SPARTAN: A Framework for Smart Phone Assisted Real-Time Health Care Network Design

Shinan Wang*, Weisong Shi*
*Dept. of Computer Science
Wayne State University
Detroit, MI, USA
Email:{shinan,weisong}@wayne.edu

Bengt B. Arnetz†
†Dept. of Family Medicine and PHS
Wayne State University
Detroit, MI, USA
Email: barnetz@wayne.edu

Clairy Wiholm‡
‡Dept. of Public Health and Caring Sciences
Uppsala University
Stockholm, Sweden
Email:clairy.wiholm@pubcare.uu.se

***Abstract*—Leveraging body area sensor network (BASN) for health care is a very promising application domain for wireless sensor networks. In a typical BASN health care application, usually, bio-sensors and environmental-sensors connect to a local *Preprocessing Unit (PU)* first, e.g., a smartphone or a laptop, which in turn extracts the meaningful data and performs necessary processing before the *PU* transmits the data to a *Central Server (CS)*. In this procedure, we realized the system designers have to work on many repeated jobs in different BASN systems. Even worse, changing one component of the system usually requires designers to rewrite a large portion of code. In this paper, we present a Smart Phone Assisted Real-Time heAlth care Network framework ($SPARTAN$), to simplify the development procedure and extend the flexibility of BASN systems. In order to demonstrate the effectiveness and efficiency of $SPARTAN$, we implement a smartphone assisted stressor examination and warning system. The experimental results show that the $SPARTAN$ framework can reduce the workload with low overhead and simplify several procedures such as replacing the sensor or changing the sensor configuration.**

I. Introduction

With more and more practical and theoretical research on real-time, long-term health care systems utilizing modern computing devices and facilities, the community faces even more challenging issues, such as system architecture with heterogeneity, real-time data processing, and energy conservation. Therefore, many techniques, such as Body Area Sensor Network (BASN), pervasive health, and participatory sensing, encounter similar problems. Leveraging body area sensor network (BASN) for health care is a very promising application domain for wireless sensor networks. In a typical BASN health care application, usually, bio-sensors and environmental-sensors connect to a local *Preprocessing Unit (PU)* first, e.g., a smartphone or a laptop, which in turn extracts the meaningful data and performs necessary processing before the *PU* transmits the data to a *Central Server (CS)*. The goal is building a data-oriented, real-time, people-centric, mass computing systems that benefit the majority of people [5]. For example, central systems monitor the health condition of elderly people in real-time remotely with low cost in the future. On the other hand, the professionals are able to take advantage of BASN to access the physiological data for research purpose [1, 4].

Given a lot of research work proposed to build the architecture, framework, and health care-related applications based on BASN, there are various sensors and solutions to choose. Usually, the design of a BASN system would involve hardware implementation, software implementation, and integration of both. This fact may result in a long term design cycle. In addition, the established systems lack of flexibility. For example, if a hospital, which equips with the advanced remote patient monitoring system, decides to upgrade sensors, there may be some extra work have to be done on other parts of the system. Moreover, the theoretical research of wireless sensor network tends to be mature so that currently it is vital to develop killer applications. We commonly regard that the application scientists usually focus on usability rather than complexity, which often becomes the neglected part of the system design. Our community needs a technique to speed-up the development of health-care applications for common users.

In this procedure, we realized the system designers have to work on many repeated jobs when they design different BASN systems. Even worse, changing one component of the system usually involves rewriting a large portion of code. In this paper, we propose to build a framework, $SPARTAN$, which is short for Smart Phone Assisted Real-Time heAlth care Network framework, to facilitate the process of designing a BASN as well as adding flexibility to the system. Usually, the commonly used technique to achieve efficiency lies on module design. We adopt this approach to develop a framework in application layer. We expect the proposed framework would be easy to use, effectively reduce the workload of building a system, and introduce less overhead. The $SPARTAN$ framework consists of *Communication Module, Data Retrieve Module, Data Processing Module*, and *Feedback Module*. The *Communication Module* along with the *Data Retrieve Module* are in charge of receiving data and extracting information. The *Data wecessing Module* applies different preprocessing mechanism to the collected data stream, such as data compression. We implement Feedback and warning system in *Feedback Module*.

Our contribution are three-folds. First, as far as we know, the framework proposed is the first one touching the field of ease the development of BASN applications. Second, we utilizes module design in the framework to extend the flexibility of systems and provides efficient means to adjust the system.

For example, simply changing the parameter in the *Overall Controller* does the work for replacing a commercial sensor product. Finally, we developed a physiological monitoring system to evaluate the proposed framework.

The reminder of the paper is organized as follows. In Section II, the existing approaches and previous research are briefly discussed. We then present the $SPARTAN$ framework in Section III, followed by a prototype of smartphone assisted stress examination and warning system based on the proposed framework in Section IV. The challenges and opportunities of the similar systems are described in Section V. Finally, we summarize the paper and describe future work in Section VI.

II. Related Work

The wide spread mobile devices and the fact that various multi-functional sensors becoming ubiquitous, make the long-term monitoring and self-management possible. The foundations of urban sensing are built upon mobile devices, which are equipped with powerful computational components, as well as, multiple communication methods such as Wi-Fi and Bluetooth [9]. Two facts have pushed the system design in mobile computing to a new age. Participatory sensing turns people-centric systems to become feasible. Mobile devices are utilized to build a real-time, human-machine interactive system that provides more diverse and current data to professional to gather, analyze, and share the information. Machine learning techniques and algorithms, along with the recognition of their value in system design, increase the chances of better understanding the huge volume of data collected throughout the world. As a result, a number of projects have been launched targeting this field.

