### Modeling and analysis of the gateway node in body sensor networks

M. P. Shopov*, G. V. Spasov* and G. I. Petrova**

* Technical University of Sofia / Department of Computer systems and Technologies, Plovdiv, Bulgaria
** Technical University of Sofia / Department of Electronics, Plovdiv, Bulgaria
mshopov@tu-plovdiv.bg

Abstract - The paper suggests a model for implementation of a body sensor network's gateway functions. The gateway is based on a mobile smartphone with a Linux kernel. Analysis on traffic management schemes based on different queue disciplines is presented in the paper. Since the gateway is modeled as a standard mobile device, the typical traffic flows for such devices are taken into account and their influence on the medical sensors data flow traversing through the gateway is considered. The paper presents the results from experimental analysis on performance parameters. Testbed experiments are conducted on a Nexus One smartphone platform and Linux traffic control tool.

### I. INTRODUCTION

Body sensor networks are one of the key components of the new and emerging personal healthcare systems. They are formed by low-power wireless sensor nodes placed on or around the human body. These networks are opening the way for new healthcare and wellness applications by giving the individual person a more central role in its treatment and prevention process, and by giving healthcare professionals an access to data, collected under natural activities and environment.

Body sensor networks (BSN) often utilize a star topology with a clearly separated master/slave roles. The master node (also known as personal monitor, personal server, BSN aggregator and etc. [1], [2], [3]) is a device characterized with higher processing and communication capabilities that acts as a coordinator for the network. It performs multiple functions including sensing, fusing data from multiple sensors, serving user interface, and bridging BSN to higher-level infrastructures [3], [4], [Ullah09]. There are two approaches for realization of the coordinator node. The first one is to use a custom designed device [5], [6] that will be more reliable and optimized for the BSN applications in both hardware and software terms. The second and most preferred approach however is to use off-the-shelf mobile device such as PDAs, smartphones, tablets, etc. [1], [2], [4]. This not only brings the benefits of rapid prototyping, small form factor, and continuously increasing processing and communication capabilities, but also reduces the cost for development and acquirement by the end-users since it is very common that they already own such a device. The disadvantage of the solution is that the BSN application will have to share hardware and software resources with the other applications running on the device. In this paper a standard smartphone device will be assumed for the role of a BSN coordinator.

### A. Motivation

Quality of service is a fundamental factor to achieve reliable and timely data delivery in healthcare applications. The primary QoS parameters are bandwidth, delay/latency, delay variation (jitter), packet loss and error rate. To meet the needs of healthcare applications without totally corrupting the performance of the other smartphone applications a flexible network management has to be applied. To control the latency and jitter parameters introduced by the BSN gateway an appropriate packet scheduling and queue management mechanisms should be used.

### B. Background

Priority queuing (PQ) is the most effective scheduler when there is a prioritized traffic, because it is able to forward this traffic as soon as possible. However, PQ is not able to guarantee a minimum share to lower priority traffic. Link-sharing mechanism explicitly enforced at the output interface of the gateway can prevent starvation of lower priority traffic, while still satisfying the needs of higher-priority ones. Requirements for link-sharing, taken together with requirements for real-time services, lead to a requirement for hierarchical link-sharing [7]. It allows control on the bandwidth distribution between various entities that are organized in classes in a tree structure. Hierarchical link-sharing and packet scheduling are involved in the same process of choosing the packet to be sent next on the output link and can be implemented together.

One of the first implementation of this approach is Class-based queuing (CBQ) mechanism first presented in [7]. The architecture of CBQ combines a link-sharing scheduler with general scheduler (Fig.1). Incoming traffic is classified into appropriate queues based on a set of rules. The general scheduler then extracts packets from those queues ensuring each class receives its nominal bandwidth, taking into account the assigned priorities of the classes. The estimator together with link-sharing scheduler is used to control the distribution of excess bandwidth [7].
Hierarchical Token Bucket (HTB) [8] is another implementation of the group of hierarchical link-sharing algorithms. HTB works similarly to CBQ, but unlike CBQ that uses calculated idle time periods to shape classes, HTB uses token bucket filter (TBF) algorithm. TBF is a simple queue discipline that only passes packets arriving at a rate not exceeding an administratively set limit, while still allowing short bursts. Each HTB class is associate with buffer constantly field with tokens. A packet is dequeued only when a token is available in the buffer. The size of the buffer defines the burst period. The advantage of TBF is that it does not depend on interface characteristics like bandwidth of the output interface [8].

The main elements of traffic control in Linux kernel are classification, scheduling and queuing. Linux traffic control combines queuing and scheduling into queuing disciplines. These queuing disciplines can be nested in each other. The traffic control in Linux works in the kernel space. To configure and control the Linux traffic control from user space a tool called tc is used.

