Energy Efficient Data Broadcasting in a Body Area Network Middleware for Medical Applications

Lokesh Kumar Soni
M.E. Student, Department of Computer Engineering, PVPIT/JSPM, Bavdhan, Pune, Maharashtra, India

Arati M. Dixit
Professor, Department of Technology, Savitribai Phule Pune University, Pune, Maharashtra, India

ABSTRACT

A Wireless body area network (WBAN) consists of communication technology, sensors technology, a middleware and end user applications has emerged as a technology targeted at monitoring physiological and ambient conditions surrounding human beings and animals. Use of middleware enhances the power of BAN which handles the dynamic changes very smoothly. It also provides scalability to the system. This paper outlines the implementation of an energy efficient routing of data at middleware for the WBAN using the methodology of dynamic binding of data at the time of broadcasting it. A middleware is presented which handles the communication protocol according to the needs of various applications connected to it. A simulated environment is constructed for communications, sensors, middleware and applications to demonstrate feasibility of the system.

General Terms

Data Broadcasting Algorithm.

Keywords

Body Area Network; Wireless Sensor Networks; Middleware; Health Care.

1. INTRODUCTION

A Body area network (BAN) technology under the umbrella of wireless sensor network (WSN) technology has emerges as a key focused area of researchers due to its immense direct benefits to the society. The general architecture of the WBAN consists of some wearable and implanted computing device called sensors, which in turn communicates with the central monitoring system like middleware node which is responsible for the data or the message transfer to the central server positioned at the health care unit or mobile applications. Sensors link the physical world with the digital world by capturing and revealing real world phenomena and middleware converting these into a form that can be processed, stored, and acted upon. The advancements in wireless communications have motivated the development of WSNs for low-cost and easy-deployable physical and environmental monitoring. Demand for better health care and concerns for the cost have triggered the generation and development of wearable medical sensors. During the last few years, there has been a significant increase in the number of wearable health monitoring devices, ranging from simple pulse monitors, portable hypertension monitors to sophisticated and implantable sensors. Wearable devices are a key technology in restructuring health care system toward a more proactive, affordable citizen-centered health care system. For health care application, wearable, medical sensors can be integrated into wireless nodes. Health care data which are collected automatically, continuously, and remotely by BANs can be stored for a long period of time. These data are beneficial to doctors and patients tracking and monitoring, drug administration, and early disease detection and prevention.

The main issue in the context of these sensor nodes implanted in the human body which forms the body area network is the limited availability of the energy [1]. The main constraint that haunts any BAN is the unavailability of continuous power supply to them [2]. Also the same case becomes complex for the nodes that are planted inside the human body. In BANs, because of the energy constrained of tiny sensor nodes, effective energy consumption is a challenge which results into network longevity as a major challenge. A BAN connects independent nodes by using a central processing unit, known as a middleware node. An effective middleware node deployment strategy can influence the network lifetime eminently. This leads toward effective node placement and effective data handling at middleware level to protect middleware node energy through valuable management of sensor nodes, within a BAN. In this paper our main focus is on middleware data handling capabilities and its way to achieve energy efficient data broadcasting.

The rest of this paper is organized as follows. Section II contains some of the related work in middleware domain. The proposed middleware architecture with energy calculation concept is described in section III. Section IV talk about the simulated graphical user interface (GUI) for sensors, middleware and applications. A simulated environment with control and data flow result is reported in section V.

2. RELATED WORK

A number of middleware have been proposed so far. Development of Middleware is the promising and growing field of sensor networks and it facilitate the programmer task and bridge the gap between the applications and the hardware.

MyHealthAssistant[3] is an event-driven middleware architecture for multiple medical applications on Smartphone’s. MiLAN[4] works on priority of sensors, allows flexible combination of applications and sensors. Self-managed Cell (SMC)[5] proposed a policy-based body sensor networks (BSNs) middleware. The middleware platform COSMOS[6] (Common System for Middleware of Sensor Network) as a national project in Korea, which provides integrated data processing over multiple heterogeneous sensor networks based on sensor network abstraction called the sensor network common interface. The BiSNET[7] (Biologically-inspired architecture for Sensor NETworks) is a middleware architecture that addresses several key issues in multi-modal wireless sensor networks (MWSNs) such as autonomy, scalability, adaptability, self-healing and simplicity. Based on the observation that various biological systems have developed mechanisms to overcome these issues, BiSNET follows certain biological principles such as decentralization, food gathering/storage and natural selection to design MWSN applications. The OASiS[8] is a lightweight service-oriented architecture for sensor networks, which provides dynamic service discovery and can be used to develop ambient-aware applications that adapt to changes in the network and the environment. An important advantage of OASiS is that it allows seamless integration with Web services. SINA (System Information Networking Architecture) [9] is cluster-based middleware, and its kernel is based on a database. Mires [10] propose distributed systems middleware based on message orientation. It provides asynchronous communication model, which are event driven in most cases, and has more advantages over the traditional request-reply model. Impala [11] middleware considers the application itself exploiting mobile code techniques to change the functionality of the middleware executing at a remote sensor. The key to energy efficiency for Impala is for the sensor node applications to be as modular as possible, enabling small updates that require little power during transmission. Middleware openAAL [12] is flexible component based architecture, which shows different behavior based on installed components.

3. PROPOSED MIDDLEWARE ARCHITECTURE

A middleware for sensor networks decouples the application from the underlying sensing and communication tasks. This abstraction from the sensor network reduces the complexity for developers which accelerate the application development and, thus, the deployment of the system. Furthermore, a layered middleware architecture increases the system's flexibility and, hence, the adaptability to new sensors and circumstances. Hardware and protocols can be changed without touching the application itself. In addition to this, using a common middleware in a network allows running multiple applications using the same nodes. The proposed middleware consists of six major components, sensors and applications are shown in figure 1. It handles the communication protocol in energy efficient way and the sensors data rate as per user application requirement to provide dynamic needs of critical health monitoring. The event handler interprets incoming events, identifies general situations on which the system has to react and creates a corresponding derived event. The security Manager module provides security to the system data. The System Monitor measures the overall system status and detects critical situations such as a low battery level. The message handler handles the direct input/output data from sensors and applications. The basic idea behind sensor manager module is to, controlled communication between sensors and middleware interface. It controls data rate of sensors according to applications requirements. It makes the specific sensor sleep/active according sensor data requirement by active applications. Hence helps in energy efficiency of system through sensor management. The application manager module understands the requirement of different applications and takes intelligent decisions regarding: a) Sensors data broadcasting depending on number of applications. For example an application A1 is the only active application; require data from two sensors S1 and S2. It's always beneficial in this case to broadcast both the sensors data as a single message, as compare to broadcast two different messages. It reduces the load of data transfer; hence optimize power of the system. Similarly the optimization of the above trade-off between single stream and multi-stream broadcast of data can also be addressed based upon power efficiency and processing level complexity. b) Sleep/Active state of a sensor, managed by 'Sensor Manager' according to Sleep/Active state of respective applications.

