J. Biomedical Science and Engineering, 2016, 9, 20-28
http://www.scirp.org/journal/jbise
ISSN Online: 1937-688X
ISSN Print: 1937-6871

AndroidCare: A Simple and Low Cost Assisted Living Solution

Abraham Otero, Alejandro Escario, David G. Márquez, Gabriel Caffarena, Rodrigo García-Carmona, Ana Iriarte, Rafael Raya

Department of Information Systems Engineering, University San Pablo CEU, Madrid, Spain
Email: abraham.otero@gmail.com, aotero@ceu.es

How to cite this paper: Otero, A., Escario, A., Márquez, D.G., Caffarena, G., García-Carmona, R., Iriarte, A. and Raya, R. (2016) AndroidCare: A Simple and Low Cost Assisted Living Solution. *J. Biomedical Science and Engineering*, **9**, 20-28.
http://dx.doi.org/10.4236/jbise.2016.910B003

Received: March 17, 2016
Accepted: September 20, 2016
Published: September 23, 2016

Abstract

Population aging places a growing stress on society's resources. There is a need for Assisted Living (AL) technologies that allow the elderly to live independently as long as possible. The AndroidCare open source project aims to explore what functionality can be provided in a low cost AL solution where no professional health organization is involved in the deployment or maintenance of the solution, nor in supervising the patient; all these tasks are carried out by a relative of the elder. Therefore, in the system's design simplicity of use has prevailed over having a lot of features. It is based on standard off-the-shell commodity hardware (a smartphone) and it provides 1) assistance to the elder in complying with the treatment of chronic diseases; 2) monitors and alerts of the occurrence of risk situations such as falls; and 3) simplifies the supervision of the elder's therapy and behavior by the caregiver.

Keywords

Assisted Living, Aging, Chronic Diseases, Behavior Monitoring

1. Introduction

Population aging and the growing stress it places on national health systems are a major challenge in a growing number of countries. In the medium term horizon, an untenable situation has clearly appeared. In Spain, for instance, just ten years ago there were two working-age citizens for every non-working-age citizen. In 2050, each working age citizen will have to support one non-working age citizen [1]. Most developed countries are in a similar situation. It is clear that the current model of health care, of welfare state, and of society in general, will have to adapt to the new circumstances [2].

A key factor in this change is the development of Assisted Living (AL) technologies that allow the elderly to live independently as long as possible, but without decreasing their quality of life. Most AL initiatives are based on a top-down approach: the state through its public health system or some private health care organization deploy the new technologies in elderly patients' homes, and health professionals are responsible for checking up on the patient [3] [4]. The installation is a complex task performed by qualified engineers that involves installing and configuring hardware for patient monitoring, the set-up of virtual private networks for sending data to hospitals, and the installation of effectors to help the patient perform some tasks... Moreover, several visits to the patient's home to explain how the system works and to troubleshoot problems are generally required [5]. Installation and use of the system without the support of engineers and/or health professionals is not viable due to its complexity.

Involvement of the family in caring for the elderly will be necessary to support the demographic reality by the middle of the XXI century, even when those family members do not live in the same house or even in the same city as the elder. There is a need for solutions that empower the family to provide the care needed by the elders. While top-down AL approaches arising from health systems will be part of the solution to the aging problem, they will not be enough when a single worker has to support one unproductive citizen. Top-down approaches will need to be complemented by bottom-up approaches led by the family and/or friends of the elderly. The goal of AndroidCare is to provide support for these bottom-up approaches.

The AndroidCare open source project [6] aims to explore what functionality can be provided in a low cost AL solution where no professional health organization is involved in the deployment or maintenance of the solution, or in supervising the patient. It will use standard off-the-shell commodity hardware—a smartphone—and the caregiver will typically be a relative of the elder. The following section presents AndroidCare's general architecture. Section 3 describes its main functionality and Section 4 describes the tests conducted to verify the proper operation of the system. Finally, the results are briefly discussed and future lines of work are proposed.

