PAS: A Wireless-Enabled, Cell-Phone-Incorporated Personal Assistant System for Independent and Assisted Living

Zheng Zeng, Sammy Yu, Wook Shin and Jennifer C. Hou
Department of Computer Science
University of Illinois at Urbana Champaign
Email: {zzeng2,sammyyu2,wookshin,jhou}@uiuc.edu

Abstract

Advances in networking, sensors, medical devices and smart phones have made it feasible to monitor and provide medical and other assistance to people either in their homes or outside. Aging populations will benefit from reduced costs and improved healthcare through assisted living based on these technologies. However, these systems challenge current state-of-the-art techniques for usability, reliability, and security. In this paper we present the PAS open architecture for assisted living, which allows independently developed third party components to collaborate. Furthermore, we incorporate cell phones in PAS as the local intelligence in order to enhance the robustness and ubiquity. We discuss key technological issues in assisted living systems, such as software architecture layout, power preserving, security and privacy; and results from our pilot study in a real assisted living facility are presented.

1 Introduction

The aging of baby boomers has become a social and economical challenge. According to MIT's magazine TECHNOLOGY REVIEW, July/August 2003, "In the United States alone, the number of people over age 65 is expected to hit 70 million by 2030, doubling from 35 million in 2000, and similar increases are expected worldwide." This predication has been corroborated by the Digital 4Sight's Health-Care Industry Study which indicates "Along with the increase of elderly people population, the expenditures of the United States for health-care will project to rise to 15.9% of the GDP ($2.6 trillion) by 2010." Unless the cost of senior care can be significantly reduced by technological means, it could bankrupt the already shaky social security and Medicare systems.

A major potential saving in the financial drain of senior care is the manpower, money, and efforts taken to care for elderly people who are still capable of living independently with modest assistance, but are consigned to expensive nursing homes. When a person starts to age, his/her capabilities to sense and interact with the environment, such as memory, eye sight, hearing, dexterity and mobility, deteriorate. In addition, many also suffer from chronic diseases that require medication and clinic visits on a regular basis. If a low-cost, robust and open assisted living environment could be developed that exploits *inexpensive* technologies to enable elderly people to regain their capability of independent living, the senior care costs could be significantly reduced.

Thanks to rapid technology advances in sensing (for tracking, presence idenification, and localization), computing, and wireless networking (e.g., Bluetooth/IEEE 802.11 enabled devices and infrared-equipped remote controls), many component technologies needed for realizing such an open environment are already available. Moreover, new and improved devices and applications are expected to emerge. Although the need to facilitate assisted living has recently receiving increasing attention and there exist several R&D projects, those projects either target for population with certain disease [15, 14, 8] or daily life monitor using sensors [11, 12]. *What is missing is the design, implementation, and evaluation of a low-cost, robust, secured and open environment that allows disparate technologies, software components, and wireless devices of different protocol families to work together in a low cost, dependable, and secure fashion with predictable properties.*

In this paper, we present our experiences in designing, developing, and deploying such a wireless-based software infrastructure, called the *Personal Assistant System (PAS)*. In a nutshell, PAS exploits inexpensive, "off the shelf" technologies to assist elderly people to maintain the capability of independent living through time-based reminders of daily activities, non-intrusive monitoring of physiological functions and mobility profiles, and real-time communications with remote care providers and clinicians. After giving an overview of PAS (Section 2), we introduce the PAS software architecture that has been laid with the objectives of openness/extensibility, security/privacy, robustness, and user-friendliness, and elaborate on the functionality of its building blocks (Section 3). Although most of the objectives have been met, PAS is subject to the single point of failure (of Internet access). To improve the robustness (to network connectivity) and the ubiquity of PAS, we then discuss how we incorporate cell phones into PAS as the local intelligence (for data aggregation/acquisition) that exploit GPRS and cellular network coverage for communications (Sections 5.2–5). Use of the cell phone as the local intelli-

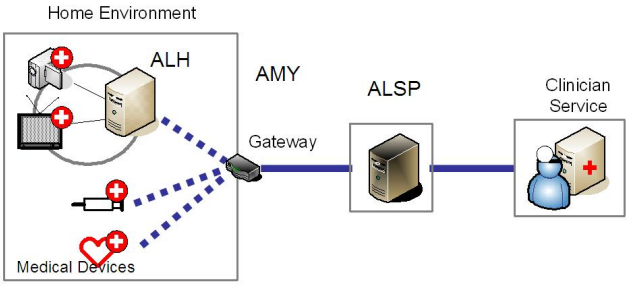


Figure 1. PAS Overview

gence is motivated by the fact that, as reported by Gartner's report (August 1, 2007), worldwide cell phone usage will top 3.5 billion connections by 2008. We elaborate on how we (i) preserve, with a mobile and power-constrained cell phone as the local intelligence, openness/extensibility and security/privacy features of the original PAS architecture; (ii) leverage features already available on cell phones, e.g., *Short Message Service* (SMS), to facilitate communications; and (iii) quantify, based on real-life calibration of cell-phone power consumption, the design trade-off of different communication modes and design communication strategies accordingly.

We have also worked with geriatricians at Washington University in Saint Louis in evaluating operations of PAS in Nazareth Living Center for Assisted Living with patients with diverse medical needs, and will report the pilot study result in Section 6.