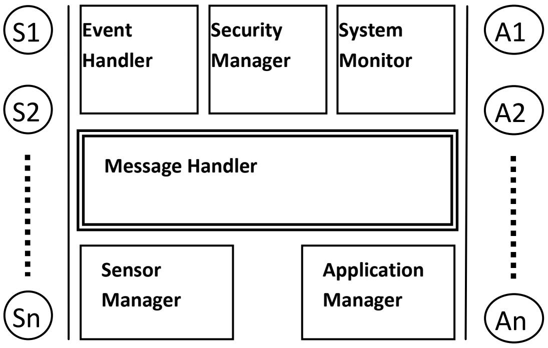


Fig 1: Architecture

In order to process k number of bits over a distance D with path loss coefficient $\boldsymbol{\eta}$, a radio spends energy during transmission is $\left(E_{T_x}(nJ)\right)$ [13].

$$E_{T_x}(k, d, \eta) = E_{T_{x\_elec}} * k + E_{amp}(\eta) * k * D^{\eta} \qquad \text{…….. (1)}$$

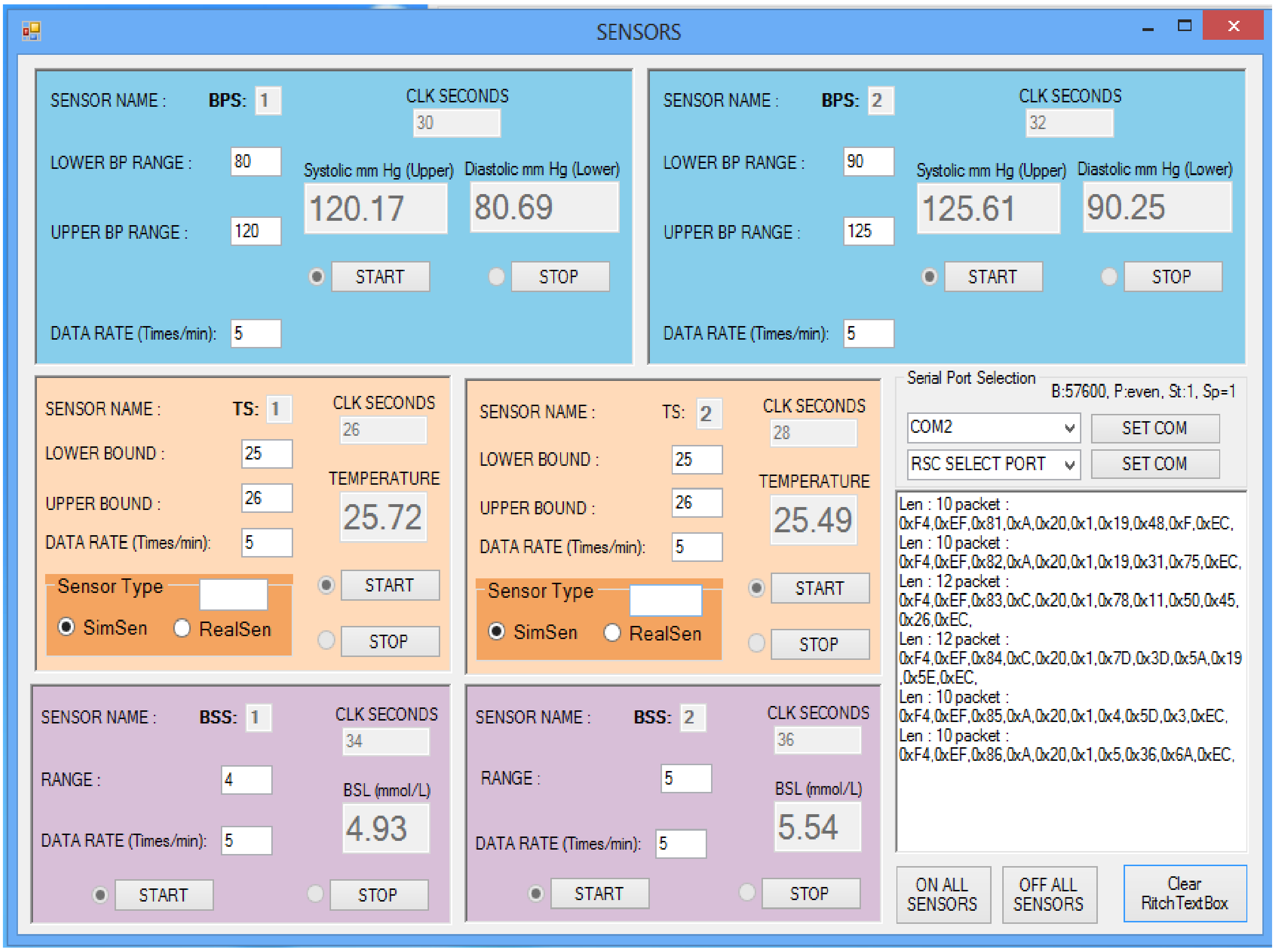
Here, $E_{T_{x\_elec}}$ is the energy dissipated by the radio to run the circuitry of the transmitter and $E_{amp}(\eta)$ is the energy dissipated by the transmit amplifier. Now except transmission bits (k), if other parameters are constant then (1) becomes

$$E_{T_x} \boldsymbol{\alpha} \ k \qquad \text{……….(2)}$$

Hence the energy consumption by an middleware is directly proportion to number of bits transmitted by it.

4. SIMUULATED GUI

Now refer simulated GUI of sensors in figure 2, middleware in figure 3 and applications in figure 4 to discuss the solution map of problem. All the three GUI's are works on settings; Baudrate: 57600, Parity: even, StartBit: 1 and StopBit: 1 with respective communication port connections. In sensors simulation GUI figure 2, there are six sensors namely two of each type; i.e. blood pressure sensor (BPS), temperature sensor (TS) and blood sugar sensor (BSS). Both the TS sensors have provision to connect and take real sensor data as per protocol of real data exchange. All the sensors have three different data exchange rate as an option from 1/3/5 pulse per minute i.e. number of sensor data per minute.



