An Analytical Model for Estimating the Delay in Bluetooth Communications with Serial Port Profile

M.J. Morón, R. Luque, E. Casilari and A. Díaz-Estrella
Dept. Tecnología Electrónica , University of Málaga, Málaga, Spain
mjmoron,jrlg,ecasilari@uma.es

ABSTRACT

Bluetooth is currently a major technology for the deployment of wireless short range communications. This paper presents an ana-lytical model to compute the delay of Bluetooth transmissions with Serial Port Profile (SPP), which is nowadays widely utilized by commercial Bluetooth-enabled devices. In particular, the pro-posed equations permit to estimate the packet delay in ideal transmission conditions (when no retransmission occurs) and also when environmental noise induces losses and consequently there exist a certain probability that a packet has to be retransmitted. The model takes into consideration the overhead and segmenta-tion introduced by the protocols involved in the transmission as well as the extra delay introduced by the retransmissions. The model has been empirically validated through the measurements of Bluetooth connections in an actual test-bed.

Keywords
Bluetooth, Serial Port Profile, transmission delay.

1. INTRODUCTION

Bluetooth (BT) has been widely adopted as the basic technology for the implantation of many communication services that require wireless low power short range communications. In this sense Bluetooth is especially indicated for the development of PAN (Personal Area Network) and BAN (Body Area Networks) applications. Aiming at guaranteeing interoperability of BT devices, the BT standard specifies the so-called profiles [1]. BT Profiles provide BT nodes with standard interfaces to intercommunicate and utilize a specific service. Nowadays, the Serial Port Profile (SPP) (basis for other BT profiles) is one of the most implemented profiles in commercial BT devices, ranging from Blackberry units and Smartphones to peripherals such as keyboards, GPS or wireless biosensors. Furthermore, programming interfaces (such as JSR 82 for Java) demand the use of RFCOMM (the protocol employed by SPP) to develop Bluetooth applications.

In the literature, there are significant works that analyze the performance of Bluetooth piconets [2] [3]. Some of these studies empirically characterize the behaviour of actual BT networks (especially in the presence of other interfering communications), without providing any analytical model. Conversely, proposed analytical models of BT performance are not normally experimentally validated. Moreover, these models neglect the effect of utilizing a particular BT profile and the data segmentation that takes place at the different layers of the BT protocol stack.

The following section proposes an analytical expression to compute the delay in BT communications utilizing SPP. The validity of the model is confirmed in the third section by extensive measurements in a real BT network.

2. MODEL FOR TRANSMISSION DELAY WITH SPP

a) Delay in ideal conditions

Serial Port Profile defines the use of RFCOMM protocol, which permits BT devices to emulate RS232 cable communications. RFCOMM sends the user data (organised in frames) to the lower layers of Bluetooth stack via L2CAP (Logical Link Control & Adaptation Protocol). L2CAP (implemented at the Bluetooth host, as RFCOMM) is responsible for managing the Bluetooth QoS (Quality of Service), as well as for multiplexing, segmenting and reassembling data flowing from/to the upper layers.

The data fragmentation at RFCOMM is performed so that there is always just one RFCOMM frame contained in each L2CAP frame. L2CAP is in turn layered over Bluetooth Baseband, physical layer implemented in the Bluetooth controller. Therefore, before being sent to the radio medium, RFCOMM/L2CAP frames are fragmented in a series of Bluetooth packets at the Baseband Layer.

In order to compute the minimum delay (t_R) for transmitting N user data bytes under SPP, we must consider the impact of the protocol overhead introduced by RFCOMM and L2CAP as well as the data fragmentation that L2CAP and Baseband layers perform.

Equation (1) calculates this delay (see also [4]) taking into account that user data can be fragmented in different L2CAP frames (n_{nff} "not-final" or intermediate frames and one final frame of L_{ff} bytes) so that the reception is assumed to be finished when the last BT packet of this last frame is received.

$$t_R(N) = n_{nff} \cdot t_{ACK}(L_R + O_R(L_R) + H_L) + t_{TX}(L_{ff} + O_R(L_{ff}) + H_L) \quad (1)$$

The variables in this equation are defined as follows:

-L_R is the size at which RFCOMM will have to split the user data into a series of RFCOMM frames to deliver them to L2CAP. This size is limited by both the Maximum Frame Size (*N1*) of RFCOMM [1] and the Maximum Transfer Unit (MTU) of L2CAP for RFCOMM (M_R). Consequently, we have that:

$$L_R = \min(N1, M_R - O_{R\max}) \tag{2}$$

where O_{Rmax} is the maximum possible overhead of RFCOMM (5 bytes) so that the difference (M_R-O_{Rmax}) represents the maximum number of user data that can be conveyed in a L2CAP frame without exceeding the limit imposed by M_R. On the other hand, *N1* has a default value of 127 bytes [5], although it can be negotiated by the BT terminals in the range 23-32767 bytes.