2. AndroidCare Architecture

Figure 1 shows the overall architecture of AndroidCare. The system has two different user interfaces (UI), one for the elder and another for the caregiver. The elder UI is a native Android application. Nowadays there are two dominant mobile platforms: Android and iOS. We have chosen Android for the following reasons:

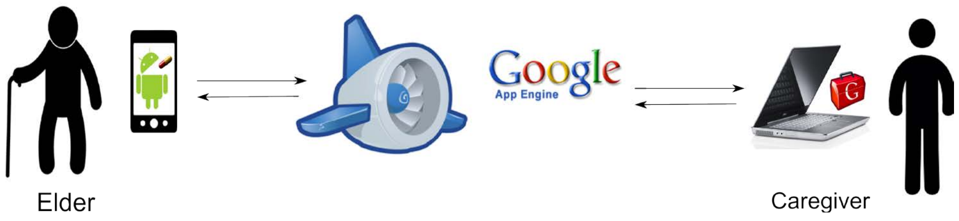


Figure 1. AndroidCare's general architecture.

- iOS-based devices are considerably more expensive than the low-end Android devices; therefore, choosing iOS would limit access to the solution.
- All iOS devices are based on touch, which is not ideal for every elder. There are Android devices with physical keyboard.
- iOS platform prevents apps from accessing some functionality that is needed in AndroidCare. For example, in iOS it is not possible to make a call or send a text message without user interaction. This would prevent reporting an alarm without the interaction of the elder when he is unconscious or sick.

The caregiver UI is a web interface. It allows the caregiver to supervise the elder's compliance with his treatment, to find out the elder's location at any time, to define reminders and to set up alarms that supervise for risk situations. This web interface is built with Google Web Toolkit.

The backend was built on the Google App Engine cloud computing platform. It has REST APIs that are consumed both by the Android application and the AJAX web client. It uses OAuth 2.0 for authentication and access control and Java Persistence API as data access layer.

3. AndroidCare Functionality

3.1. Reminders

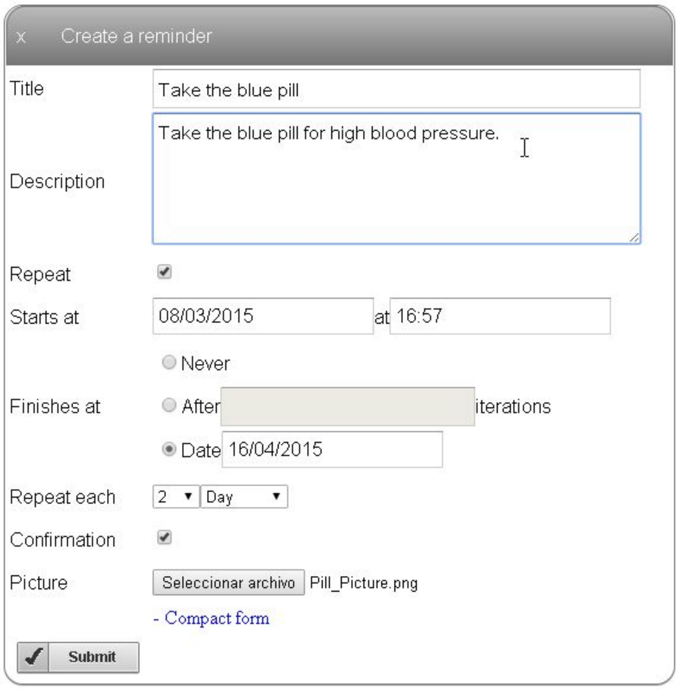
Failures in following therapies such as forgetting to take pills or taking the wrong pill are a major cause of loss of effectiveness in the treatment of chronic diseases, hence the interest in the reminder functionality. In addition, it is not unusual that elderly people forget to perform some tasks, such as visiting the doctor.