2 PAS Overview

Figure 1 shows the overall architecture of PAS. We envision an open environment in which a security-enhanced, assisted-living device called *Authentication Manager for You (AMY)*, co-exists with a home PC or a specialized device called the *Assisted-Living Hub (ALH)*. Equipped with one or more wireless interface cards (IEEE 802.11, Bluetooth, Ultra Wide Band, and Infrared), the ALH hosts the PAS software infrastructure and serves as the intelligence for the environment. It also communicates over the Internet with the *Assisted Living Service Provider (ALSP)* at which healthcare providers and medical experts can retrieve/analyze data, monitor the resident, and if necessary give instructions to the resident. In some sense, we follow the ADT security systems model in which a black box (i.e., the ALH) is installed in the home environment, is responsible for coordinating all the services, and communicates with the ALSP. Such ALH devices are nascent today but will be commonplace in future years. Finally, wireless-enabled medical meters, consumer electronics, and/or sensor devices are introduced into the environment after being appropriately authenticated. They are responsible for collecting various physiological functions, mobility profiles of residents, and/or giving proper reminders and instructions to residents. With this architecture, several applications can be provided:

Application 1: PAS can help residents with performing daily activities. For example, the ALH can obtain from the ALSP updated prescription and appointment records of residents. When it is time for the resident to carry out their time driven routines, the ALH locates active wireless-enabled devices (e.g., PDAs, digital frames, cell phones, wearable headsets) and sends reminder messages to one or more devices that are in the proximity of the resident. Whether or not these routines are followed as advised is detected by either asking the resident to tap the keyboard of the ALH or exploiting sensor localization and tracking technologies.

Application 2: A number of physiological functions critical to maintaining homeostasis for different medical conditions can be measured by Bluetooth-enabled medical meters, transmitted to the ALH and then to the ALSP to be evaluated by healthcare providers. These physiological functions include blood glucose in the management of diabetes, daily body weights in the management of congestive heart failure, O_2 saturation in the management of pulmonary disease, and blood pressure in the management of hypertension. Measures will have a prescribed desired range and deviations from that range will generate an alert from the ALSP to the health care provider. This alert system enables prompt intervention before the situation deteriorates to a point requiring hospitalization or may simply be in the form of additional instructions to the resident to increase or decrease a medication.

Application 3: PAS can profile elderly people's movement in a privacy preserving manner (e.g., without the use of surveillance video cameras) and detect falls and early warning signs of illness, such as Alzheimer's disease and Parkinson's disease. The spatio-temporal movement data can be collected without intrusion of privacy, and transmitted to the ALSP. Residents can also wear sensors equipped with accelerometers and converted to bracelets/watches or pendants. In the case that the combination of the speed and orientation changes of the body matches typical profiles of a fall, an alert message will be sent to a designated healthcare provider.

Application 4: With the combination of ultrasonic/RFID technologies as the underlying sensing mechanism for real-time tracking of objects, PAS is able to locate personal belongings attached with tags, such as eyeglasses, hearing aids, key chains, and purses/wallets. When a resident cannot find their belongings, they can issue a simple vocal command to the ALH which then helps locate the object.

3 PAS System Architecture

We have designed the PAS system architecture for the home environment, and implemented four applications: time-based reminder service, physiological function monitoring service, fall detection and emergency help service, and human mobility tracking and profiling service. The first two applications leverage Bluetooth-enabled medical meters), while the latter two applications exploit MicaZ motes [5]. Because the main intent of this paper is to introduce the PAS software architecture (and its enhancement to incorporate cell phones as local intelligence), we focus on the first two applications. Most of the descriptions hold true in the case that the latter two applications are also included,

except that the ALH has to be equipped with the Zigbee stack (and its corresponding interface card, e.g., the Motorola LSR module, for communications with MicaZ). The interested reader is referred to [7] for a detailed account.

Our design and implementation aims at achieving the following goals:

1. **Openness and extensibility:** The environment should be open and allow new, authenticated devices to be readily plugged in and integrated with the system with a wide variety of data transport protocols. New applications and functionality should also be added to the application framework (through the use of rich, well-defined APIs)

2. **Safety, robustness and availability:** Critical services will be failure safe, and delivered in spite of the failures of useful but non-critical services. Moreover, the system as a whole will have high availability and robustness.

3. **Security and privacy:** Medical and personal data should be protected with different levels of information disclosure to different users (healthcare providers, medical team, relatives, and residents). Because wireless networking will be the predominant communication medium, security mechanisms will be built in in both information storage and communication facilities.

4. **Light-weight, easy-to-use HCIs for elderly residents:** The user interfaces should be unobtrusive, easy-to-use, safe, accommodating with respect to user mistakes.

In what follows, we introduce each of the PAS components, their functionality, and how they collaboratively accomplish the above objectives.

3.1 ALSP

As shown in Figure 2, ALSP consists of three major components: *Encrypted Database*, *Server Framework* and *Active State Monitor*.

The encrypted database is the data repository that contains information about the residents. This data is encrypted in such a way that a database administrator who has access to the database cannot interpret the private data of the residents.

The server framework is currently written in Ruby On Rails, and contains the business logic of the system and defines two different controller entities. The *monitor tier controller* provides direct web interface (i.e., a web portal) to be used by the healthcare providers, clinicians, and designated relatives. It takes advantage of the Ruby On Rails web binding and typical web application Model-View-Controller paradigm. The model consists of Active Record definitions which act as object to relational database mappings. The view consists of HTML and CSS files that give the web portal its look and feel. The controller implements the logic of processing web requests and rendering the view. There are a multitude of functions supported by the controller that revolve around the management of residents, vital signs, reminders, clinicians, and family members. The *resident controller*, on the other hand, allows for encrypted communications (in the REST fashion) with the ALH. REST bindings over HTTP are provided by the Ruby On Rails (RoR) frameworkwhich allows for resources to be created, founded, updated, and deleted in limited-SOAP like manner.

The active state monitor is used to monitor the states of residents and alert them with impending reminders. Because HTTP is a stateless protocol, an efficient mechanism is required to alert a resident of some message exchanges initiated from the ALSP (e.g., reminder messages). This could be done by having the ALH periodically pull the ALSP for pending reminders. However, this may be inefficient especially for power-constrained handheld devices. The solution is then to devise the active state monitor, a Java component that monitors the schedules for registered clients. If the monitored ALH is the home PC (or a handheld PDA), the messages are sent through the normal TCP/IP mechanism.

3.2 ALH

The ALH is written in Java and communicates with the ALSP primarily through HTTP. It runs on top of the J2SE platform. Specifically, the ALH consists of the following components:

ApplicationManager: This component is the entry point into the system. During initialization of the ALH, the system logs into the ALSP in order to validate the credentials of the resident. This process is completed through the *ResidentWebService* component. *ApplicationManager* is responsible for managing (starting, suspending, stopping, and restarting) all the PAS components and applications. Furthermore, it provides a means for applications to find and communicate with other applications.