An elder home helper system is developed in [19]. The major components of the system are a pen type image sensor, an Internet client computer, and a wireless Internet mobile phone. A handwritten care request from the elderly person triggers the system, and the request is further transmitted to the server computer in the Home Helper Central Office from the client computer via Internet. The server computer automatically sends the request to the Home Helpers' mobile phones. In this way, the system offers a emergency service to the elderly people with minimal effort. In [10], a simple electrocardiogram (ECG) diagnosis algorithm is used at the cell phone with a wireless dongles to monitor the physiological signs of the patient. The signal is finally transmitted to the medical center via LAN or CDMA. The authors try to detect life-threatening arrhythmias with software assists for analyzing P-wave, QRS complex, and T-wave of the ECG signals. A MIMOSA architecture is presented in [20]. Regarding as a open platform for Ambient Intelligence, the system includes four hardware types: terminal devices, sensor radio nodes, RFID sensor tags, and back-end servers. They deploy two types of health care applications on top of the platform: ECG acquisition and Glucose level monitoring. [13] shows a biomedical digital assistant, which tries to satisfy several criteria: portability, wearability, minimal size, weight, and power consumption. In the prototype, the researchers use ECG, PPG signals to monitor patient's states. Trying to recognize the daily activities, [11] demonstrates mechanism detecting the lifestyle of the user by employing wearable sensors, which likely decreases the occurrence of chronic diseases. As an real BASN application, the system presented in [23] consists of sensors, stargate gateways, iPAQ PDAs, and PCs. Each query triggers data collection process. Based on the environmental and physical data, the system determines circadian activity rhythms of residents, and feedback the system to design the context-aware sensing system.

Several BASN system architectures have been proposed [3, 14]. The former platform has been implemented in a real scenario, Johns Hopkins hospital, to monitor the heart beat rate and blood oxygen levels of Emergency Room patients [12]. Another platform enables motion capture application in BASN is presented in [8]. Basically it contains six-degree-freedom sensing hardware components. In order to processing data more precisely, a modified realtime operating system is integrated in the platform. In [17], the hurdles that encountered in implementation a BASN is demonstrated by designing a prototype that enables heart beat rate and activity monitoring. Several issues are considered in the prototype such as energy conservation, synchronization, and data processing.

However, how to evaluate the design for such system remains an issue. As [15] in their project CodeBlue points out, the reliability of the network becomes the highest priority in system design. A dedicated tool is implemented in [18]. The goal is to help the designers understand the network behavior of each node in terms of signal strength and delay efficiency. Recently, the research community realized that it is necessary to compared the two mostly used technologies in BASN, IEEE 802.15.4 (Zigbee) and Bluetooth. [21] gives a more comprehensive study about the two technologies. Overall, the Bluetooth outperform IEEE 802.15.4 in terms of goodput while IEEE 802.15.4 is more energy efficient. However, the Bluetooth provides "Sniff" mode in order to reduce the operating power.

On-chip data processing not only reduces the communication burden but also provides convenience for the data analysis components. However, a certain level energy increase would be observed because of inappropriate trade-off between computation and communication [7]. Two families of data compression algorithms, Huffman encoding and delta encoding are evaluated in [6]. Dynamic delta encoding is chosen to adaptively change the size of delta bits, which represent the difference between the current reading and its predecessor. Basically, this approach guarantees a reasonable compression ratio and relative lower computation workload. For a particular type of data, such as ECG data, [16] points out that some of the leads in the ECG data are redundant since the complete information can be reconstructed by only part of them. Therefore, additional compression gain is obtained.

By implementing the wireless health care applications, researchers encounter more challenges [22], such as infrastructure reliability, context awareness, service quality and pervasive feasibility.

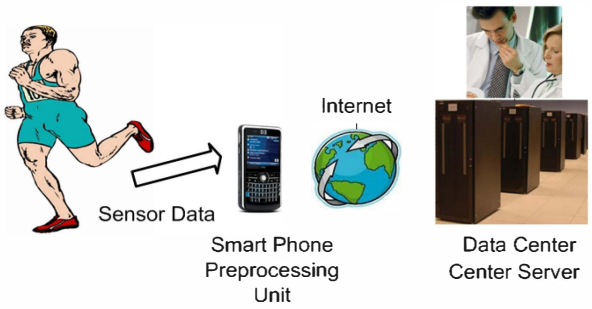

Fig. 1. A typical scenario for health care monitoring applications.

III. System Design

In general, most of the systems assume the body area sensors are connected to the *precessing unit (PU)* at first. *PU* works as a gateway to communicate with the *central server (CS),* which is the storage and command center for the whole system. A PDA or a smartphone is used as a typical *PU*. Between *PU* and *CS*, or sensors and *PU*, a system probably has several connecting points to form a sensor network [3]. Though adding sensor networks would increase both system scalability and reliability, the networking issues are beyond the scope of this paper. We assume the three tiers, the sensors, *PU*, and *CS*, connect to each other based on certain networks. The basic system architecture is shown in Figure 1.

A. System Architecture

In $SPARTAN$, *PU* becomes our primary optimization target due to the following reasons.

The *PU* is the direct controller of the sensors. On one hand, sensor manufacturers are unwilling to let the system designers change the hardware directly, since malfunctions are likely to occur. On the other hand, system developers need extra knowledge to modify the hardware, which probably delays the design cycle. Hence, it is common that the sensor manufacturers let the sensors receive commands from the *PU* in order to change their configurations. As a result, *PU* becomes the primary target for us to customize.

Compared with *CS*, *PU* needs to be more flexible to adopt changes. The former serves the public, but the latter usually is a personal device. Therefore, the functions between the *CS* and *PU* are prone to be fixed. For example, a patient monitoring diabetes sometimes would like to change his/her diabetes meter with a more advanced one, which requires modifications on *PU*. But the service provided by *CS* probably remains the same.