Through the web UI the caregiver can define different reminders for the elder. These can be used to remind the elder about any repetitive or occasional task; for example, "Take the blood pressure pill every day at 9:00 and at 15:00", or "On April 26, 2015 at 10:30 you have a doctor's appointment". Besides text, reminders may also have associated a picture of, for example, the pill to take. The user interface simplifies the definition of repetitive reminders (e.g., a pill that has to be taken every other day, see **Figure 2**). The definition of the reminder is sent from the web client to the backend. Every six hours the Android client checks for new reminders and downloads them.

Reminders alert the elder when he must perform some action through a configurable combination of vibrations, a ringtone melody, displaying text, reading the text, and/or displaying a picture associated with the reminder. The elder must acknowledge or dismiss the reminders. A log of the action taken by the elder, or the fact that reminder has been ignored, is sent to the server. From the caregiver UI, a log of these actions can be consulted, allowing the caregiver to supervise the elder's compliance with his treatment.

3.2. Location Supervision

It is quite common for patients with dementia to go through a stage in the evolution of their condition in which they still can enjoy a high degree of independence, but occasionally they get disoriented and get loss while performing an everyday task around



Title	Take the blue pill
Description	Take the blue pill for high blood pressure.
Repeat	✓
Starts at	08/03/2015 at 16:57
Finishes at	Never / After iterations / [✓ Date] 16/04/2015
Repeat each	2 Day
Confirmation	✓
Picture	Seleccionar archivo Pill_Picture.png

Figure 2. Form of AndroidCare's web interface used to create reminders.

their home, such as going for a walk to the park or shopping at the local supermarket.

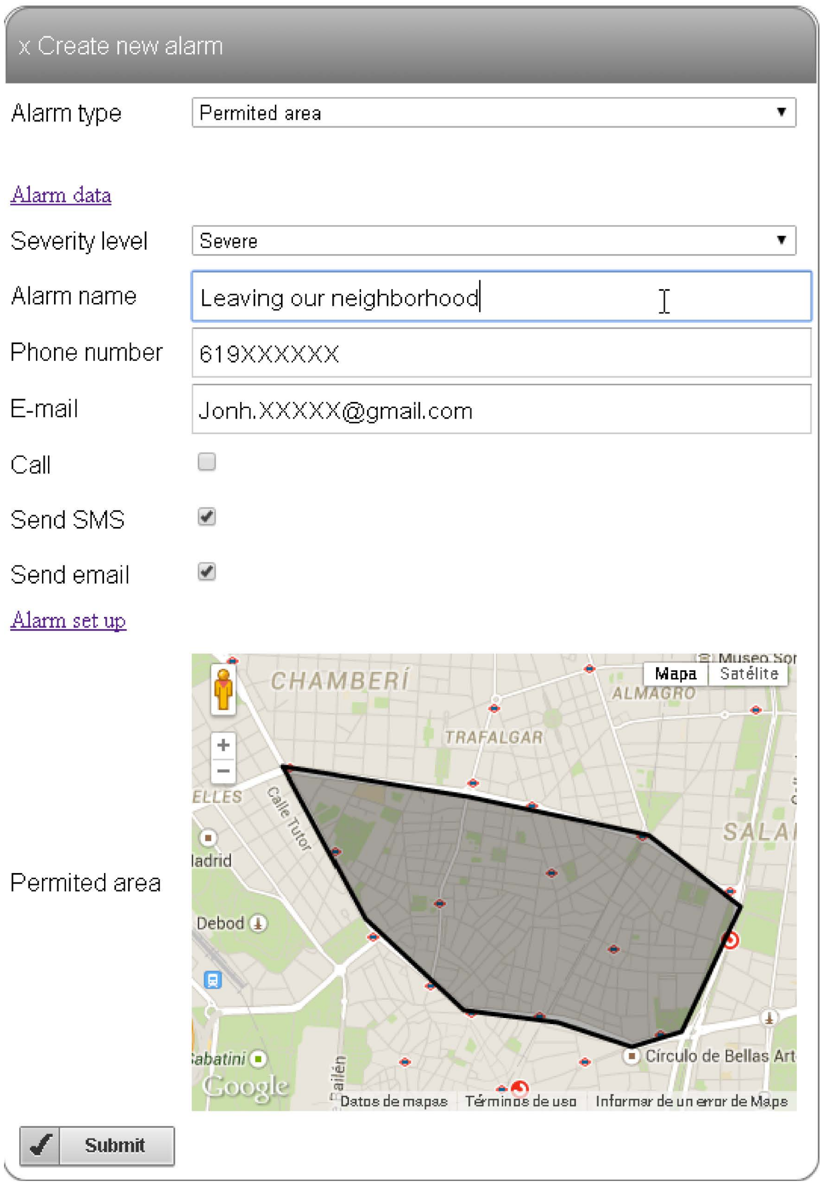
Smartphones have location sensors based on cell tower triangulation, Wi-Fi networks and GPS. The AndroidCare client application obtains the location of the phone every 5 minutes (the time interval can be configured; this is the default value). This feature can be disabled. If it is enabled, locations are sent to the server, where the caregiver can consult them over a map built on top of the Google Maps API. The caregiver UI also permits the definition of alarms that are triggered if the elder leaves a certain geographical region; this feature will be explained in more detail in the next section.