ConfigManager: Configuration Manager is responsible for locating the configuration file, loading it up, and making it available to the applications. The configuration file contains information about the resident, the server location, the configuration of the components, a list of applications to load, a list of medical devices that have been authenticated, and a (prioritized) list of ambient devices that can be used for reminders. This file is pre-configured based on the resident's medical needs.

Logger: The logging framework is an important part of debugging and tracing the system. Even when the system is fully operational and bug-free, it provides application developers a means of debugging their applications and calibrating the system performance.

DeviceManager: The device manager administers both measurement and reminder devices. It supports devices that use Bluetooth, WiFi, and RFID as a means of communication. Our current implementation focuses on Bluetooth devices.

The device manager maintains a list of nearby active devices which are represented as *proxies* in the device list. In the case of Bluetooth, an inquiry scan is constantly running in the background. A proxy for the device is removed and considered inactive when the device has not been seen for a pre-configured period of time (the current value is 5 minutes). An application accesses to the device manager via the application interface, through which it queries the device list and determines the pertinent devices via a configuration file. The major function of proxy is to encapsulate vendor-dependent communication semantics from applications. Specifically, there exists a set of well-known

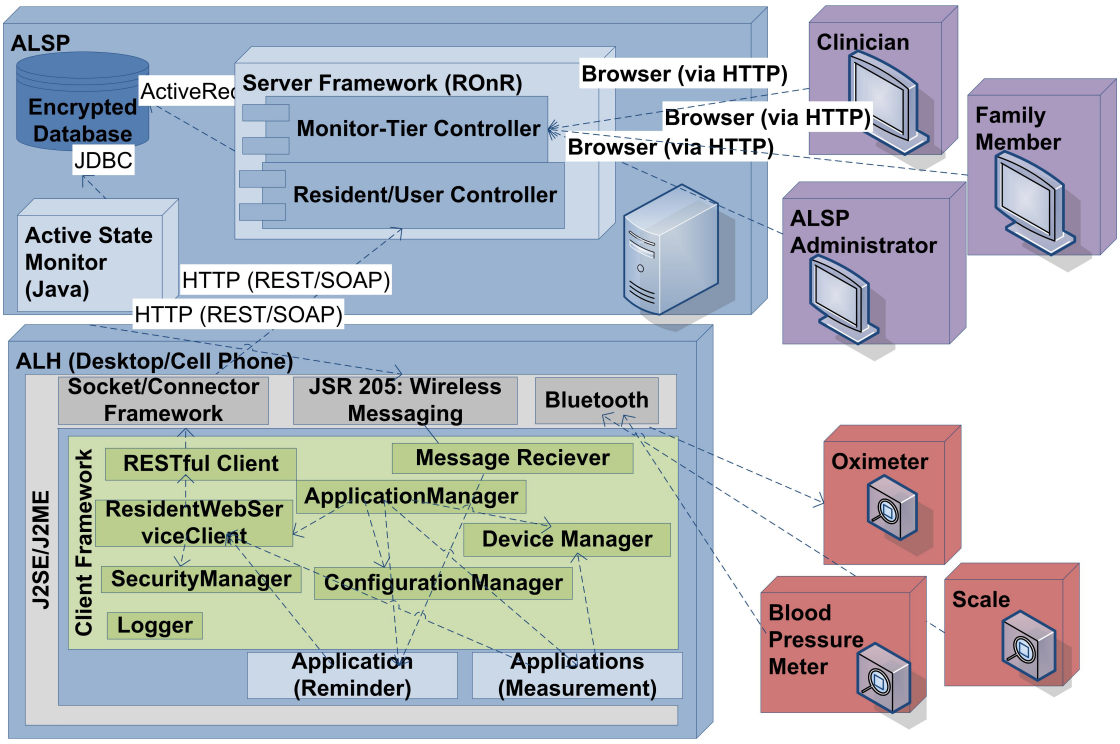


Figure 2. PAS systems architecture

APIs to access certain types of devices, such APIs to access blood-pressure meters, oximeters, and scales. A vendor for blood-pressure meters (respectively oximeters) shall provide his/her proxy code that complies with a set of prespecified APIs specific to blood pressure meters (respectively oximeters). The vendor can either write the proxy code him/herself, or assign a third party software developer to do that. Each proxy uses the underlying Communication APIs to access its device, but the "proxy-layer" protocol (i.e., the semantics communicated between the proxy and its device) is, however, vendor dependent. In some sense, proxies provide another layer of flexibility: On the one hand, applications can now be built upon well-known device APIs instead of vendor specific APIs/semantics. On the other hand, any off-the-shelf devices (especially those manufactured without the knowledge of the PAS system programming interfaces) can be integrated into PAS, as long as the vendor provides the semantics specifications on how to communicate with the device.

ResidentWebServiceClient/RESTful Client: The RESTful client is a generic REST client that allows for the 4 basic REST operations: *CREATE, UPDATE, DELETE*, and *LIST*. A ResidentWebServiceClient binding is built on top of the RESTful Client to provide the communication to (the resident controller of) the ALSP. The ResidentWebServiceClient provide semantics for logging in/off clients and sending/receiving/storing messages (e.g., querying/insertion of vital signs and querying/updates of reminder messages). The security aspects of encrypting and decrypting vital sign measurements are handled by the SecurityManager.

Security Manager: Our security requirements include two major goals. First, we have to secure the wireless transmission and protect the information from an attacker who can inject, intercept, and construct message transmissions. Second, we have to preserve privacy of medical data even when the ALSP is compromised and leaks all of the information it has. The reason why we cannot totally trust ALSP is because it it is likely to be supplied by an Information Technology (IT) specialist such as an Internet Service Provider (ISP), due to the fact that ALSP requires sophisticated networking and storage service administration. As such, it is desirable that ALSP supplier does not "know" the private medical information that it is relaying.