As more powerful as the *PU* becomes, more and more functions are moved into *PU*. Some of the techniques originally built in sensors have migrated to *PU* since they are not implemented in the commercial products. For example, there is no standard data compression algorithm or security mechanism currently; therefore the manufacturers usually do not support them and leave the workload to the system designers. Some techniques are transfered from *CS* to *PU* in order to alleviate workload on *CS* and achieve shorter response time. For instance, some simple algorithms detecting the condition and state of a patient can be easily implemented on *PU*. In addition, the data can be stored in the *PU* before they are transmitted to the *CS*. Obviously, the functionality achieved on *PU* comes with the trade-off of energy. As a result, which function and what mechanism to support are purely dependent on the system designer. We only consider the basic functions in our framework because other functions can be easily added as modules into our framework. Based on these facts, we build the $SPARTAN$ framework.

B. Overview

The framework is demonstrated in Figure 2. Because of the aforementioned reasons, the whole framework mainly benefits the *PU* development. In the center of the framework, a *Overall Controller (OC)* contains all the necessary, application-specific, and personal information, which guides the behavior of each module. The information is dispatched to different modules at the initial stage of the system by the *Center Parser (CP)*. The framework assembles the system information hierarchically to enable adaptive system design. As Figure 3 illustrates, there are two main data flows in the system. One of them represents the sensing data, another one represents the reverse data flow, such as commands or feedbacks from servers. Adding one more module in the system, the information needs to be inserted into the appropriate place. A more clear system overview can be obtained from the structure. Even better, each module is easily to be located, changed, and deleted.

The *Communication Module* is designed to diminish the redundant work have been done in implementing communication functionality because of different communication protocols (usually IEEE 802.15.4), such as Bluetooth and Zigbee. The *Data Retrieve Module* targets on extracting necessary information from the raw data of sensors. It is common that different companies design sensors would have different data formats. Hence, it is non-trivial effort that a developer should put on to deal with the information. The *Data Retrieve Module* simplifies this procedure. Usually, the system designers suggest the retrieved data have to be processed before further investigation. The preprocessing can reduce the total amount data that either need to be sent to the central server or stored locally. The *Data Preprocessing Module* simply provides unified interfaces to the applications and perform the most often used data preprocessing techniques. In some physiology monitoring systems, the feedback is of the most significance component of the system. Therefore, the *Feedback Module* provides a feedback mechanism as questionnaires. On top of the four modules, the *Overall Controller (OC)* is responsible to present the interfaces to various applications as mentioned. In reality, the systems might need more modules than we design, but one of the benefit using module design is that necessary functions can be added independently. For example, a *Security Module* can be plugged in the system to encrypt the data before the application using *Communication Module*.

In general, $SPARTAN$ works as follows. The system

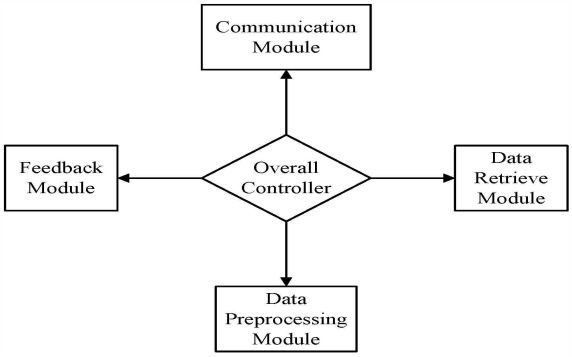

Fig. 2. The five components of the $SPARTAN$ framework.

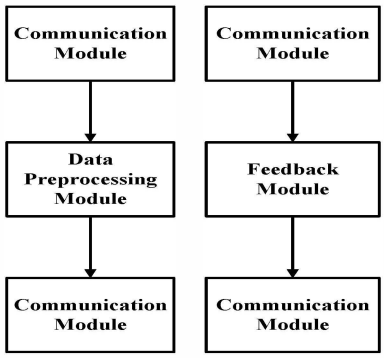

Fig. 3. An example data Flow.

initializes with extracting the information from *OC* using *CP* and loading information onto different modules. The data flows comply with the structure defined in *OC*. For example, the left part in Figure 3 defines the data flows coming from the sensors, through the Bluetooth or Zigbee, reaching the *PU*, then being transmitted to the *CS* after preprocessing.

In our design, the initial goal is to find a popular tool being supported on different mobile platforms. J2ME fits our purpose well. First, most widely used devices incorporate J2ME, such as Symbian, Blackberry, and Windows Mobile. Also, there are various JSRs packages available, such as security (JSR 219), GPS(JSR 179), and Bluetooth (JSR-82). By doing that, we are trying to include a larger set of mobile devices candidates.

C. Overall Controller

The *OC* stores all the information necessitating a basic BASN. Three main criteria have to be implemented in the design process.

- Clearly defining a language for expressing characteristics, and semantically organizing the attributes of each module are our first priority.
- We also expect to emphasize simplicity, generality, and usability to make it work naturally in a heterogeneous environment.
- At last, as we mentioned in Section I, the *OC* has to be easy to change, so that adding one module, deleting one module, and changing one module would not be troublesome.

```
Class Communication {

INITIALIZE (String protocol, String sensor);

SETCOMMAND (byte[] command, int size);

RECEIVEPACKET(byte[], int size);
}
```

Fig. 4. The communication interfaces provided by communication module.

Because the *OC* is closely related to each module, we will introduce partial of it across the rest of this section along with different modules, finally form a complete view of*OC*. Our design goal is to make most of the system information exposed to the application through the *OC* component. XML-liked file is used to fulfill our purpose.