SENSOR NAME : BPS: 1
CLK SECONDS 30
LOWER BP RANGE : 80
UPPER BP RANGE : 120
Systolic mm Hg (Upper) 120.17
Diastolic mm Hg (Lower) 80.69
DATA RATE (Times/min): 5

SENSOR NAME : BPS: 2
CLK SECONDS 32
LOWER BP RANGE : 90
UPPER BP RANGE : 125
Systolic mm Hg (Upper) 125.61
Diastolic mm Hg (Lower) 90.25
DATA RATE (Times/min): 5

SENSOR NAME : TS: 1
CLK SECONDS 26
LOWER BOUND : 25
UPPER BOUND : 26
DATA RATE (Times/min): 5
TEMPERATURE 25.72
Sensor Type ● SimSen ○ RealSen

SENSOR NAME : TS: 2
CLK SECONDS 28
LOWER BOUND : 25
UPPER BOUND : 26
DATA RATE (Times/min): 5
TEMPERATURE 25.49
Sensor Type ● SimSen ○ RealSen

SENSOR NAME : BSS: 1
CLK SECONDS 34
RANGE : 4
DATA RATE (Times/min): 5
BSL (mmol/L) 4.93

SENSOR NAME : BSS: 2
CLK SECONDS 36
RANGE : 5
DATA RATE (Times/min): 5
BSL (mmol/L) 5.54

Serial Port Selection
B:57600, P:even, St:1, Sp=1
COM2
RSC SELECT PORT

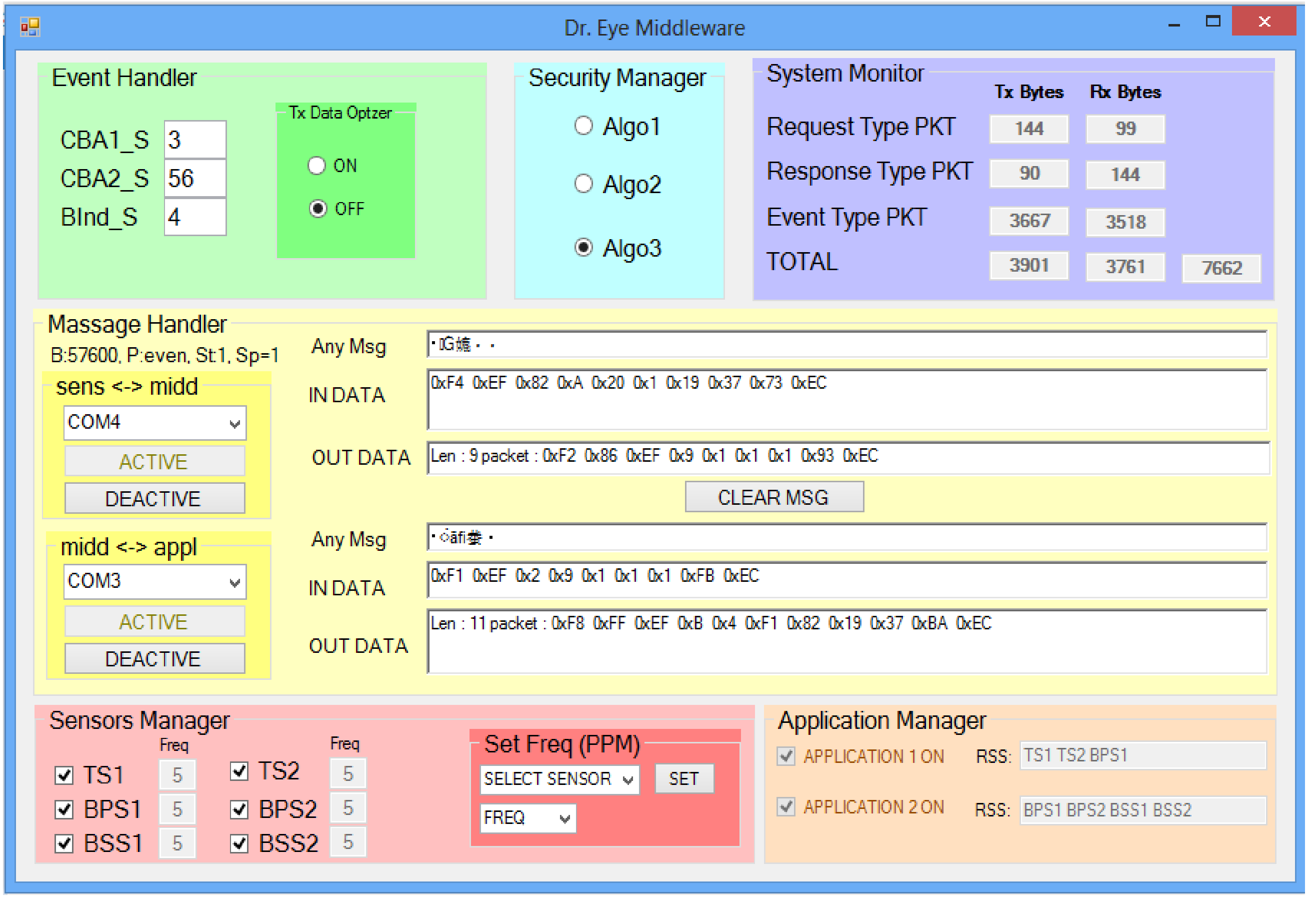
```
Len : 10 packet :
0xF4,0xEF,0x81,0xA,0x20,0x1,0x19,0x48,0xF,0xEC,
Len : 10 packet :
0xF4,0xEF,0x82,0xA,0x20,0x1,0x19,0x31,0x75,0xEC,
Len : 12 packet :
0xF4,0xEF,0x83,0xC,0x20,0x1,0x78,0x11,0x50,0x45,
0x26,0xEC,
Len : 12 packet :
0xF4,0xEF,0x84,0xC,0x20,0x1,0x7D,0x3D,0x5A,0x19
,0x5E,0xEC,
Len : 10 packet :
0xF4,0xEF,0x85,0xA,0x20,0x1,0x4,0x5D,0x3,0xEC,
Len : 10 packet :
0xF4,0xEF,0x86,0xA,0x20,0x1,0x5,0x36,0x6A,0xEC,
```

Fig 2: Simulated Sensors GUI

In figure 3 a simulated middleware GUI is presented with different components. Security components have three basic security options, Algo1 for secure data with XOR operation, Algo2 with NOR operation and Algo3 without any protection. Sensor manager in this GUI use to control different sensors in sensor simulated GUI based on operational status and their operational frequencies. Application manager here maintain the current working applications set and their required respective sensors set. Its Massage handler checks in/out data bundle and take effective decision based on defined strategies. System monitor maintain the records of total bytes received and transmitted from the middleware using different packet format types. Here event handler plays a major roll which generates derived events to optimize the broadcast data. Application simulator in figure 4, have two applications which can be activated with chosen selected sensor set by the users. A user of application can select dynamic rate of particular sensor as per need. All the three GUI's are connected with each other using PC serial ports as shown in figure 5 of simulation system. Hence a complete simulated environment generated to analyze the optimized data bytes broadcasting at middleware level. The results and optimization method are described in next section.