Message Receiver: The message receiver is the compliment of the active state monitor on the ALSP. It processes (i.e., parse in XML format) incoming messages from the active state monitor and, based on the type of messages, determines how to proceed. For example, when it receives a *Reminder List* (RL) message, it forward it to the reminder application.

3.3 Applications

The application layer is where various applications reside, and is built on top of the unified APIs and services provided by the middleware layer. Under this architecture, numerous applications can be supported as long as they specify needed services and new devices can be plugged in to provide new capabilities anytime.

4 Incorporating Cell Phones

As shown in Figure 1, one major deficiency of the PAS systems architecture is that all data transportation between ALH and ALSP is through the gateway, which may become unavailable when either the Internet service is not available

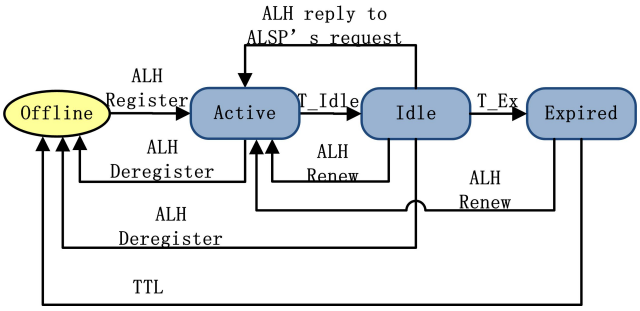


Figure 3. State Transaction

or when the resident is away from the home environment. This pretty much limits the ubiquity and robustness of PAS. To deal with this deficiency, we propose to leverage the cell phone initially as a wireless modem and later as the local intelligence for data aggregation and management. To use the cell phone as a wireless modem, we utilize the *Dial Up Network Profile (DUN)* service. Note that *DUN* provides PCs and other network devices access to a LAN or WAN via telephone lines, and is available on most current Bluetooth-enabled phones.

Leveraging the cell phone as a wireless modem does help enhancing the robustness of PAS. However, this may not be sufficient due to the following reasons. First, cell phones are battery-powered and hence are resource-constrained. Unfortunately configuring the cell phone in the *DUN* mode incurs large power consumption. For each packet transported, the cell phone must first receive it through Bluetooth and then send it via GPRS/CDMA/GSM. In the case of saturated traffic, the cell phone can run out of battery within 4-5 hours. Second, the transmission rate with the cell phone configured as a wireless modem is quite low. In our preliminary study with the use of cell phones with GPRS support, we found that the peak rate is no more than 30kB/sec, although a GPRS connection can in principle provide a throughput of 120kB/sec. This is because when the cell phone is configured in the *DUN* mode, it cannot transmit a packet via GPRS and receive another packet via Bluetooth simultaneously. Last, when a resident is away from home, what is lacking is not only the Internet service but also the ALH that serves as the local intelligence. Restoring the wireless data transport capability is not enough.

Due to the limitation of the *DUN* service, we propose to leverage the programming capability of smart phones, and enable them to become the local intelligence, i.e. replace the home PC. The PAS systems architecture is augmented with two operational modes: in the home environment, the ALH serves as the local intelligence; and in the mobile environment, the cell phone replaces the ALH and becomes the new coordinator. How to instrument battery-powered cell phones to perform a full fledged set of services in an energy efficient way is a challenging task, and will be treated in the next Section.

5 Cell Phone-Incorporated Architecture

5.1 Objectives

To incorporate cell phones into the PAS systems architecture and enable it to carry out the same set of PAS services, the following requirements have to be met:

(1) The openness and extensibility goal originally set forth should not be compromised. As in the original static-ALH architecture, a middleware layer should be laid with well-defined APIs.

(2) To ease programming loads, features already available on cell phones, e.g., *Short Message Service* (SMS) and availability of cellular networks for wider coverage, should be leveraged to the maximal extent.

(3) Since cell phones are constrained by battery power, both the applications and the underlying PAS architecture have to be tailored to achieve better energy efficiency.

(4) Due to the fact that cell phones may be more easily accessed by third parties (thus introducing security/privacy holes), security/privacy is now an even more important issue. However, security/privacy should be ensured/preserved without incurring excessive power consumption.

In what follows, we discuss how we revise the PAS systems architecture and/or design new strategies to fulfill each of the above requirements.

5.2 Architecture Modification

Thanks to the fact that most smart cell phones support the J2ME platform on top of Windows and/or Linux, we can readily port the ALH software onto cell phones, thus providing the same level of openness and extensibility as in the case that a home PC serves as the ALH. However, because the functions provided by J2ME are more limited as compared to those provided by J2SE, we have to import several third-party Java libraries in oder to port the ALH software. The most important library is (part of) the Bouncy Castle package for the J2ME platform, because it provides the encryption/decryption mechanism for securing data transmission between the cell phone and the ALSP. In addition, we have augmented the PAS software with the following features.

The first feature is to switch from Home-PC-Based ALH to Cell phone. Recall that when Internet access is not available or when the resident is away from the home environment, the cell phone should take over the ALH functionality. In the former case, we include at the home-PC-based ALH a *Network Monitor* application that periodically polls the ALSP to check network connectivity. If the network monitor cannot contact the ALSP after a pre-specified time interval, it notifies the cell phone to take over the ALH functionality. This is accomplished by searching for nearby Bluetooth devices with the PAS UUID. In the latter case, the resident activates the cell phone as the ALH with a GUI provided by the phone. The cell phone then notifies the network monitor of the role switch, and henceforth takes charge of ALH functionality.

The second is to keep track of ALH at ALSP. Because cell phones are portable devices and as a result may not have

stable GPRS connections, the ALSP needs to keep track of which ALHs are active (and thus need to be monitored). To deal with this problem, we augment the *active state monitor* at the ALSP with a *client lease* protocol. Figure 3 gives the state transitions of the client lease protocol. From the perspective of the ALSP, there are four states associated with a cell phone when it acts as an ALH: *offline*, *active*, *idle*, and *expired*. When the cell phone is offline, ALSP does not keep any record of it. When the cell phone is in one of the active, idle, or expired states, ASLP maintains a client model object, keeping track of its state, its location identifier and its *last seen date*, where the *last seen date* is the last time the ALSP received a message from the cell phone, either by *small message service* (SMS, to be discussed below) or HTTP. The state of the cell phone is based upon this last seen date.