The rest of the paper is organized as follows. Section II presents a model of BSN coordinator and traffic patterns. Section III describes the analysis on performance of traffic control schemes for BSN scenarios with test-bed experiments. Section IV shows some graphical representation of results from the analysis. Finally, section V gives some conclusive remarks and presents the future work directions.

II. BSN GATEWAY MODEL

A. System model

For the BSN gateway model, a standard smartphone device with Linux-based operating system is considered. The gateway is modeled as a set of queues and a traffic control logic (Fig.1). Ingress queues describe traffic sources and could be separated in three general groups. Body sensor network group describes packets coming from sensor network. Smartphone applications group describes traffic generated by applications running on the device, and System and tethering group describes system traffic generated from the operating system itself and from shared Internet connection.

B. Traffic model

BSN traffic is composed of multiple flows with periodic characteristics. An aggregation of these flows can be expressed by addition of the arrival functions [9]. Individual arrival function \( A_i(t) \) together with the resulting aggregated arrival function \( A_T(t) \) are expressed as:

\[
A_i(t) = \frac{L_i}{T_i}, \quad t \Rightarrow \\
A_T(t) = \sum_{i=1}^{n} A_i(t) = t \sum_{i=1}^{n} \frac{L_i}{T_i}. \quad (1)
\]

where \( L_i \) is the packet size and \( T_i \) is the interval between packets. Typical values of traffic generated by sensors of some of the most important physiological signals are summarized in Table 1. If we choose to select the minimal interval among all, than given the aggregated data rate from (2) the packet size will be given by:

\[
L_x = T_{\text{min}} \sum_{i=1}^{n} \frac{L_i}{T_i}. \quad (2)
\]

A study on smartphone applications traffic is presented in [10]. The extracted results indicates that more than 50% of the traffic is a web traffic. E-mail, media, and online maps corresponds to approximately 10% of the traffic each. Another considerable source of traffic not considered in the paper is the tethering or sharing Internet connection with other devices over wireless local area networks. The results for transferred packet sizes show an advantage of small data transfers (30% of transfers under 1K bytes) [10]. In [11] the same authors propose a usage of smartphones models. The authors proposed a mixture of exponential and Pareto distributions to model payload size [11]:

\[
r \cdot \text{Exp} \left[ \lambda \right] + (1-r) \cdot \text{Pareto} \left[ x_m, \alpha \right], \quad (3)
\]

where parameter \( r \) gives the relative mixture between two distributions, \( \lambda \) is the exponential rate, \( \alpha \) and \( x_m \) are Pareto shape and scale respectively.
Exponential distribution is used to model short user sessions, while Pareto distribution is proposed to covers long user sessions. According to [11] the screen timeout value is a best fit for the scale parameter $x_m$. The other three parameters are determined using EM algorithm. Since more than 50% of the traffic constitute Web traffic the inter departure time (IDT) will be best modeled using Weibull distribution [11], [12].

The tethering traffic constitutes the traffic routed from a shared Internet connection (Fig.2). Since this traffic is usually generated from one or more mobile devices locally connected to the BSN gateway, it is selected to be modeled as a multiply of the second group of traffic - smartphone applications. As the number of devices that use the shared connection increases, the aggregated traffic changes from Weibull distribution to exponential [12].

C. Evaluation scenarios

The processing and transmission delays are fairly constant while the access and buffering delays are highly time-variant due to variable occupancy levels of buffers. The total time delay $T_{\text{delay}}$ can be broken into three parts [13]: delay at the source node, on the network - $T_{\text{tx}}$, and at the destination node $T_{\text{tx}}$. Time delay at the source node includes the preprocessing time $T_{\text{pre}}$, and waiting time $T_{\text{wait}}$, which is the sum of the queue delay $T_{\text{queue}}$ and the blocking delay $T_{\text{block}}$:

$$T_{\text{delay}} = T_{\text{pre}} + T_{\text{wait}} + T_{\text{tx}} + T_{\text{post}}$$

$$T_{\text{wait}} = T_{\text{queue}} + T_{\text{block}}.$$  \hspace{1cm} (4)

The queue delay $T_{\text{queue}}$ is the time a message waits in the buffer at the source node while previous messages in the queue are being sent. The evaluation of the effect of different traffic control configurations on the queueing delay is one of the tasks in the presented analysis. Android kernel supports only a limited set of queue disciplines which determine the choice of scenarios to be evaluated.

The first scenario considers a FIFO queue management. This queue management discipline has no provisioning for QoS. All traffic flows are equally contending for bandwidth and no prioritization and reservation is assumed. This discipline is expected to give worst case results for the queue delay of BSN traffic.