5. SIMULATION ENVIRUNMENT AND RESULT

The above maintained GUI's are developed using VB.NET on Windows 7 platform. Interconnectivity between the GUI's are provided through PC USB ports with the help of USB to RS232 converter. The results of data optimization for energy conservation are analyzed on the basis of three different factors; a) Number of running shared sensors between two applications, b) Effect of sensor data rate and c) Effect with respect to time. In designed communication protocol for the simulations, address of each sensor is of single byte and their actual reading data is of two bytes for TS & BSS and four bytes for BPS. Each data message has extra eight bytes to support communication over transmission.



Event Handler

CBA1_S 3
CBA2_S 56
BInd_S 4

Tx Data Optzer: ○ ON ● OFF

Security Manager: ○ Algo1 ○ Algo2 ● Algo3

System Monitor

	Tx Bytes	Rx Bytes	
Request Type PKT	144	99	
Response Type PKT	90	144	
Event Type PKT	3667	3518	
TOTAL	3901	3761	7662

Massage Handler
B:57600, P:even, St:1, Sp=1

sens <-> midd: COM4
Any Msg: ·ŪG媸 · ·
IN DATA: 0xF4 0xEF 0x82 0xA 0x20 0x1 0x19 0x37 0x73 0xEC
OUT DATA: Len : 9 packet : 0xF2 0x86 0xEF 0x9 0x1 0x1 0x1 0x93 0xEC

midd <-> appl: COM3
Any Msg: · ◌̈āfi㚇 ·
IN DATA: 0xF1 0xEF 0x2 0x9 0x1 0x1 0x1 0xFB 0xEC
OUT DATA: Len : 11 packet : 0xF8 0xFF 0xEF 0xB 0x4 0xF1 0x82 0x19 0x37 0xBA 0xEC

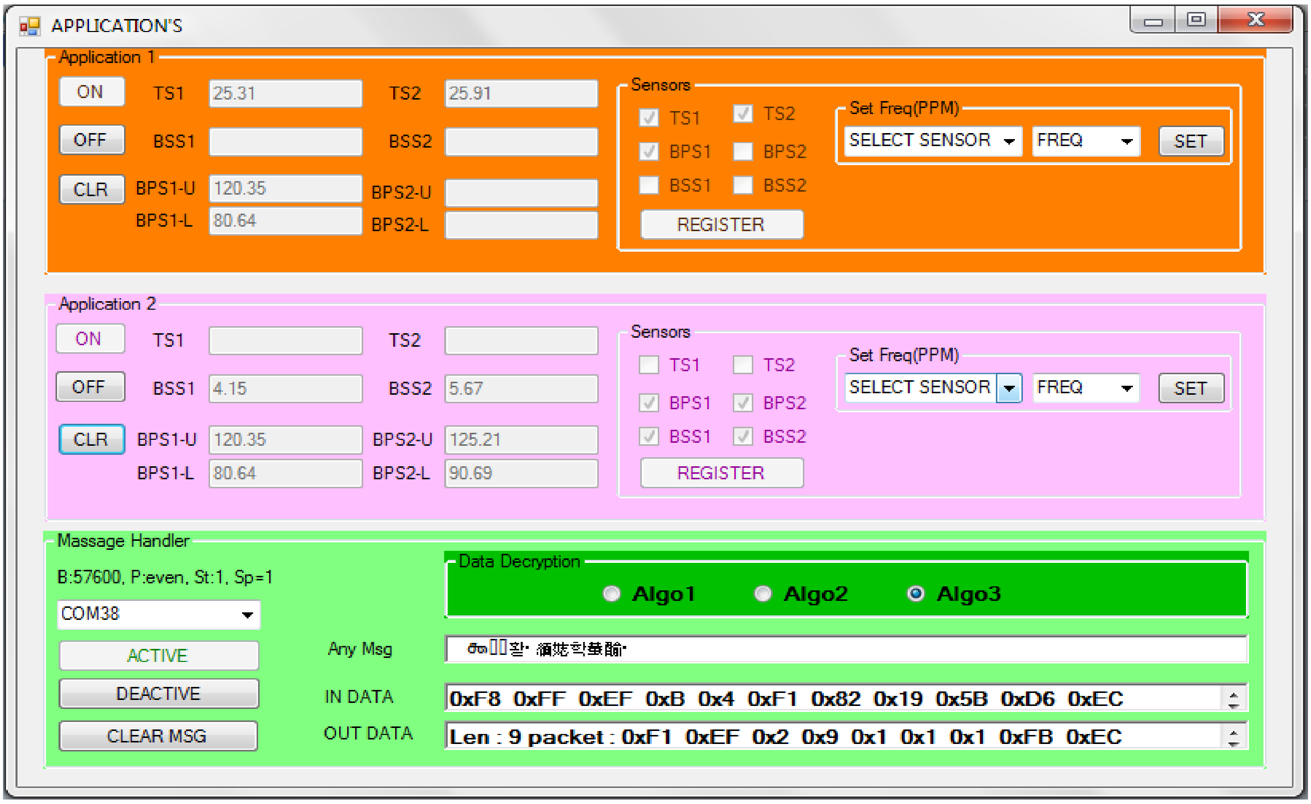
Sensors Manager

	Freq		Freq
☑ TS1	5	☑ TS2	5
☑ BPS1	5	☑ BPS2	5
☑ BSS1	5	☑ BSS2	5

Application Manager

☑ APPLICATION 1 ON RSS: TS1 TS2 BPS1
☑ APPLICATION 2 ON RSS: BPS1 BPS2 BSS1 BSS2

Fig 3: Simulated middleware GUI



Application 1

TS1	25.31	TS2	25.91
BSS1		BSS2	
BPS1-U	120.35	BPS2-U	
BPS1-L	80.64	BPS2-L	

Sensors: ☑ TS1 ☑ TS2 ☑ BPS1 □ BPS2 □ BSS1 □ BSS2

Application 2

TS1		TS2	
BSS1	4.15	BSS2	5.67
BPS1-U	120.35	BPS2-U	125.21
BPS1-L	80.64	BPS2-L	90.69

Sensors: □ TS1 □ TS2 ☑ BPS1 ☑ BPS2 ☑ BSS1 ☑ BSS2

Massage Handler
B:57600, P:even, St:1, Sp=1
COM38

Data Decryption: ○ Algo1 ○ Algo2 ● Algo3

Any Msg: ፊ๛▯▯잘· 緬妩확華餔·
IN DATA: 0xF8 0xFF 0xEF 0xB 0x4 0xF1 0x82 0x19 0x5B 0xD6 0xEC
OUT DATA: Len : 9 packet : 0xF1 0xEF 0x2 0x9 0x1 0x1 0x1 0xFB 0xEC