When the cell phone takes over the ALH role, it registers itself with the ALSP via a GRPS connection. After the ALSP receives and approves the registration, it creates a a client model object accordingly. If no data communication takes place between the ALSP and the cell phone within an interval of T_{Idle}, the ALSP records the ALH state as idle, and starts to send, via SMS, *Are You There (AYT)* queries to check its availability. If one of the AYT messages is acknowledged within an interval of T_{Ex}, the ALSP updates the ALH state as active. Otherwise, the ALSP updates the ALH state as expired. Note that when the ALH is in the idle or expired state, it may initiate communication to the ALSP (which we term as ALH renew), thus moving the ALH state back to active. If the ALH is in the expired state more than an interval of TTL, the ALSP assumes that the ALH is turned off and releases the corresponding client thread. If the same cell phone would like to communicate with the ALSP as a ALH, it has to re-register itself. During any time, the ALH may also choose to deregister itself, in which case the ALSP also evicts the ALH. The active state monitor will only send reminders to an ALH that is either in the active or idle states.

Correspondingly, the message receiver at the ALH supports receipt of two types of messages: AYT and Reminder List (RL). When an AYT message is received, the ALH responds by renewing its lease via *ResidentWebServiceClient*. RL messages, on the other hand, are forwarded to the reminder application for processing.

5.3 Taking Advantage of Small Message Service

Another problem with incorporating cell phones into PAS is that cell phones are not equipped with IP addresses and hence it is difficult for the ALSP to locate, and initiate communication with, the cell phone. To resolve this problem, we take advantage of the SMS service already available on most cell phones through Internet. When the ALSP would like to communicate with the ALH but a TCP connection has not been established (by the ALH), the ALSP sends a SMS message. The message first arrives through the Internet at the *Short Message Gateway (SMG)* provided by a third party. Then the SMG forwards the message through the Internet to the *Short Message Service Center (SMSC)* of the cell phone carrier. Finally the SMSC sends the message to the cell phone through a cellular network.

If the cell phone would like to initiate communication with the ALSP, it can either take advantage of SMS or establish a TCP connection. In the former case, it may take multiple short messages to send one application packet because cell phone carriers have their own character restrictions on short messages. For example, T-mobile restricts the length of its short message to 160 characters. Note that most SMGs support two-way communications, and hence the cell phone can send messages to the ALSP as long as the ALSP purchases the SMG service. The latter case (TCP connection) is made possible if the resident subscribes to the data plan from the corresponding cell phone carrier.

5.4 Preserving Battery Power

Recall that cell phones are constrained by battery power, and hence both the applications and the underlying PAS software have to be tailored to achieve better energy efficiency. In this subsection, we discuss the strategies we use for this purpose.

5.4.1 Having the ALSP Push Messages

When a home PC serves as the ALH, the only way for the ALSP to deliver reminder messages is for the ALH to first establish TCP connections and then *pull* reminder messages periodically at a relatively high frequency. This is because the IP address of the ALH may reside in a private LAN and its IP address may be dynamically assigned (by DHCP). On the other hand, when the cell phone serves as the ALH, the same mechanism may drain the battery of the cell phone quickly and is thus impractical. In our calibration study, we found that the battery of the Motorola Bluetooth enabled PDA-cellphone, A1200, can sustain for 4-5 days in the stand-by mode, but only has a lifetime of less than 4 hours if data transmission takes place periodically over GRPS . Thus, we enable the ALSP to *push* reminder messages via SMS. One implementation challenge is how to send the message to the PAS applications on the cell phone, other than the default inbox of the cell phone. Our solution is to enable the message receiver at the cell phone to listen to a pre-specified port and the ALSP to send reminder messages to that specific port.

5.4.2 Enabling Applications to Transmit Bulk Data

In order to calibrate the major sources of power drain incurred in the GPRS communication between the ALH and the ALSP, we have carried out experiments to measure the power consumption incurred in sending HTTP posts of different message sizes to ALSP. We choose HTTP connection specifically because this is the way ALH sends data to ALSP.

The measurement device is composed of a digital multimeter (Agilent 41110A), a benchtop DC power supply (Agilent E3646A), a smart phone (Motorola A1200) and a very low resistance precise resistor ($R = 0.025\Omega$). We build up

Table 1. Power consumption incurred in the HTTP posting

HTTP Post Size (Byte)	Normalized Energy Consumption	Operation Duration (sec)
10000	445	15.2
5000	387	13.3
2500	322	12.1
1000	286	11.8
500	259	11.2
250	257	10.9

the experiment circuit by putting the resistor in series between the power supply and the cell phone [4]. The power supply provides constant 4.1V input voltage (denoted by V_{input}), and the multimeter continuously measures the voltage (denoted by $V(t)$) across the resistor at a rate of up to 10,000 times per second. Then the instant power consumption of the cell phone at time t (denoted by $P(t)$) can be indicated by one voltage reading:

$$P(t) = \frac{(V_{input} - V(t))V(t)}{R} \quad (1)$$

Let r denote the sampling rate of the multimeter, and P_{idle} denote the average power consumption of a cell phone when it is idle with screen off and default Bluetooth service on. In our calibration study, we found that when there is an active Java application running or a wireless transmission taking place, the power consumption increases one order of magnitude higher than P_{idle}, which makes it easy to determine the duration of one operation. Suppose one operation lasts for n readings, then the extra energy consumption caused by this operation is

$$E_{op} = \frac{1}{r}\Sigma_{i=1}^{n}[P(t_i) - P_{idle}]. \quad (2)$$

Table 1 shows the energy consumption and time to send different sizes of data. For each size, we periodically post the data to ASLP (with screen off) and each number in Table 1 is the average value from 20 posts. The energy consumption value is normalized with respect to the average energy consumed by one idle phone for 1 second. Figure 4 gives a detailed view of the power consumption for one HTTP post when data size is 250B, 1000B, and 10000B respectively.