D. Communication Module

Usually, from the system design point of view, the communication functionality is trivial. However, the implementation in real system could be troublesome. Firstly, in order to implement different communication protocols, repeated work are required. For example, most of the commercial sensor products equipped with either Bluetooth or Zigbee. A unified communication module is expected to hide the details about the communication protocol and provide interface to the system designers. In addition, for a particular sensor product, it might support different data transmission rate and application command set. We designed interfaces that provides basic communication functions as well as different application level commands.

The Figure 4 shows the mostly used interfaces. First, the system needs to initialize the communication from *PU* to the sensors with appropriate protocol and sensor name. If success, the basic operation is to receive the packets from the sensors. The commands are sent to the sensor to adjust the data transmission rate and configuration occasionally. These three operations are the basic ones to utilize a commercial sensor product in BASN and sometimes the only ones functionally supported.

Bluetooth and Zigbee are two most commonly used protocol stacks in BASN. In terms of language support, JSR-82 is a dedicated J2ME component API for Bluetooth, but none of the most popular smartphones support Zigbee currently. It is encouraging to see that the research community attempts to overcome this obstacle. Recently, some researchers designed JXBee, a Java driver for Maxstream XBee and XBee-PRO 802.15.4/ZigBee wireless modems [2]. In addition, these two popular technologies, 802.15.1 (Bluetooth) and 802.15.4 (ZigBee), have been comprehensively compared with each other in terms of design cost, performance, and energy efficiency [24].

By conclusion, although Zigbee is designed as low power dissipation while Bluetooth technology delivers better goodput at a similar success rate.

E. Data Retrieve Module

Among the data transmitted from the sensors, usually there are two parts. One is the vital signs, such as heart beat rate, blood pressure, and ECG. Another part contains sensor information, such as remaining battery and current working state of the sensor. The objective of adding the *Data Retrieve Module* to the system is to eliminate the repeated procedure that extracts meaningful data: the vital signs, and reports the extra information to the system developers.

$SPARTAN$ consists of a bit granularity data extraction mechanism. Based on each packet received, the developers can specify which bit(s) are needed to obtain meaningful data. For example, the blood pressure information can be stored in several different bytes and needs some basic calculations. The designers are asked to input the required information into the *Overall Controller (OC)*, the *Center Parser (CP)* will retrieve the information to form several available APIs. The format is *Retrieve(name,*$bit_{pos(N)}$,$bit_{pos(N-1)}$.....bit_1*)*. For example, *Retrieve("Blood Pressure",125,210,211,212,213,214,215,216, 217)* means the name of the data is "Blood Pressure" and

$$BloodPressure = bit_{125}*2^8+bit_{210}*2^7+bit_{211}*2^6+bit_{212}*2^5$$

$$+bit_{213}*2^4+bit_{214}*2^3+bit_{215}*2^2+bit_{216}2^1+bit_{217}*2^0$$

F. Data Preprocessing Module

The most common used data preprocessing techniques are data compression and data aggregation. The primary concern is that the communication energy consumption can be reduced by applying those technologies[7]. The trade-off must be made between the computation and communication. It is ideal to choose a data compression algorithm with less computation and higher compression ratio. Usually, the on-chip data compression is desirable since the data size can be controlled from the beginning. However, most sensor products on the market do not support such functionality because a standard data compression algorithm can be hard to achieve from the industry point of view. First, the compression ratios of collected data sets can vary. Second, it puts burden on system designers to handle different algorithms to decompress the data. Third, the system flexibility would be limited even if data compression saves energy.

Hence, we argue that the data compression can be put into the *PU* because of the above reasons. We implemented two mostly used data compression algorithms in this module, Huffman Encoding and Delta Encoding. Before transferring the data to the *CS*, we expect to provide options to the system designers to reduce the data set size. The system designer can specify which one to use in the *OC*. The choice could be none, Huffman, or Delta.

G. Feedback Module

The feedback to the end user can come from dedicated abnormal detection algorithms from *CS*, which are based on expertise of a particular symptom. Other simple algorithms are also non-trivial, especially, when detecting the accumulated effect of numerous tiny activities of the end users. Those simple feedback mechanisms can be implemented on the *PU*. The feedback module is divided into two parts, the feedback trigger and the feedback content.

Usually, some basic statistical results are needed to implement the trigger and the trigger is a combination of multiple criterion. It is computation expensive to calculate the statistical value for a long term monitoring system. Thus, we use window to specify the limit that needs to be considered. For example, $get_average(HeartBeat, N)$ returns the average heart beat rate value of the last N number of readings. N is the window size. $get_correlation(HeartBeat, BloodPressure, N)$ is used to calculate the correlation of the "Heart Beat" and "Blood Pressure" in the last N readings.

The feedback content is implemented as string of warning or reminders. The contents are defined in the *OC* with a specific name for each of them. Combined with the trigger, the feedback mechanism can be implemented as

$$if(current\_reading1 > get\_statistical1$$

$$(data\_name1, window\_size))\&\&$$

$$if(current\_reading2 > get\_statistical2$$

$$(data\_name2, window\_size))$$

$$trigger(warning)$$

H. Overall Controller

Given the four commonly used modules, it is easier to understand the reason why *OC* is needed. *OC* organized the information together. 5 demonstrates an example of *OC*. At first, the basic information about the sensors is collected. Then, the system designer can specify which data compression can be utilized for a specific sensing data set. In this example, "1" represents the Delta algorithm while "0" stands for Huffman algorithm. The feedback consists of the sensing data name, statistical variable, and the threshold. The *Center Parser* is able to get the information and dispatch it.