Fig 4: Simulated application GUI

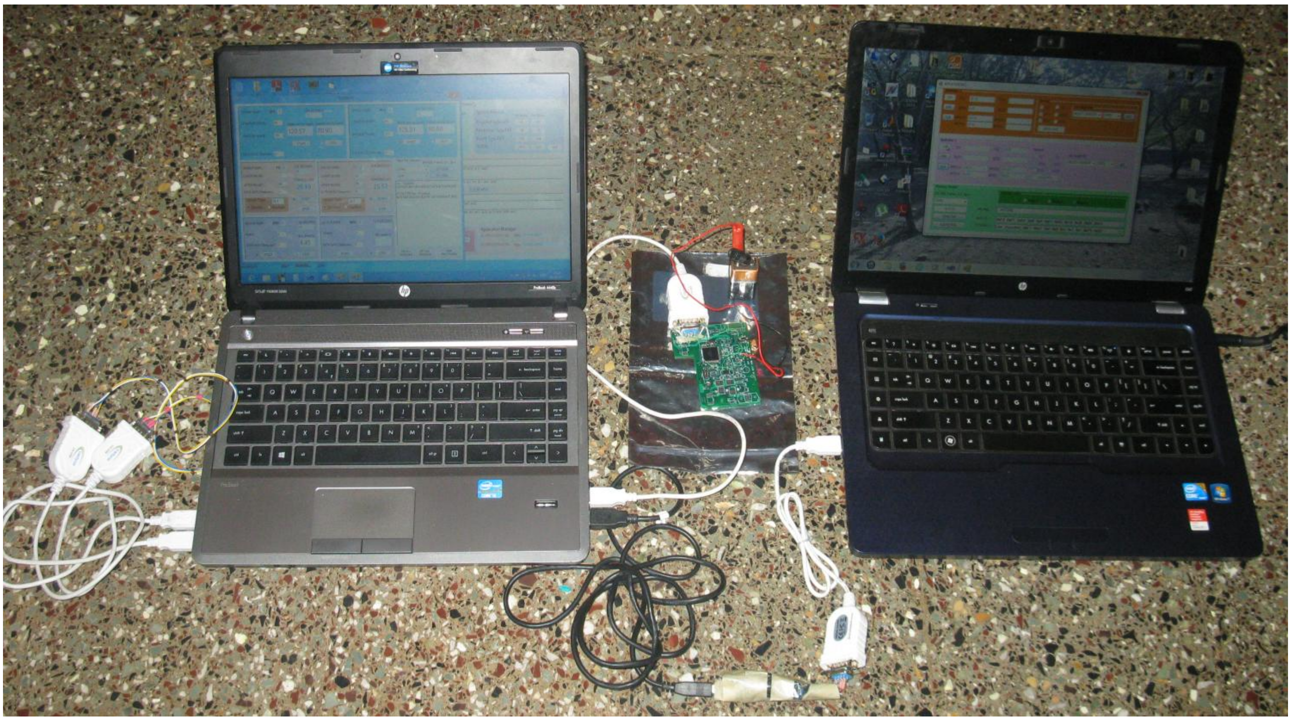


Fig 5: Simulation System

Hence a TS & BSS data message has eleven bytes and a BPS data message has thirteen bytes in their communication packets. If a packet has two TS data then it has fourteen bytes in total i.e. 8+3+3. In figure 6, 7 and 8; the total bytes transmitted in direct and optimized broadcast are compared using above defined factors. The total bytes sends by the middleware in direct broadcast is equal to the total number of each type of single sensor broadcast as per requirement. A shared sensor means it is used by more than one application. In optimized broadcast all the non shared sensors of an application are combine to form a single packet to reduce the total number of bytes to transfer. In figure 6, middleware controls all the six sensors data as per requirements of two applications when there are 0 to 5 common sensors between them. Their plotted chart shows statistics of conclusion when each sensor works on one Pulse Per Min. (PPM) for one hour duration. Maximum difference between direct and optimized broadcast plotted lines means more number of data saving which leads towards less number of bytes to transmit in optimized broadcast, hence less energy consumption by middleware to transmit same data bytes in optimized case. As number of shared sensors increases, the optimization level is reduced but steal it is better or equal to direct broadcast energy efficiency of middleware. On the same lines, as the PPM rate increases for the sensors, more & more bytes are transmitted and which in turned more & more energy conservation by the optimized broadcast w.r.t. direct broadcast as shown in figure 7. Similarly as the time of operation of the system increases the benefits of data binding visualized the perfect energy efficient system as shown in figure 8.

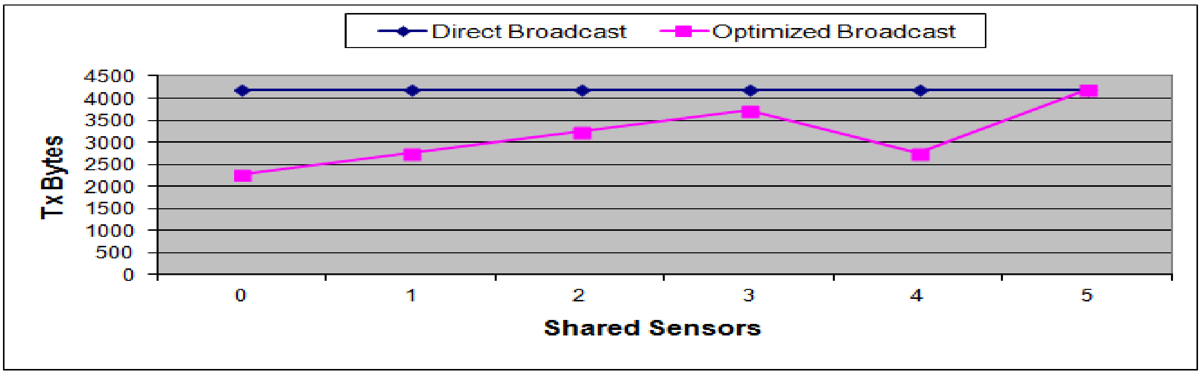


Fig 6: Optimization w.r.t. shared sensors between applications

Figure 9 shows a single simulated solution to visualize the concept in broader way. The extensive simulation consists of all the three parts sensors, middleware and applications into a single GUI. There are seven different types of sensors, each has eight instances, and in total there are fifty-six sensors. These are activated as per user selection request through NUMBER combo-box in the GUI. Actual data length of each type of sensor can also selectable through drop down menu. The GUI provide maximum of eleven applications, which are selectable.