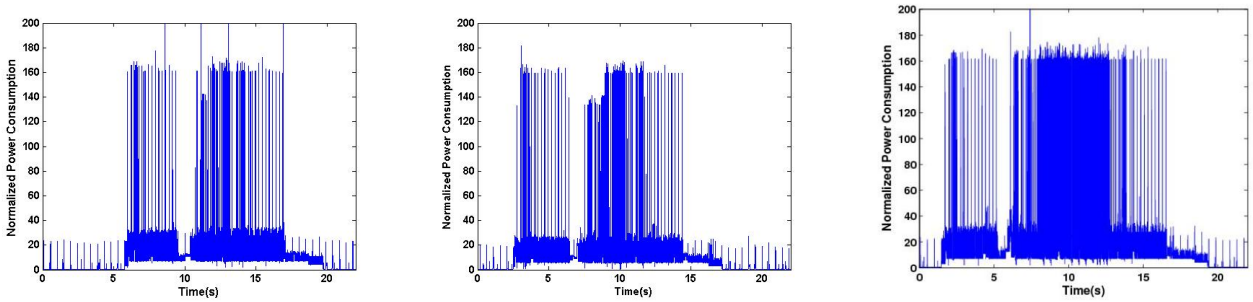
As shown in Table 1, the power consumed increases much slower than the increase in the message size. Two reasons may account for it. First, it takes around 4 seconds to build up a HTTP connection and this overhead, which doesn't vary with post size, is a major source of power drain. Figure 4 illustrates this by clearly showing that one post has two stages where power consumption reaches peak value frequently. The first stage is connection construction and the second one is "pure" data transmission. Second, the data transmission over GPRS connection needs time to reach peak transmission rate, therefore small packets are comparatively transmitted at lower rate, which matches the observation that the pure transmission time does not increase linearly with data size. Actually when the data size becomes 39 times larger, then pure transmission time only doubles as shown in Figure 4. Although when transmission rate increases, the instant power consumption reaches peak value more frequently, its influence is much less significant than the transmission time. Therefore even without considering connection overhead, the energy needed to transmit unit data is less when we adopt bulk transmission.

The experiment results suggest that, in order to transmit data in a more energy-efficient manner, it is desirable to transmit bulk data, rather than numerous small pieces of data. Enlightened by this empirical finding, we have instrumented the physiological function monitoring applications (e.g., the Nonin 4100 Oximeter application and the A & D Scale application) to be capable of local data aggregation and management. Instead of passively relaying all the measurement results, the applications are instrumented to infer the status of the resident based on a set of pre-specified rules, and only transport readings in *real-time* to the ALSP when abnormal situations are identified. If a vital sign reading is within the normal range, it will be buffered and transmitted in bulk either periodically or with the abnormal reading. The set of pre-specified rules can be downloaded from the ALSP to the phones, and can be either simply threshold-based or derived based on a more complicated correlation between readings from different devices. These rules also need to be personalized based on medical records, current health status, and medical prescriptions for the resident. This enables energy-efficient data transmission, while not compromising the need to timely transport data in case of abnormal situations.

5.4.3 Adopting Device Pushing Mode in Bluetooth Communications

With respect to the Bluetooth communication between medical/ambient devices and the ALH, there are two data transmission modes: the ALH pulls data from the device, or the device pushes data to the ALH. Specifically, in the former case (ALH pulling), the ALH first scans the environment to locate the target device. This step is necessary because a medical device is only turned on when it's being used. After ALH has located the target device, it connects to the device and retrieves measurement results from it (which may take dozens of seconds to obtain stable readings). In the latter case (device pushing), the ALH keeps a specific Bluetooth service open. Whenever a Bluetooth device obtains stable readings, it sends the data to the ALH by connecting to that service. The ALH then acknowledges receipt of the data, and the device deletes the readings from its memory.

Note that most of the devices are off-the-shelf and their communication protocols are proprietary(i.e. designed by vendors) and cannot be altered. For example, the Nonin Medical oximeter [10] adopts the ALH pulling mode, while the A & D Medical scale [1] adopts the device pushing mode. However, in our calibration study, we found that the mode in which devices and the ALH communicate impacts the energy performance quite significantly. We use the same methodology that we used in Section 5.4.2 to compare them. Table 2 shows the energy cost related to the two communication modes. The power consumption for idle state is the average value over 10 minutes, and the power consumption for various Bluetooth operations is the average value over



Data Size = 250Byte Data Size = 1000Byte Data Size = 10000Byte

Figure 4. The detailed power consumption of one HTTP post operation with different post size

20 measurements. Recall that both modes have the measurement retrieval stage, and besides that the pulling mode has a scan stage while the pushing mode has overhead for opening an extra Bluetooth service on the cell phone. We conclude that the device pushing mode actually gives better performance due to the following reasons: (1) The device discovery phase incurs much more energy cost in the case of device pushing. In our implementation the cell phone will keep scanning for the oximeter continuously after it receives a related reminder message. We set the interval between two scans zero to minimize the time that the user has to wait after he/she turns on the oximeter till the cell phone locates the oximeter. Fig. 5 shows such discovery process drains the battery at almost the same rate as a Bluetooth transmission. In the 2-week pilot study at The Windsor Of Savoy [2], we have 10 residents (with average age 88) receiving daily oximeter reminders. The average discovery phase length is 17 minutes and 32 seconds, which can consume the cell phone's energy that suffices to get 346 measurements from the scale operating in pushing mode. Even when we set the scan interval to 15 seconds, the energy this stage costs suffices to get 65 measurements from the scale. Such huge delay is mostly due to the resident not responding to the reminder quickly since they were in another room. Moreover, [13] suggests that frequent device scan aggravates the Bluetooth pollution problem. (2) More data transmissions are incurred with the use of ALH pulling, because most types of medical devices can get several unstable readings before it finally obtains a stable one. For example, it takes 3-5 readings (10-20 seconds) for the Nonin Medical oximeter to get a valid measurement. In contrast, in the case of device pushing, only after stable readings are obtained will the device initiate the connection with the ALH and it takes less then 5 seconds. The last two graphs in Fig. 5 clearly shows the difference. (3) The overhead of opening extra Bluetooth service is negligible assuming that the users tend to turn on the Bluetooth in daily life. In light of the above findings, we recommend vendors to adopt the device pushing mode in the Bluetooth communication between devices and their (likely handheld or portable) controller.