3.3. Alarms

Smartphones are equipped with acceleration sensors which allow AndroidCare to monitor potentially dangerous situations for the elder, such as falls. They are also equipped with location sensors that can be useful to detect that an elder is lost and unable to return home.

The AndroidCare platform defines a standard life cycle for alarms; the main stages in this lifecycle are: *created*, *activated*, *fired*, *dismissed*, *fired and un-notified* and *notified*. When an alarm is *activated* the trigger conditions are being continuously monitored. This is the only customized step in the alarm's life cycle; creating a new alarm only requires the implementation of this functionality.

If trigger conditions are met, the alarm state transitions to *fired*. When an alarm enters this state, for 60 seconds the application gives the elder the opportunity to cancel it (presumably because it is a false positive) or trigger it immediately. If the alarm is canceled it transitions to the *dismissed* state; a log of the incorrect firing is created and sent to the server, and the alarm goes back to the *activated* state. If there is no intervention by the elder for the 60 seconds, or if the elder indicates that the alarm should be triggered immediately, the alarm transitions to the *fired and un-notified* state. In this state, the Android app tries to alert of the caregiver. The currently supported notification mechanisms are sending a text message, initiating a phone call, and/or sending an e-mail (see **Figure 3**). In the caregiver user interface any combination of these notification channels can be selected for each alarm, as well as the appropriate configuration information (phone number or e-mail). Once the alarm has been successfully notified, the alarm transitions to the *notified* state. A log of alarm firing is created, which will be sent to the server, and the alarm transitions back to the *activated* state.


x Create new alarm

Alarm type	Permited area

Alarm data

Severity level	Severe
Alarm name	Leaving our neighborhood
Phone number	619XXXXXX
E-mail	Jonh.XXXXX@gmail.com
Call	☐
Send SMS	✓
Send email	✓

Alarm set up

Permited area

Figure 3. Form of AndroidCare's web interface used to create an alarm showing the definition of a location alarm.

AndroidCare currently supports three alarm types: **wake up**, **fall** and **position** alarms. The wake up alarm supervises that the elder wakes up before a certain time (for example, the elder must get up no later than 9:00 am). If the elder has not awakened at the usual time he may be sick and in need assistance. This alarm has been created based on a personal experience of one of the developers of the project involving one of his parents. The alarm uses the acceleration sensors to identify the typical phone movement related to being in a pocket, instead of being still in, for example, the nightstand.

The position alarm supervises that the elder does not leave one or more predefined geographic regions. Using the caregiver UI, these regions can be drawn on a map (a Google Maps instance embedded in our application) with the mouse (see **Figure 3**). This alarm fires if the elder leaves these geographic regions. The regions are stored as polygons and to check if the elder is within them the ray casting algorithm is used [7].