IV. Case Study: Stress Examination and Warning System

We target on evaluating the usability and effectiveness of $SPARTAN$. Therefore, we design a stress examination platform using our framework. The motivation of this work is to understand the relation between the physical human body stress and the subjective feelings. We expect the framework is able to reduce the workload of system designers, to offer flexibility to the system, and to satisfy the application specification effectively.

```
<?xml version="1.0" encoding="UTF-8"?>
<OC>
<sensor>
    <name>sensor_name1</name>
    <communication>Bluetooth or Zigbee</communication>
    <framesize>frame size</framesize>
    <configuration>#1</configuration>
    <data>
        <value_name1>data retrieve information</value_name1>
        <value_name2>data retrieve information</value_name2>
    </data>
    <name>sensor_name2</name>
    <communication>Bluetooth or Zigbee</communication>
    <framesize>frame size</framesize>
    <configuration>#2</configuration>
    <data>
        <value_name3>data retrieve information</value_name3>
    </data>
</sensor>
<compression>1</compression>
<compression>1</compression>
<feedback>
    <sensor>sensing data name</sensor>
    <trigger>average/variance,threshold</trigger>
    <question>feedback to the user</question>
</feedback>
</OC>
```

Fig. 5. An example of Overall Controller.

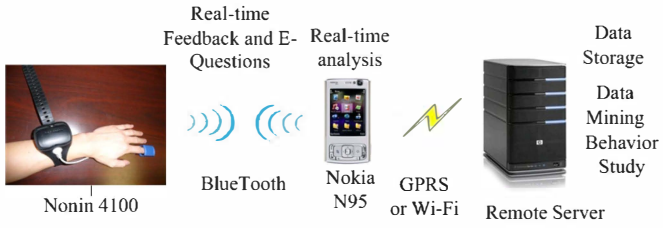

Fig. 6. The prototype of the system.

A. Prototype

As Figure 6 shows, our prototype is a 3-tier architecture including sensors, a smartphone, and a remote server. The first tier provides physical data collected from the patients. It is worth noticing that the smartphone, for some models, can act as multiple sensors as well. In this case, there is no need of the communication module. The body area sensor network communicates with the smartphone via Bluetooth.

The number and type of the sensors are determined by the system functions. In the stress study, we utilize Nonin 4100 as sensor to measure heart beat rate and SPO_2. We focus on heart beat rate in the study since it more likely physically represents the stress level of human body. The oxygen saturation range is from 0 to 100% and the pulse rate range is from 18 to 300 beats per minute. In the implementation, we use configuration $NO.2$ of the Nonin 4100; Hence, the Bluetooth interface of the Nonin 4100 is $5bytes$, $75per/sec$, $8-bitpleth$.

The second tier contains a PDA/smartphone. In our implementation, we use a Nokia N95, which supports Bluetooth v2.0 with A2DP and Wi-Fi 802.11 b/g. N95 equips a very powerful ARM 11 dual CPU. Internal storage is approximately 160 MB with 64 MB RAM. Java Technology, including JSR 118, JSR 139, JSR 82, and roughly ten more J2ME technologies, are supported by the N95.

A remote server is set up to store all collected sensing data. After these data are collected, they will be stored in a formatted data warehouse. Time-series data mining algorithms are deployed at the server to discover the time-series patterns and rules in all collected data. Multi-modality data mining algorithms are utilized to mine the correlation between the biomedical, environmental, and location data. The specific algorithm incorporated in the server is beyond the scope of this paper. We implemented the aforementioned four modules and use the framework to develop the system.

There are some specific requirements of the stress examination system. In Table I, six activities with specific feedback mechanisms are listed. $Duration$ is the time duration of each test. Feedbacks are displayed on the smartphone when the heart beat rate reaches criteria listed at $WarningsandFeedback$ column. During the $Easystresstest$ and $Difficultstresstest$, we ask the participants to continuously calculate subtraction to mimic the office environment. $Max.amountofwarnings$ is self-explanatory. We set up $Min.timebetweentwowarnings$ for replying the questionnaire. The random questionnaire is sent because of the requests by professionals to better understand the underlying pattern.

The contents of the feedback questionnaire are listed in Table II. We have six questions focusing the current feeling of the participant. The first question tries to validate the system is synchronized with the participant.

In summary, the system works as follows: First, the information is collected from a Nonin 4100 attached to the patients' bodies. Secondly, a smartphone, Nokia N95, acting as a gateway, collects data from body sensors and transmits them to a center server. It then runs certain simple algorithms to trigger the feedback. In addition, immediate feedback and warning are displayed on the screen. In this case, when certain criteria are satisfied, the participant's feeling is collected using a feedback message on N95. The feedbacks, which contain six queries with six possible answers, are demonstrated in Table I. Finally, a more powerful database, running on a remote server, is in charge of data recording and applying more specific algorithms towards the collected data.

B. Experiments and Results

The framework is evaluated based on three categories. First, how easy to use; Second, how much overhead; and how flexible it is.

First, we demonstrate the effectiveness of the framework. The whole *Communication Module* along with the*Data Retrieve Module* can be used in most BASN applications to avoid repeated work. In addition, the data compression algorithms can be modified easily in *OC* to spped up the preprocessing. According to the feedback module of our framework, the criteria to trigger the questionnaire can be adjusted dynamically based to the needs of the domain experts and the users easily. In addition, the questionnaire can be changed as well. In the current test, we are using six activities, that can be changed to a very large scale of daily activities. For example, if the person is exercising between 6 am to 7 am, we could set

	Duration	Heart Rate(HR)	Warnings and Feedback	Random number of warnings	Max. amount of warnings	Min. time between two warnings
Relaxation	5 min.	Continuous	After 4 min. and 30s	0	1	N/A
Easy stress test	5 min.	Continuous	HR $>$ 20% of baseline or when HR $<$ 10% of baseline for the next 5 beats	1	5	45s
Difficult stress test	5 min.	Continuous	HR $>$ 20% of baseline or when HR $<$ 10% of baseline for the next 5 beats	1	5	45s
Rest	3 min.	Continuous	After 2 min. and 30s	0	1	N/A
Walking around	10 min.	Continuous	HR $>$ 20% of baseline or when HR $<$ 10% of baseline for the next 5 beats.	3	7	45s
Step test	6 min.	Continuous	HR $>$ 30% of baseline for the next 5 beats.	1	3	1 min.
Sitting reading	10 min.	Continuous	Random	3	3	1 min.