5.5 Light-Weight Security and Privacy

The ALSP and the protocols to communicate with it must satisfy basic security requirements. Since ALSP requires sophisticated networking and storage services administration, it is likely to be supplied by some IT specialist such as Internet Service Provider other than health care provider. Therefore it is desirable that ALSP does not "know" the private medical information that it is relaying. Based on the concerns above, our security requirements include two major goals.

Table 2. Power consumption incurred in ALH pulling and device pushing modes.

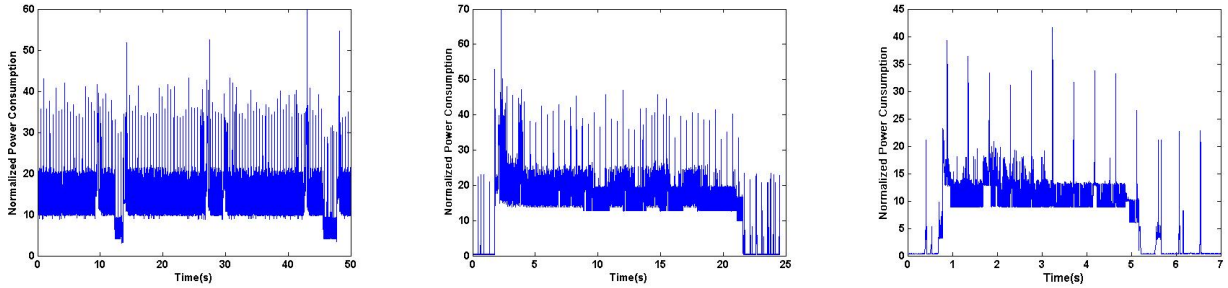
Operation	Normalized Energy Consumption	Operation Duration (sec)
Idle with BT off	0.833	1
Idle with BT on	1.0	1
Idle with our BT service on	1.0	1
BT scan	13.7	1
One measurement retrieval (oximeter)	213	12.3
One measurement retrieval (scale)	41.6	4.4

First, we must protect all information from being compromised during transmissions, which includes the Bluetooth communication between medical devices and ALH, and the data transfer between ALH and ALSP. The former is protected by Bluetooth built-in PIN mechanism. For the latter, we use Transport Layer Security (TLS) to guarantee end-to-end security over Internet.

Second, we must preserve the privacy of medical data even when ALSP is compromised and leaks all the information it has. We proposed *Drop-Box Architecture* in [9] to meet this requirement . In short, we separately encrypt the resident's medical information and administrative information, such as timestamp, so that ALSP can deliver and store the encrypted private data without knowing what's inside. The following describes how private information encryption is implemented (please refer to [9] for more details). Assume each person involved in PAS has a private key and a public key. The user u may have multiple types of private data such as heart rate, glucose level, etc. Each type t keeps an recipient list l_t such that only people in l_t are granted access to u's data. Then for every piece of private data D_t of type t, the ALH generates a structure E_{D_t} that will be sent to and stored in ALSP.

$$E_{D_t} = \{E[D_t]_{K_{session}}, KeyList\} \quad (3)$$

$$KeyList = \{(ID_i, E[K_{session}]_{Pub_i}) | i \in l_t\} \quad (4)$$



Scan for BT devices nearby | Measurement retrieval from oximeter | Measurement retrieval from scale

Figure 5. The detailed power consumption of different Bluetooh operation

Table 3. Overhead in encryption of 10Byte data

Operation	Data size/ Key size	Encrypted Data size	Duration (ms)
RSA Key Fetch	128B	N/A	N/A
RSA encryption	16B	128B	387
AES Key Generation	16B	N/A	17
AES encryption	10B	16B	32

where $K_{session}$ is a symmetric key generated per encryption, Pub_i is person i's (asymmetric) public key, ID_i is person i's unique identifier, and the expression $E[D]_K$ denotes encryption of D using Key K. Any person in l_t can use his private key to get $K_{session}$ and furthermore access D_t. In this way we protect users' private information from unauthorized parties. However, this security mechanism brings in overhead in terms of processing time (generating $K_{session}$, encryption of D_t using $K_{session}$, and generation of $KeyList$) and extra storage ($KeyList$). Table 3 shows the overhead of securing a 10Byte vital sign measurement where AES (symmetric encryption) and RSA (asymmetric encryption) are used. If the recipient list size is 5, then it takes at least 3 seconds to generate the structure E_D of size over 650 Bytes. Obviously the overhead is non-negligible and most of it comes from $KeyList$.

Fortunately, data aggregation and management proposed in section 5.4.2 can help reduce the overhead. For example, if we send buffered multiple oximeter measurements in bulk, then instead of generating and attaching $KeyList$ to each measurement, all the measurements may share the same $KeyList$. Therefore the overhead per measurement is significantly reduced.

6 Implementation and Pilot Study Results

6.1 Devices

In our current implementation, we use the following equipment:

(1) **ALSP**: We use a laptop installed with the Windows XP operating system and MySQL database serves.

(2) **ALH**: In the case of home-PC-based ALH, we use a laptop with the Windows XP operating system, Java runtime environment standard edition 1.5.0_06, and Avetana Bluetooth stack. In the case of cell-phone-based ALH, we use a Motorola A1200 cell phone (Fig. 6 (d)). Note that A1200 is a Bluetooth enabled PDA-cellphone with the Linux operating system. Its J2ME library includes implementation of JSR 82 (Bluetooth). In addition, we acquire special firmware, a faked J2ME application certificate (that can only be used with the special firmware), and programming SDK (in C language) from Motorola. The certificate enables us to sign our J2ME jar file so that the PAS applications can be trusted by the cell phone and gain access to certain protected domains such as file reading/writing. The SDK allows us to gain access to several cell phone functions that J2ME APIs cannot provide. All these features make A1200 a relatively ideal device for pilot implementation.