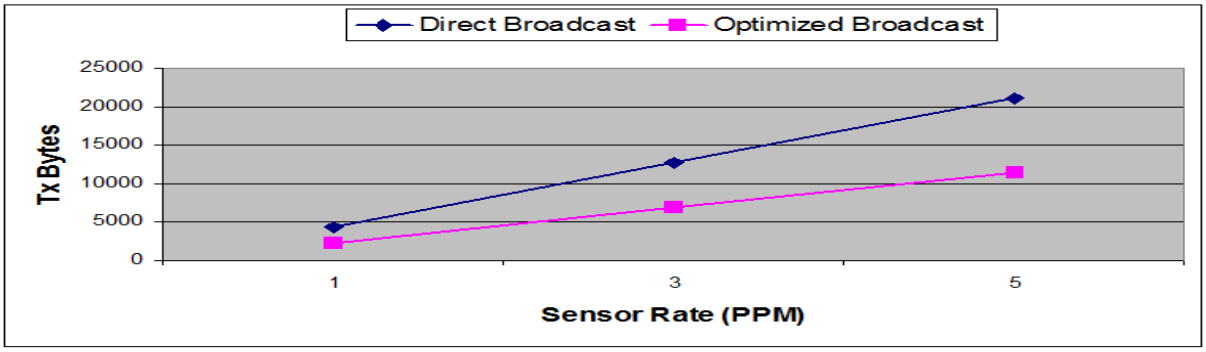


Fig 7: Optimization w.r.t. sensor data rate

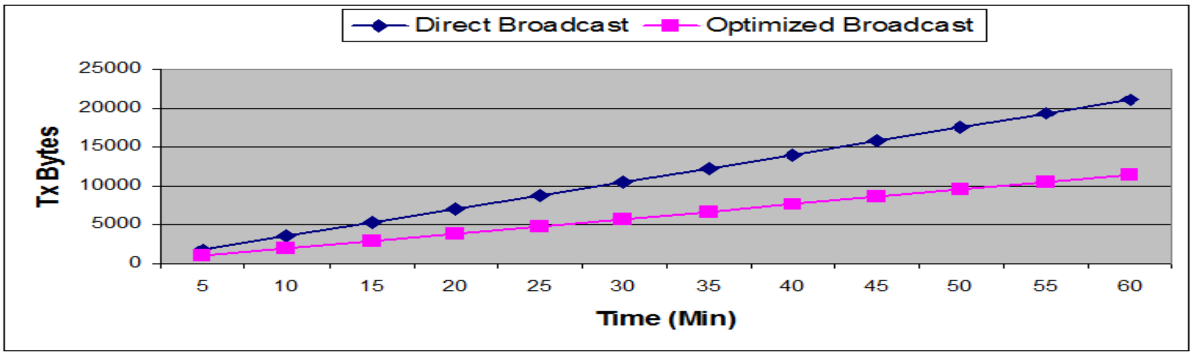
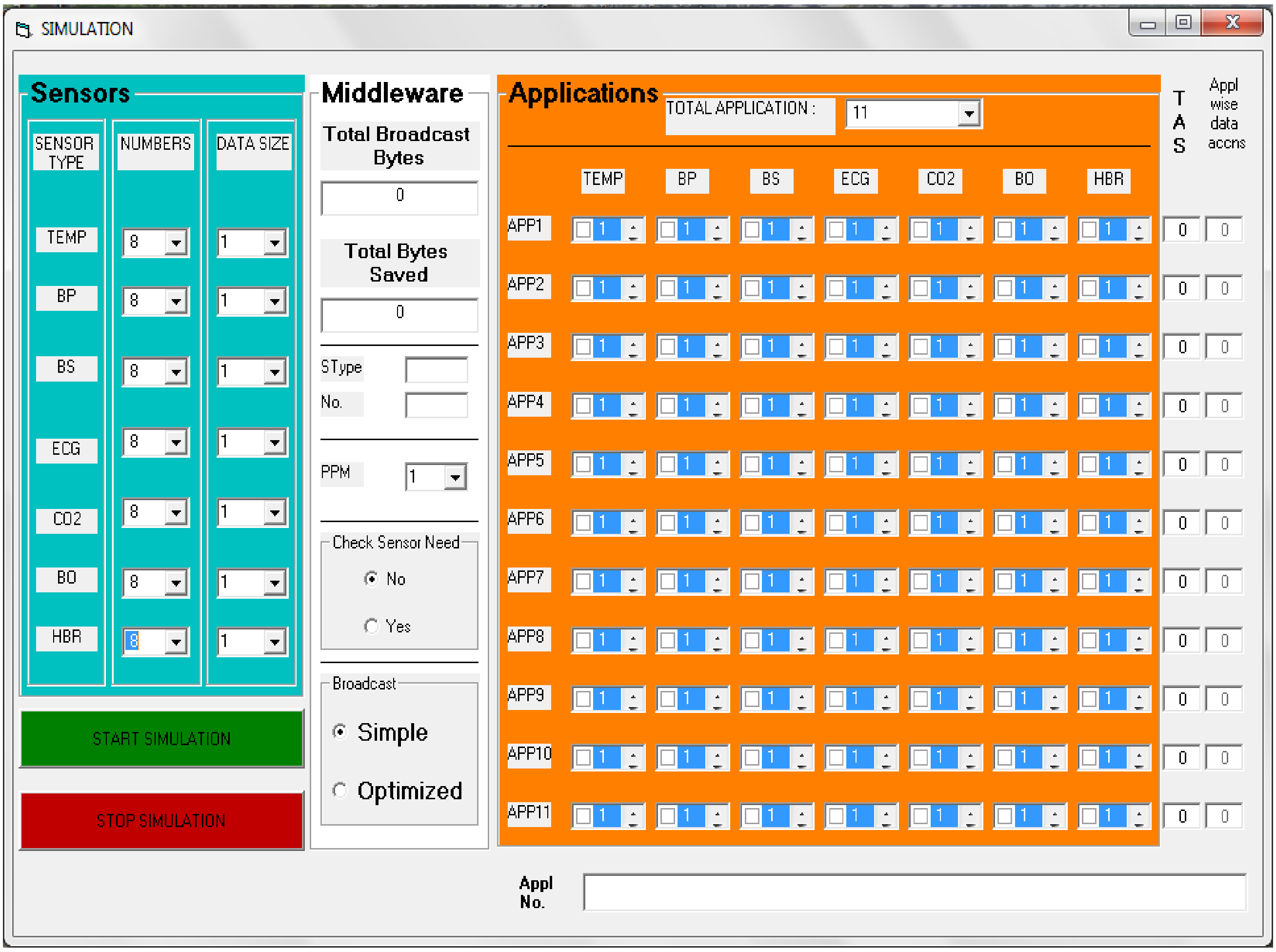