The second scenario considers a strict priority queue management discipline where BSN traffic is configured with highest priority, stream video and VoIP traffic is given the next priority level and all other traffic is configured with the lowest priority level. This scenario is expected to give the best possible results for the queue delay of BSN traffic, but at the price of a worse queue delay for stream video and VoIP traffic.

The third scenario under evaluation considers a hierarchical link-sharing mechanism, where each traffic type is isolated in a separate class with bandwidth reservation and prioritization configured appropriately.

III. Experimental analysis

In order to evaluate how different queue management and packet scheduling mechanisms could improve the delay and jitter values of the time critical traffic types on a BSN gateway a number of experiments have been carried out using the experimental testbed depicted at Fig.3. All of the experiments are conducted on a smartphone device HTC Nexus One (development version). The device is characterized with the following parameters: Processor Qualcomm QSD8250 Snapdragon ARMv7 @1GHz, RAM 512MB, Linux kernel version 2.6.35, Android version 2.3.6.

Traffic control in Linux is the name given to a set of queuing systems and mechanisms by which packets are received and transmitted on a gateway node. The tc tool, part of iproute2 suite, is used to manipulate the traffic control settings in Linux kernel [14]. Its components include schedulers (qdisc), shapers (classes), filters, classifiers, policers, and handlers. In this paper we use tc to configure the evaluation scenarios described in the previous section. The queue disciplines used in the experiments are pfifo (packet-based FIFO queue), pfifo_fast (PF, provides three strict priority pfifo queues), HTB (hierarchical link-sharing queue discipline).

The performance evaluation study has been performed using D-ITG (Distributed Internet Traffic Generator [15]). D-ITG is a platform capable to produce traffic at packet level accurately replicating appropriate stochastic processes for both IDT (Inter Departure Time) and packet size random variables with a set of supported statistical distributions. We have then proceeded to obtain empirical data from the different scenarios to be processed and the model described in the previous section. Output sequences are derived from packet numbers, transmission and delivery times, available by use of D-ITG. Since both sender and receiver component of D-ITG are running on the same device (Fig.3) the block delay $T_{\text{block}}$ and transmission delay $T_{\text{tx}}$ (4) are not considered in results.

Several traffic flows are used in the experiments confirming to the calculations in previous section. Only four of them are considered interested and their performance parameters are being monitored. BSN traffic, VoIP traffic, stream video traffic and Web traffic. The first three of them are considered because of their requirement for QoS. The fourth traffic is selected because it compromise significant percent of the smartphone traffic and is used as a marker of how different traffic control schemes influence rest of the traffics. Tethering
traffic is presented in the experiments as additional three of each of other smartphone traffic flows, assuming that the connection is shared among three clients.

IV. Results

In this section we present and discuss the experimental results of the evaluation. The influence of the three queue management disciplines (FIFO, PQ, and HTB) on the delay and jitter parameters of the four selected traffic flows is shown on next four figures. Fig.4 shows the experimental results for BSN traffic. As expected, we observe lowest maximum and average delays and jitter when priority queue is used. When HTB is used these values are a little bit higher, but still in the same magnitude as in the case of PQ. The same tendency is observed for the VoIP traffic as shown on Fig.5. For the video traffic however (Fig.6), the use of HTB results in values for maximum and average delay closer to that of FIFO queue discipline. Finally, Fig.7 shows how web traffic is influenced by the different queue management disciplines. Since Web traffic goes in the lowest priority queue, it is not surprisingly that PQ results in highest maximum delay values. However, the average delay and jitter are almost identical for all three disciplines.

The distribution of delay and jitter of the packets from BSN traffic flow for the scenario with HTB queue discipline are given on Fig.8 and Fig.9 respectively. The
results show that most of the packets experience delays closer to average values, while only few packets experience higher delay and jitter. This peaks in the graphics can be explained with burst traffic from other flows scheduled for transmission.

V. CONCLUSION

In this paper, we investigated the use of standard smartphone device for mobile healthcare applications. In addition to its typical usage the role of a gateway for a set of wireless medical sensors is assumed. Its functions and traffic patterns are then modeled in the paper. To address the special need for bandwidth, delay and packet loss of traffic from sensors nodes several traffic control mechanisms are being discussed. An experimental analysis is then conducted to evaluate how this mechanisms influence the performance of the other smartphone applications if this new application is going to be accepted by users. The obtained results indicate that applying a flexible traffic control could not only guarantee the needs of BSN applications, but could also improve the performance of other delay sensible applications such as VoIP and stream video.

ACKNOWLEDGMENT

The presented work is supported by National Science Fund of Bulgaria project “Modern approaches in the design of clinical information system (CIS) for observation of pediatric surgery diseases. (PedSurgCIS)” under contract DVU-10-0100/2010.

REFERENCES