(3) **Peripheral devices**: There are two types of devices: (i) Bluetooth-enabled medical devices for physiological function monitoring, i.e., the blood pressure meter from A & D medical, Inc., the scale from A & D Medical, Inc., and the oximeter from Nonin Medical, Inc. (Fig 6 (a)-(c)); and (ii) Ambient devices for displaying reminder messages, i.e., Bluetooth/WiFi-enabled digital frames, a laptop (by which a piece of pre-recorded audio is displayed), or the cell phone (on which pre-recorded audio clips and images are displayed).

6.2 Pilot Study Results

With the help of geriatricians at Washington University in Saint Louis, we have carried out a pilot study at the Nazareth Living Center for Assisted Living in June-July 2006. We deployed two PAS prototypes (because of budget constraints) during a three-week pilot study at Nazareth Living Center in Saint Louis MO. The prototype is home environment version without security and privacy mechanism implemented. This facility houses 110 well-educated, predominantly female residents, whose average age was 88. Of 30 residents who attended a presentation, 14 agreed to participate. After administering a standard cognitive assessment, two residents were consented to test the PAS prototypes and ten residents to carry/wear a placeholder device for a period of two weeks.

Pilot Results: The qualitative and quantitative data from the pilot study comprise several hundred pages. Summarized below are several of our major findings that pertain to the issues considered in this article:

1) Residents found PAS useful and were willing to wear the sensing apparatus: *i*) The two residents using the PAS prototype found it to be quite useful. Residents not chosen to use the working prototype expressed their desire to use

(a) Blood pressure meter (b) scale (c) oximeter (d) cell phone

Figure 6. Devices

the working version as well, suggesting that PAS was, in general, well received by residents. *ii*) Nine of the ten residents with placeholder devices said they wore the device every day in the two week study.

2) Residents lack in confidence in PAS when it did not work properly: Since Ethernet access only exists in nurses' room but not in residents' rooms, ALH(the laptop) must connect to a wireless router put in the nurses' room(a low-end Linksys WRT54G wireless router as used in the pilot study) to access ALSP. During the study, this, coupled with concrete walls between residents' rooms and the nurse station, led to intermittent connectivity. When the wireless connection was down, sometimes the host operating system (Window XP) failed to reset the connection. Residents, as a result, were not confident in relying solely on PAS for medical monitoring. This indicated both *i*) the reliability of PAS software and *ii*) the connectivity and QoS of the underlying wireless communication have to be improved.

3) Nurses/caretakers desire interfaces that provide security/privacy. With a high resident-to-nurse ratio, nurses were usually very busy and could not ensure that the information being displayed would not be viewed by unauthorized personnel. The need for privacy should be addressed by designing adequate access control to PAS. Also, nurses inquired whether or not medical data could be securely transmitted via wireless technology. This implies they also had concerns about PAS security.

In summary, the pilot study cheers us up by proving that PAS is quite well-received by elder people. Meanwhile, it identifies several deficiencies of the PAS prototype at that time as mentioned in 2) and 3). As a matter of fact, the pilot study results motivated us to develop the mobile verion PAS with cellphone and bring in *Drop-Box Architecture* to protect users' privacy.

7 Conclusion

In this paper we present the PAS open architecture for assisted living, which exploits inexpensive, "off the shelf" technologies to assist elderly people to maintain the capability of independent living through time-based reminders of daily activities, non-intrusive monitoring of physiological functions and mobility profiles, and real-time communications with remote care providers and clinicians. PAS allows independently developed third party components to collaborate. Motivated by the pilot study at the Nazareth Living Center, we incorporate cell phones in PAS as the local intelligence in order to enhance the robustness and ubiquity, and provides basic security function to protect users' privacy. We wish that with the help of PAS, aging populations will benefit from reduced costs and improved healthcare.

References

[1] A & D Medical, Inc. http://andmedical.com/.

[2] B. AlShebli. The assisted living reminder system field study at the windsor of savoy. Technique Report, 2007.

[3] I. M. Author. Some related article I wrote. *Some Fine Journal*, 99(7):1–100, January 1999.

[4] J.-M. P. Creighton Hager, Scott Midkiff and T. L. Martin. Performance and energy efficiency of block ciphers in personal digital assistants. In *Percom*, 2005.

[5] I. Crossbow Technology. Global leader in sensory systems. *http://www.xbow.com*.

[6] A. N. Expert. *A Book He Wrote*. His Publisher, Erewhon, NC, 1999.

[7] J. C. Hou, L. Ball, S. Birge, M. Caccamo, Chin-FeiCheah, E. Gilbert, C. Gunter, E. Gunter, C.-G. Lee, KarrieKarahalios, N. Nitya, L. Sha, W. Shin, V. V. andQixin Wang, Y. Yu, and Z. Zeng. Pas: A wireless-enabled, sensor-integrated personal assistance system for independent and assisted living. In *Proc. of High Confidence Medical Device Software and Systems Workshop (HCMDSS/MD PnP'07)*, June 2007.

[8] Intel Corporation. Age-in-place. *http://www.intel.com/research/prohealth/cs-aging_in_place.htm*.

[9] M. J. May, W. Shin, C. A. Gunter, and I. Lee. Securing the drop-box architecture for assisted living. *The 4th ACM Workshop on Formal Methods in Security Engineering*, 2006.

[10] Nonin Medical, Inc. http://www.nonin.com/.

[11] U. of Rochester. Center of future health. *http://www.futurehealth.rochester.edu/news*.

[12] U. of Virginia. Smart in-home monitoring system. *http://marc.med.virginia.edu/projects_smarthomemonitor.html*.

[13] D. N. K. Somil Asthana. The problem of bluetooth pollution and accelerating connectivity in bluetooth ad-hoc networks. *PerCom*, 2005.

[14] University of Washington. ACCESS (Opportunity Knocks). http://cognitivetech.washington.edu.

[15] University of Washington. Assisted Cognition. *http://www.cs.washington.edu/assistcog*.