The last type of alarm supported at this time is the fall alarm. The fall detection algorithm applies a derivative to each of the three axes of the accelerometer sensor, then it integrates the three signals into a single signal, and checks if the final value exceeds a certain threshold for at least 300 milliseconds. This algorithm worked well on the phone used for its initial design (a Google Nexus 4), but it had a very poor performance on other phones. On some, it was too sensitive, firing, for example, when the user was going down the stairs. On other phones, it was too insensitive and did not identify all falls. The cause of the problem was that although theoretically all accelerometer sensors of all phones should provide the same readings during the same fall, this is not true in practice.

To solve this problem we have designed a calibration process that must be performed on each phone. The first time the phone downloads the fall alarm from the server, a notification is displayed in the notification bar warning that the alarm is not calibrated and, therefore, it is not running. Through this notification the user is instructed on how to calibrate the fall detection alarm. He should go to the bedroom, pick up the phone, reach out over the bed with his arm at an angle of 90˚, and drop the phone. Based on the information obtained in this controlled fall the threshold to be used by the fall alarm is customized.

The second problem with the fall alarm is battery consumption. To save battery, smartphones may turn off the CPU if the user is not interacting with the phone. This means that any code (in our case the fall detection code) that is running in the background may stop running. The Android API allows an application to get a CPU lock, preventing this behavior. But in this state battery usage increases considerably. A standard solution for this problem in applications that monitor the physical activity of the user, such as Google Fit, is to use a sampling strategy. The application wakes up periodically (e.g., every two minutes) to check if the user is performing some physical activity. The application assumes that if at that moment the user was performing a physical activity, he also was performing it during the previous sampling interval. This is an acceptable strategy when the goal is to have an estimation of for how long the user has run or cycled throughout the day. But if the goal is to detect a fall, which could last only a few hundred milliseconds, this is not an acceptable strategy. At present we are still working to find a strategy that yields a reasonable battery life for the phone while ensuring the detection of all falls.

3.4. Communications with the Server

Network communication is one of the main factors in battery consumption on smartphones. An application must have a correct policy for network usage to avoid waking up the phone and turning on the network continuously. To achieve this, all AndroidCare communications are managed from a single centralized service. Every message to be sent to the server is assigned a priority level (INFO, MEDIUM or HIGH), and then sent to the communications service. If there is no HIGH priority message, the communication service only sends messages to the server every 15 minutes, accumulating all the messages generated during this period (e.g., several logs of the elder's position). When it's time to send the messages to the server, if there only are INFO priority messages in the queue, the service tries to connect once to the server, and if it fails it waits for 15 minutes before trying again.

When there are MEDIUM priority messages in the queue, if communication fails it retries for a total of three times. Each attempt to establish a connection starts the http connection from scratch. On mobile devices it is common for the network connection to change continuously, for example from the cell network to a Wi-Fi network. This can cause objects related to the network connection that were cached to be incorrectly configured to operate with the current active network. If the communication service receives a message with HIGH priority, it tries to send it to the server immediately. If the connection fails, it attempts to reestablish the connection every two minutes until the communication is successful.

4. Platform Testing

The Android client has been tested on the smartphones Samsung Galaxy SII, SIII mini, IV, Google Nexus 4 and HTC One. The functionality that is more terminal dependent is the fall detection. During the development of the fall detection algorithm, the falls were simulated by placing the terminal in the pocket of a dummy. Later, tests were also conducted with human volunteers that simulated falls on a mat.

To date, there has been a limited testing of the platform beyond testing under laboratory conditions. Before beginning a more exhaustive testing with elderly volunteers we want to address the battery consumption problem when fall detection is enabled. The complete solution has been tested in the real world by five non-elder volunteers, adding up to a total of 800 hours of use. During these tests, there was only one false positive of fall detection. The system worked as expected in every way, although battery consumption when fall detection was enabled was very high (battery lasted 8 - 10 hours).