Fig 8: Optimization w.r.t. active time



SENSOR TYPE	NUMBERS	DATA SIZE
TEMP	8	1
BP	8	1
BS	8	1
ECG	8	1
CO2	8	1
BO	8	1
HBR	8	1

Total Broadcast Bytes: 0

Total Bytes Saved: 0

SType:

No.:

PPM: 1

Check Sensor Need: ● No ○ Yes

Broadcast: ● Simple ○ Optimized

TOTAL APPLICATION : 11

	TEMP	BP	BS	ECG	CO2	BO	HBR	TAS	Appl wise data accns
APP1	□ 1	□ 1	□ 1	□ 1	□ 1	□ 1	□ 1	0	0
APP2	□ 1	□ 1	□ 1	□ 1	□ 1	□ 1	□ 1	0	0
APP3	□ 1	□ 1	□ 1	□ 1	□ 1	□ 1	□ 1	0	0
APP4	□ 1	□ 1	□ 1	□ 1	□ 1	□ 1	□ 1	0	0
APP5	□ 1	□ 1	□ 1	□ 1	□ 1	□ 1	□ 1	0	0
APP6	□ 1	□ 1	□ 1	□ 1	□ 1	□ 1	□ 1	0	0
APP7	□ 1	□ 1	□ 1	□ 1	□ 1	□ 1	□ 1	0	0
APP8	□ 1	□ 1	□ 1	□ 1	□ 1	□ 1	□ 1	0	0
APP9	□ 1	□ 1	□ 1	□ 1	□ 1	□ 1	□ 1	0	0
APP10	□ 1	□ 1	□ 1	□ 1	□ 1	□ 1	□ 1	0	0
APP11	□ 1	□ 1	□ 1	□ 1	□ 1	□ 1	□ 1	0	0

Appl No.:

Fig 9: Single Simulation Solution

The middle portion of GUI acts as a middleware, which provide broadcasting option like Simple: a normal mode, no two or more sensor data accumulation into single packet and Optimized: depending upon sensor data usage into an application different sensor data is accumulated and transmitted through single packet as per defined policies. The middleware shows results in term of total broadcast bytes and total saved bytes. It also provides sensor data rate management and estimation of data broadcast based upon used-unused sensors. A simulated result is depicted in figure 10 and table 1, has eight instances of each type of seven sensors and eight applications to take these data. In the result different instances of particular sensor are used by each application. Start with single sensor different instances used by all the eight applications, then second sensor different instances used by all the eight applications and similarly the simulation grows up to seventh sensor. It founds that total saving on transmitted bytes increases as the number of sensors increases. Which directly contribute in energy saving. These results indicate an effective energy improvement of overall system.

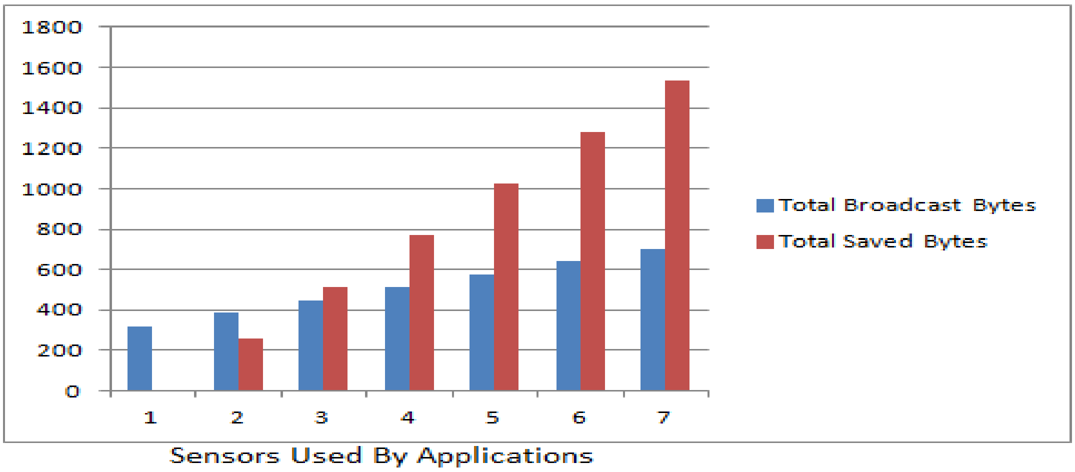
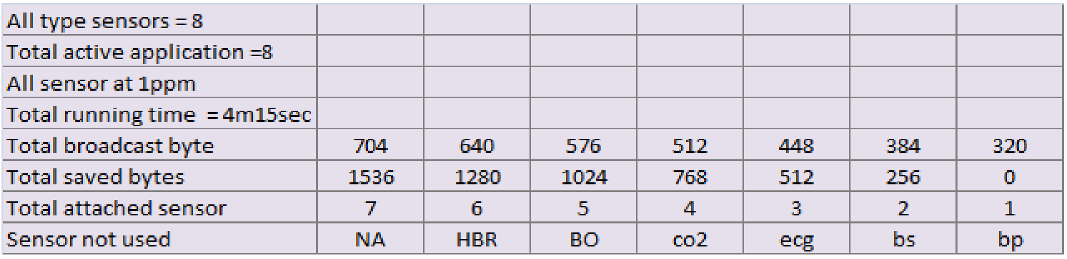


Fig 10: Single Simulation Solution Result

Table 1. Single Simulation solution quantitative Result



All type sensors = 8							
Total active application =8							
All sensor at 1ppm							
Total running time =4m15sec							
Total broadcast byte	704	640	576	512	448	384	320
Total saved bytes	1536	1280	1024	768	512	256	0
Total attached sensor	7	6	5	4	3	2	1
Sensor not used	NA	HBR	BO	co2	ecg	bs	bp