TABLE I
THE TEST SCHEME.

1) What were you doing when the phone rang? 1 = Walking 2 = Standing 3 = Sitting down 4 = Lying down 5 = Sleeping 6 = Running around
2) How stress do you feel? 1 = not at all 2 = Very slightly 3 = A little 4 = Moderately 5 = Quite a bit 6 = Extremely
3) How much energy do you have? 1 = not at all 2 = Very slightly 3 = A little 4 = Moderately 5 = Quite a bit 6 = Extremely
4) Are you moody feeling now? 1 = not at all 2 = Very slightly 3 = A little 4 = Moderately 5 = Quite a bit 6 = Extremely
5) Are you impatient? 1 = not at all 2 = Very slightly 3 = A little 4 = Moderately 5 = Quite a bit 6 = Extremely
6) Are you in control of things right now? 1 = not at all 2 = Very slightly 3 = A little 4 = Moderately 5 = Quite a bit 6 = Extremely

TABLE II
AN EXAMPLE OF QUESTIONNAIRE.

up the time range to one hour with proper criteria specified for this activity. If the person is working from 8:30 am to 12:30 am, we would adjust different stressor examination criteria to control the questionnaire during this time. As a result, people could easily schedule daily activities at the beginning of each day. Moreover, at the beginning of the test, domain experts could design different questionnaires to increase the system's flexibility and ensure the collection of more diverse information. Hence, health care professionals can analyze the stress accumulated with more options and from different angles.

We recruited 10 people with age ranging from 20 to 30 years old in our pilot study. They were instructed on how to use the system, particularly on how to manipulate the N95 to answer the popped out questions. The whole experiment for each participant lasted about 40 to 50 minutes.

Figure 7 and 8 demonstrate the collected data from the remote side. We have classified six different activities into three categories. As the figures show, the X-Axis represents the time elapsed and the Y-Axis represents both the heart beat rate and the answers to the questions. The red dots on the figure denote the heart beat monitored during the whole process. We use different colors and symbols to indicate different answers to the six questions. We are trying to establish a relationship between the heart beat rate and the actual feeling of the person. The results show that our system, built on top of $SPARTAN$, is able to perform the normal functions of BASN correctly.

In order to evaluate the data compression module, we implemented two algorithms. The data compression ratios for Delta encoding and Huffman encoding are demonstrated in 9 and 10. In general, the Huffman data compression has less compression ratio than the Delta encoding. However, Huffman algorithm has more computation demand than Delta algorithm does. Benefited from our $SPARTAN$ framework, switching between the two algorithms only needs modification of one line of code.

After we demonstrate the effectiveness of the framework, we will discuss its efficiency. The total lines of code we

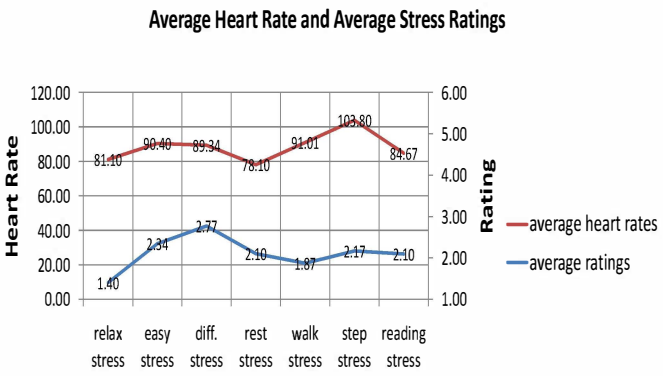

Fig. 7. The average heart beat rate and average feedback rating.

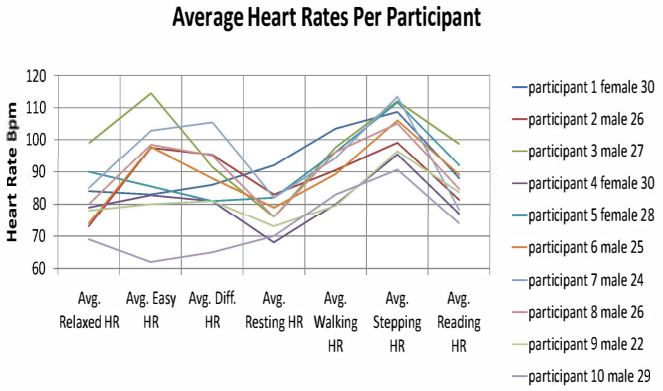

Fig. 8. The average heart beat rate of the participant.

deployed in the case study is shown in Table III. The size of the code is kept as small as possible. It is obvious that using $SPARTAN$ only add a tiny portion of code in the system. The reason is that almost half of the code becomes part of the four modules, which increased the code reusability for BASN applications.

In order to emphasize the flexibility, we change the sensor from Nonin 4100 to Zephyr HxM. We try to compare the

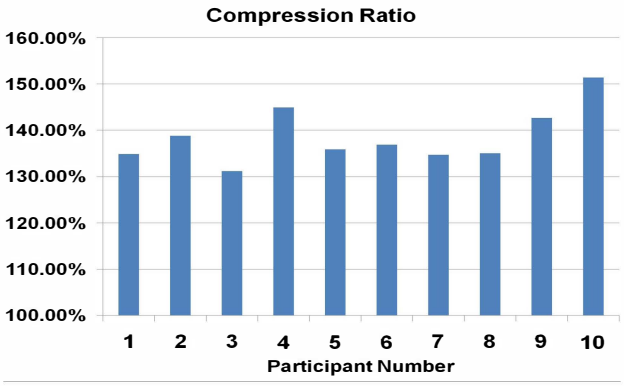

Fig. 9. The compression ratio of the delta algorithm.