An important feature in AndroidCare is to be simple enough to be used without any support. From the AndroidCare website [6] the user, typically the caregiver, can create a new account. In this process he must use must the same Google Account to be used in the Android smartphone. Once the caregiver has logged in, he is presented with a control panel divided into three sections: reminders, alarms and map. The map simply displays the positions of the elder over time. The reminders section permits the definition of new reminders, and checking whether the elder has acknowledged the past reminders. The alarm section enables the caregiver to define the three types of currently supported alarms (wake up, fall and position). The definition of the reminders and alarms is based on intuitive forms (see **Figure 2** and **Figure 3**).

The whole on boarding process should be intuitive enough to be completed by a user with basic computer skills without support. Two of the five volunteers who have tested the platform in the real world were not engineers and had no relation with the project. They both managed to complete the set up without support.

Regarding the Android app, there is no set up beyond installing it on phone. Once it has been installed, it automatically downloads the reminders and alarms, and starts sending the position to the server. This transparency in the set-up is achieved by using the same Google user in the phone and in the web application. The only additional set up that may be needed if the user wants to use the fall alarm is the calibration process.

5. Discussion and Future Work

Our initial tests show the feasibility of building a AL platform based on commodity hardware (an Android smartphone) that can be employed by the end users without the support of health care professionals or engineers. While the functionality of this platform is limited compared to other solutions that can be found in the literature, AndroidCare strengths are its simplicity, no need of support and low cost (zero if the elder already has an Android phone).

As future work, we are working in finding a strategy that yields a reasonable battery life while the fall detection functionality is enabled. We also want to try to monitor unusual behavior patterns, such as the elder being completely still for 15 minutes in the kitchen or in the bathroom, which is probably an indication of an accident or an ailment. By combining information from indoor location techniques and the acceleration sensors, we believe it is possible to develop this functionality.

Acknowledgements

This work was supported by the University San Pablo CEU under the grant PPC12/2014.

References

[1] Walker, A. (2010) The Emergence and Application of Active Aging in Europe. In: *Soziale-lebenslaufpolitik*, VS Verlagfür Sozialwissenschaften, 585-601. http://dx.doi.org/10.1007/978-3-531-92214-0_22

[2] Lee, R.D. and Mason, A., Eds. (2011) Population Aging and the Generational Economy: A Global Perspective. Edward Elgar Publishing.

[3] Park, N.S. (2009) The Relationship of Social Engagement to Psychological Well-Being of Older Adults in Assisted Living Facilities. *Journal of Applied Gerontology*, **28**, 461-481.

http://dx.doi.org/10.1007/978-3-531-92214-0_22

[4] Rashidi, P. and Mihailidis, A. (2013) A Survey on Ambient-Assisted Living Tools for Older Adults. *IEEE Journal of Biomedical and Health Informatics*, **17**, 579-590. http://dx.doi.org/10.1109/JBHI.2012.2234129

[5] Wood, A.D., Stankovic, J.A., Virone, G., Selavo, L., He, Z., Cao, Q., *et al.* (2008) Context-Aware Wireless Sensor Networks for Assisted Living and Residential Monitoring. *IEEE Network*, **22**, 26-33. http://dx.doi.org/10.1109/MNET.2008.4579768

[6] Androidcare Website. http://www.androidcare.org

[7] Hapala, M. and Havran, V. (2011) Review: Kd-Tree Traversal Algorithms for Ray Tracing. *In Computer Graphics Forum*, **30**, 199-213. http://dx.doi.org/10.1111/j.1467-8659.2010.01844.x

[8] Virone, G., Alwan, M., Dalal, S., Kell, S.W., Turner, B., Stankovic, J.A. and Felder, R. (2008) Behavioral Patterns of Older Adults in Assisted Living. *IEEE Transactions on Information Technology in Biomedicine*, **12**, 387-398. http://dx.doi.org/10.1109/TITB.2007.904157