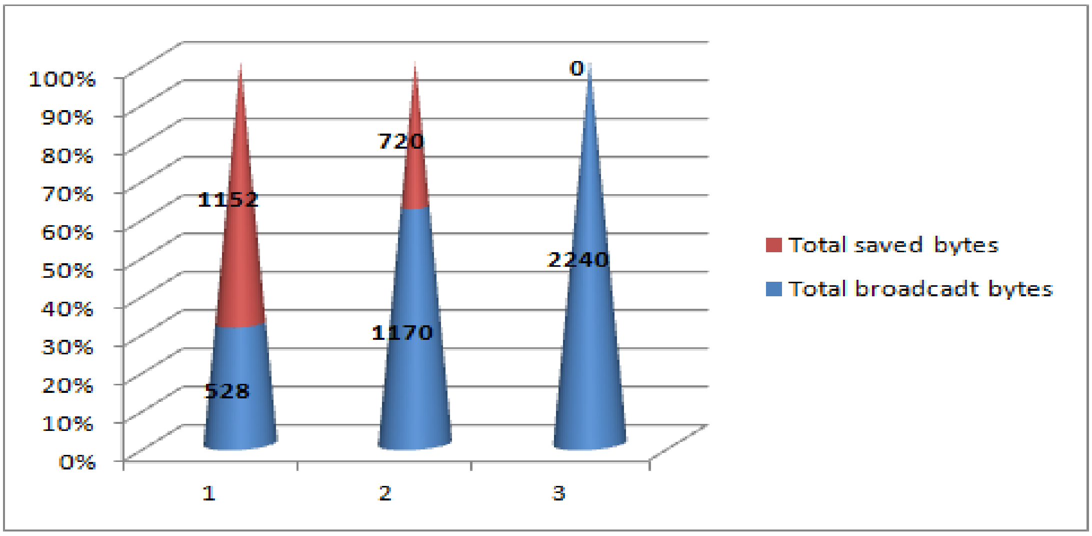


Fig 11: Best-case, Middle-case and Worst-case Comparative Analysis

The best-case, middle-case and worst-case of GUI are captured and found that in best-case the system total saved bytes are 200 percent of the total broadcast bytes. In other words the system becomes three-time energy efficient as compare to simple broadcast performance. The comparative analysis of the three cases is depicted in figure 11 with pillar 1 as best-case, pillar 2 as middle-case and pillar 3 as worst-case.

6. CONCLUSION

A BAN architecture simulation presented in this paper, integrates sensors, middleware and applications into a single complete system. The major focus of proposed system is on middleware, to handle the communication protocol in energy efficient way and to provide dynamic sensor data rate as per user application requirement to fulfill dynamic needs of critical health monitoring. The results shows that run time binding of data packets according to applications demand save the number of bytes to transfer and hence make the middleware energy efficient. In order to improve the complete system further, the future scope idea is to combine the presented system with energy efficient placement of sensors with respect to middleware node.

7. REFERENCES

[1] Chen, Min, Sergio Gonzalez, Athanasios Vasilakos, Huasong Cao, and Victor C. Leung. "Body area networks: A survey." Mobile networks and applications 16, no. 2 (2011): 171-193.

[2] Hanson, Mark A., Harry C. Powell Jr, Adam T. Barth, Kyle Ringgenberg, Benton H. Calhoun, James H. Aylor, and John Lach. "Body area sensor networks: Challenges and opportunities." Computer 1 (2009): 58-65.

[3] Seeger, Christian, Kristof Van Laerhoven, and Alejandro Buchmann. "MyHealthAssistant: An Event-driven Middleware for Multiple Medical Applications on a Smartphone-Mediated Body Sensor Network." Biomedical and Health Informatics, IEEE Journal of 19, no. 2 (2015): 752-760.

[4] Heinzelman, Wendi B., Amy L. Murphy, Hervaldo S. Carvalho, and Mark Perillo. "Middleware to support sensor network applications." Network, IEEE 18, no. 1 (2004): 6-14.

[5] Keoh, Sye Loong, Naranker Dulay, Emil Lupu, Kevin Twidle, Alberto E. Schaeffer-Filho, Morris Sloman, Steven Heeps, Stephen Strowes, and Joe Sventek. "Self-managed cell: A middleware for managing body-sensor networks." In Mobile and Ubiquitous Systems: Networking & Services, 2007. MobiQuitous 2007. Fourth Annual International Conference on, pp. 1-5. IEEE, 2007.

[6] Kim, Yung Bok, Marie Kim, and Yong Joon Lee. "COSMOS: a middleware platform for sensor networks and a u-healthcare service." In Proceedings of the 2008 ACM symposium on Applied computing, pp. 512-513. ACM, 2008.

[7] Boonma, Pruet, Paskorn Champrasert, and Junichi Suzuki. "Bisnet: A biologically-inspired architecture forwireless sensor networks." In Autonomic and Autonomous Systems, 2006. ICAS'06. 2006 International Conference on, pp. 54-54. IEEE, 2006.

[8] Koutsoukos, Xenofon, Manish Kushwaha, Isaac Amundson, Sandeep Neema, and Janos Sztipanovits. "Oasis: A service-oriented architecture for ambient-aware sensor networks." In Composition of Embedded Systems. Scientific and Industrial Issues, pp. 125-149. Springer Berlin Heidelberg, 2008.

[9] Shen, Chien-Chung, Chavalit Srisathapornphat, and Chaiporn Jaikaeo. "Sensor information networking architecture and applications." Personal communications, IEEE 8, no. 4 (2001): 52-59.

[10] Souto, Eduardo, Germano Guimarães, Glauco Vasconcelos, Mardoqueu Vieira, Nelson Rosa, and Carlos Ferraz. "A message-oriented middleware for sensor networks." In Proceedings of the 2nd workshop on Middleware for pervasive and ad-hoc computing, pp. 127-134. ACM, 2004.

[11] Liu, Ting, and Margaret Martonosi. "Impala: A middleware system for managing autonomic, parallel sensor systems." In ACM SIGPLAN Notices, vol. 38, no. 10, pp. 107-118. ACM, 2003.

[12] Wolf, Peter, Andreas Schmidt, Javier Parada Otte, Michael Klein, Sebastian Rollwage, Birgitta König-Ries, Torsten Dettborn, and Aygul Gabdulkhakova. "openAAL-the open source middleware for ambient-assisted living (AAL)." In AALIANCE conference, Malaga, Spain, pp. 1-5. 2010.

[13] Munasinghe, Kumudu S., and Abbas Jamalipour. "Body Node Coordinator Placement Algorithms for Wireless Body Area Networks." Internet of Things Journal, IEEE 2, no. 1 (2015): 94-102.