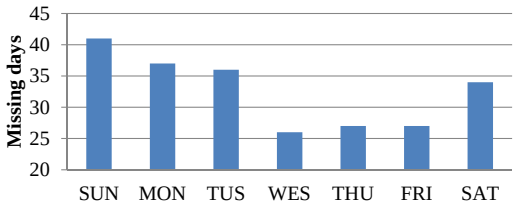

Figure 7. Total number of days with no dietary reports summed over the user population. This figure shows users usually miss their dietary inputs in weekend more than weekdays.

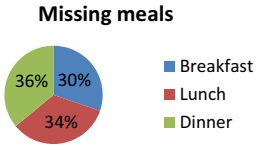

Figure 8. Missing meals among active days. It shows users miss reporting more dinners than other meals.

on active days randomly, chosen from 20 active months, as shown in Figure 8. One can see that users miss reporting dinners much more than breakfast and lunch, which makes it difficult to know whether they had dinner or not.

Given these findings we attempt to utilize DailyAlert to trigger users to improve recording their dietary intakes especially during weekends and evenings. Thanks to the schedule-driven alerts of DailyAlert, even if FatSecret datastore cannot push data to a third-party server, we can setup conditional schedule-driven alerts to pull data periodically and send alerts based on current user input. For example, we can create an alert to check FatSecret datastore every 30 minutes. To make a long-term behavior change, we use an ambient dietary display to show constant feedback of a user's current dietary state which is visible whenever users glance or interact with their smartphones.. As shown in Figure 9, the user ate a cupcake at 10:25 PM and reported to FatSecret. The alert triggered at 10:30 PM found this user was overeating, then changed the wallpaper to the right.

6 Summary

This paper presents DailyAlert, a generic mobile persuasion toolkit for smartphones. The design and implementation of DailyAlert is based on the requirements for a generic mobile persuasion toolkit targeted at smartphones: simplicity, personalization, automation, and extensibility.

DailyAlert uses an alert model to create basic persuasion actions that is extensible to multiple types of delivery schedules and formats for mobile persuasion. It follows the thin-client design to reduce power consumption on the users' phones and to simplify integration with mobile applications.

Figure 9. A passive persuasion example, using ambient display on smartphone wallpaper. The user on the right is overeating.

We present two example applications that use DailyAlert to demonstrate how it can be used in real-world scenarios.

7 Acknowledgments

We thank the reviewers for their insightful comments that helped improve the quality of this paper. This work is partially supported by the National Science Foundation under grant #0855191.

8 References

[1] Android Cloud to Device Messaging Framework. Available at: http://code.google.com/android/c2dm.

[2] DailyBurn. Available at http://www.dailyburn.com/.

[3] FatSecret. Available at http://www.fatsecret.com/.

[4] PowerTutor: a power monitor for Android-based mobile platforms. Available at: http://http://ziyang.eecs.umich.edu/projects/powertutor.

[5] Android. available at http://www.android.com.

[6] M.-c. Chiu, S.-p. Chang, Y.-c. Chang, H.-h. Chu, C. C.-h. Chen, F.-h. Hsiao, and J.-c. Ko. Playful Bottle : a Mobile Social Persuasion System to Motivate Healthy Water Intake. In *UbiComp*, 2009.

[7] S. Consolvo, D. W. Mcdonald, T. Toscos, M. Y. Chen, J. Froehlich, B. Harrison, P. Klasnja, A. Lamarca, L. Legrand, R. Libby, I. Smith, and J. A. Landay. Activity Sensing in the Wild : A Field Trial of UbiFit Garden. *CHI*, 2008.

[8] Daily Nutrient. Available at: http://market.android.com/details?id=com.dailynutrient.

[9] Droid. Droid by Motorola. available at http://www.motorola.com/Consumers/US-EN/Consumer-Product-and-Services/Mobile-Phones/Motorola-DROID-US-EN.

[10] B. Fogg. *Persuasive Technology: Using Computers to Change What We Think and Do*, page 183. Morgan Kaufmann Publisher.

[11] Google. Google Health. available at http://www.google.com/health.

[12] Google App Engine. Available at: http://code.google.com/appengine.

[13] C. Hartung, Y. Anokwa, W. Brunette, A. Lerer, C. Tseng, and G. Borriello. Open data kit: Tools to build information services for developing regions.

[14] N. D. Lane, E. Miluzzo, H. Lu, D. Peebles, T. Choudhury, and A. T. Campbell. A survey of mobile phone sensing. *IEEE Communications Magazine*, September 2010.

[15] J. H. Lim, A. Zhan, and A. Terzis. Poster: Healthos: A platform for integrating and developing pervasive healthcare applications. In *Mobisys*, New York, NY, USA, 2011. ACM.

[16] J. P. Pollak, G. Gay, S. Byrne, E. Wagner, and D. Retelny. It's Time to Eat! Using Mobile Games to Promote Healthy Eating. In *Pervasive Computing*, pages 21–27, 2010.

[17] text4baby. Available at: http://www.text4baby.org/.