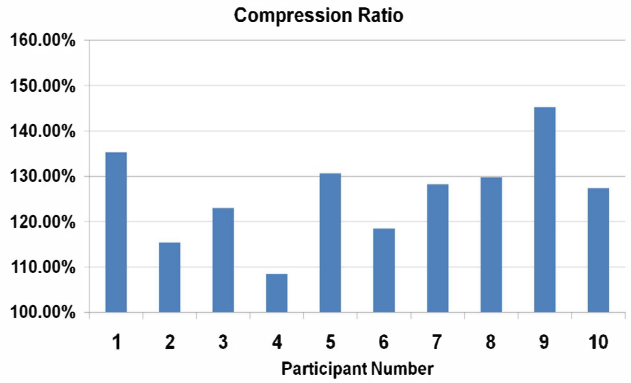

Fig. 10. The compression ratio of the huffman algorithm.

Line of Code (LoC)		
Original Code	Using $SPARTAN$	$SPARTAN$
1761	1793	884

TABLE III
THE TOTAL LINES OF CODE IN THE CASE STUDY.

lines of code needed to apply this change. The results are summarized in Table IV. Replacing the Nonin 4100 to another one, it takes adding or changing approximate 131 lines of code. Whereas only 24 lines of code need to be changed if we apply the $SPARTAN$ framework. On the other hand, if we use both sensors, the similar results can be obtained. The results show that by adding the modules in the framework, we can somehow increase the portion of code can be reused.

We collected feedback from 10 different participants in the experiments. There are some interesting observation

- People are willing to join in such a program if it is proved to be helpful. After all, the only device they have to purchase is a smartphone, which is used on a daily basis. In addition, the sensors will most likely become cheaper in the future. The one used in this work cost two hundred dollars.
- The sensor attached to the finger is inconvenient. Changing the entire program because of a sensor is unrealistic in this situation.
- Some of the participants complained about the user interface. The questionnaire system needs to be simplified by using only one button to select items rather than two.

	Replacing the sensor		Adding one sensor	
	Without SPARTAN	Using SPARTAN	Without SPARTAN	Using SPARTAN
Code Changed	131	24	175	31
Time takes for a graduate student	0.5∼1 hour	5∼10 min	1∼0.5 hour	15∼30 min

TABLE IV
THE COMPARISON OF HUMAN EFFORTS.

As we previously argued, the user interface needs to be optimized to fit the pace of the modern life. Moreover, creating a friendly user interface will involve various art design issues. This issue is beyond the framework we proposed tough.

- Being a prototype study, we set up too many questionnaires during the test. The number of them will be reduced in the future study. Based on our framework, since it is a very simple feedback mechanism, to modify the feedbacks can be as easy as modify several parameters in the *OC*.

V. Challenges and Opportunities

Although our goal is to simplify the development of BASN systems, there are other problems if we need to leverage the overall performance of the system. We list several challenges need to be considered to evaluate every system in the future.

A. Scalability

To achieve the concept of mass computing, the first issue to be solved is scalability. This issue is not separated from other computer system design principles. Naming, for example, represents the most common difficulty encountered in building large scale systems. Internet solved this problem with IP protocol. A mechanism is needed to distinguish each user so that the information can be delivered to the correct individual. Though current research has not reached the point where this problem becomes an urgent obstacle to the whole process, scalability related issues will become vital to ensure such systems finally benefit the entire society.

The concept of scalability, on the other hand, implies that the computer system be scalable so that heterogeneous components will fit in the existing scope. For example, different sensors the users wear characterized with different sampling rate, control command, and communication methods, have to be synchronized. New components are supposed to be easily added into the system.

B. Energy Consumption

One of the most significant design issues is creating sustainable computing. Most of the projects mentioned in Section II require continuous monitoring. Additionally, unless the systems designed are sustainable, they are less different from the traditional approaches. Conflicts arise when we utilize mobile devices, including sensors and mobile phones, that are only supported by batteries. Energy consumption must be considered in the system design at every level. However, the long-term monitoring is an energy consuming process that requires a constant communication channel and computation resources. It is not a good choice to leave the energy issue to the electronic engineers since the energy consumption of a particular system is unaccessible to them. As a result, the system must be energy aware. From the beginning, it must consume minimal energy to lengthen the lifetime of mobile devices. In addition, energy adaptiveness forces the system to utilize alternate strategies while dealing with differing levels of remaining power.

C. User Interface

When facing mass computing, we focus on the fact that any person can be a potential user of the system. Making friendly and straightforward design of the user interface will attract more potential users. The goal is to hide the complexity from the end users so that the operation can be done with little effort and simple instruction. In reality, however, human-machine interaction is generally a big issue. Since the system is developed for long-term monitoring, inconvenience and awkward operations create obstacles. A simple and effective user interface increases the chances of more people joining the project. Additionally, people utilizing the system may operate it at work, in a vehicle, or even in an unpleasant environment. A novel user interface will make the participatory sensing a plus of daily life rather than a burden for the end users.

VI. Summary and Future Work

In summary, we developed a framework, $SPARTAN$, to facilitate the process of developing BASN systems. The framework also provides flexibility and evolutionarily features to the system. We evaluated the framework by design a smartphone assisted stressor examination and warning system on top of the framework. The prototype attempts to use heart beat rate to evaluate the amount of stress experienced. We investigated the source of the stress by sending a feedbacks according to the different criteria. The system was tested for approximately 40 minutes on each of 10 participants. The results demonstrate the effectiveness of our framework that the results fit the needs from the phycologists very well and they are trying to find a new way to understand continuous stress. The framework is wrapped into libraries and will be open to the public at MIST lab, Wayne State University. We are currently working on another Ashtma monitoring system for Detroit Children project using SPARTAN.

In the future, we plan to develop more modules that can be combined into the framework, such as the storage module. Since the requirements for BASN change all the time and different applications have various demands, it would be meaningful to study the importance of the system components from the end user point of view. The results loom large when choosing modules for a particular system.

References

[1] Milenkovic A., C. Otto, and E. Jovanov. "Wireless Sensor Networks for Personal Health Monitoring: Issues and an Implementation". In: *Computer Communications* 29 (2006), pp. 2521–2533.

[2] Carl A. Gunter et al. *jXBee Java XBee 802.15.4/ZigBee Modem Driver*. 2009. URL: http://seclab.uiuc.edu/web/component/content/article/73-jxbee-java-xbee-802154zigbee-modem-driver-.html.

[3] JeongGil Ko et al. "Demo Abstract: MEDiSN: Medical Emergency Detection in Sensor Networks". In: *Proceedings of ACM Sensys 2008, November 2008*. 2008, pp. 361–362.

[4] P. Johnson et al. “Remote continuous physiological monitoring in the home”. In: *Journal of Telemed Telecare 2* (1996), 107113.

[5] Y.H. Nam et al. “Development of remote diagnosis system integrating digital telemetry for medicine”. In: *International Conference IEEE-EMBS*. 1998, pp. 1170–1173.

[6] Saad Arrabi and John Lach. “Adaptive Lossless Compression in Wireless Body Sensor Networks”. In: *Proceedings of the Fourth International Conference on Body Area Networks*. 2009.

[7] Kenneth Barr and Krste Asanovic. “Energy Aware Lossless Data Compression”. In: *Proceedings of the First International Conference on Mobile Systems, Applications, and Services (MobiSys 2003)*. San Francisco, California, May 2003, pp. 231–244.

[8] Adam T. Barth et al. “TEMPO 3.1: A Body Area Sensor Network Platform for Continuous Movement Assessment.” In: *Proceedings of the 2009 Sixth International Workshop on Wearable and Implantable Body Sensor Networks*. IEEE Computer Society, 2009, pp. 71–76.

[9] Andrew T. Campbell et al. “People-centric urban sensing”. In: *WICON’06: 2nd annual international workshop on Wireless Internet*. 2006.

[10] Wan-Young Chung et al. “A Cell Phone Based Health Monitoring System with Self Analysis Processor Using Wireless Sensor Network Technology”. In: *29th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*. 2007.

[11] Miikka Ermes et al. “Detection of Daily Activities and Sports With Wearable Sensors in Controlled and Uncontrolled Conditions”. In: *IEEE Transactions on Information Technology in Biomedicine* (2008), pp. 20–26.

[12] J. Ko, T. Gao, and A. Terzis. “Empirical Study of a Medical Sensor Application in an Urban Emergency Department”. In: *Proceeding of BodyNets 09: 4th Intl Conference on Body Area Networks*. 2009.

[13] Tae-Soo Lee, Joo-Hyun Hong, and Myeong-Chan Cho. “Biomedical Digital Assistant for Ubiquitous Healthcare”. In: *29th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*. 2007.

[14] B. Lo et al. “Body sensor network - a wireless sensor platform for pervasive healthcare monitoring”. In: *Proceeding of The 3rd International Conference on Pervasive Computing*. May 2005.

[15] David Malan et al. “CodeBlue: An ad hoc sensor network infrastructure for emergency medical care”. In: *In International Workshop on Wearable and Implantable Body Sensor Networks*. 2004.

[16] Maria G. et al. Martini. “Context-aware Multi-lead ECG Compression Based on Standard Image Codecs”. In: *Proceedings of the 3rd International Conference on Pervasive Computing Technologies for Healthcare, 2009. PervasiveHealth 2009*. Apr. 2009.

[17] Ar Milenkovic’, Chris Otto, and Emil Jovanov. “Wireless sensor networks for personal health monitoring: Issues and an implementation”. In: *Computer Communications (Special issue: Wireless Sensor Networks: Performance, Reliability, Security, and Beyond* 29 (2006), pp. 2521–2533.

[18] A. Natarajan et al. “Investigating Network Architectures for Body Sensor Networks”. In: *Proceeding of BodyNets 07: 2th Intl Conference on Body Area Networks*. 2007.

[19] Hidekuni Ogawa et al. “A Web-based Home Welfare and Care Services Request System Using a Pen Type Image Sensor”. In: *Consumer Communications and Networking Conference*. 2004.

[20] J.M. Quero et al. “Health Care Applications Based on Mobile Phone Centric Smart Sensor Network”. In: *29th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*. 2007.

[21] R. Shah, L. Nachman, and C-Y. Wan. “On the performance of Bluetooth and IEEE 802.15.4 radios in a body area network”. In: *Proceeding of BodyNets 08: 3rd Intl Conference on Body Area Networks*. IEEE Computer Society, 2008.

[22] U. Varshney. “Pervasive healthcare and wireless health monitoring”. In: *Mob. Netw. Appl.* 12(2-3) (2007), pp. 113–127.

[23] A. Wood et al. *ALARM-NET: Wireless sensor networks for assisted-living and residential monitoring*. Tech. rep. 2006.

[24] L. Yan, L. Zhong, and N.K. Jha. “Energy comparison and optimization of wireless body-area network technologies”. In: *Proceeding of the 2nd International Conference on Body Area Networks*. 2007.