-$n_{nff}(N)$ indicates the number of non-final RFCOMM frames in which the user data are fragmented. This number can be estimated as:

$$n_{nff} = \left\lceil \frac{N}{L_R} \right\rceil - 1 \tag{3}$$

where $\lceil x \rceil$ represents the lowest integer higher than *x*.

-L_{ff} is the number of bytes of the last RFCOMM frame, which can be defined as:

$$L_{ff} = \left((N-1) \bmod L_R\right) + 1 \tag{4}$$

-$O_R(x)$ is the protocol overhead introduced by RFCOMM in each frame: 5 bytes if the data payload (*x*) contains more than 127 bytes and 4 bytes in other case.

-H_L is the size of the L2CAP header (4 bytes in Bluetooth 1.1 and for the basic mode of Bluetooth 1.2).

Note that the previous equation (1) accounts for the segmentation that BT performs when an L2CAP frame must be transported in more than one Baseband packet. Thus, the formula considers two components, t_{ACK} and t_{TX}, which are defined as:

-The term $t_{ACK}(x)$ describes the time required by BT Baseband to send all intermediate Baseband BT packets:

$$t_{ACK}(x) = \begin{cases} 0 & x = 0 \\ 2 \cdot T_S & x \le L_1 \\ 4 \cdot T_S & L_1 < x \le L_3 \\ 6 \cdot T_S & L_3 < x \le L_5 \\ 6 \cdot T_S \cdot \left\lfloor \dfrac{x}{L_5} \right\rfloor + t_{ACK}(x \bmod L_5) & x > L_5 \end{cases} \tag{5}$$

where $\lfloor x \rfloor$ denotes the highest integer lower than *x,* T_S is the duration of a Bluetooth slot (625 µs), while L_1, L_3 and L_5 are the maximum sizes of the payload of a 1, 3 and 5-slot Bluetooth packet, respectively. These sizes are 27, 183 and 339 bytes for DH (Data High-Rate) packets and 17, 121 and 224 bytes for DM (Data Medium-Rate) packets. DM-type packets convey less user data and consequently achieve a lower throughput (if no losses take place) as they include an additional overhead that provides 2/3 FEC protection. The recursive expression in (5) computes the time necessary to acknowledge all the BT packets into which the L2CAP frames are decomposed. The formula for this minimum delay bound considers the ideal case in which no errors occur in the packets. Thus, every BT packet is always acknowledged in the next slot and with a single slot packet. Therefore, there is a constant delay of 2, 4 or 6 slots for every packet of 1, 3 and 5 slots, respectively.

-The term $t_{TX}(N)$ defines the time required for transmitting the final L2CAP frame. In this case, as the data are assumed to be received when the last BT packet of the final frame is received in the BT slave, the final acknowledgement slot has not to be computed in the delay. Consequently this time can be estimated as:

$$t_{TX}(x) = \begin{cases} 0 & x = 0 \\ T_S & x \le L_1 \\ 3 \cdot T_S & L_1 < x \le L_3 \\ 5 \cdot T_S & L_3 < x \le L_5 \\ 6 \cdot T_S \cdot \left\lfloor \dfrac{x}{L_5} \right\rfloor + t_{ACK}(x \bmod L_5) & x > L_5 \end{cases} \tag{6}$$

Note that this equation also contemplates that if the final L2CAP frame exceeds the size of a 5-slot BT packet, more than one BT packet will be needed. So, as for the case of not final L2CAP frames, the expression also computes the time to acknowledge the corresponding intermediate 5-slot BT packets.

In addition, the management of QoS in BT imposes a polling mechanism that obliges the master to address the slaves just at regular intervals (the poll interval, T_{poll}). As a result, when data are ready to be sent at the application layer, the transmission may still be delayed up to an extra period of T_{poll}. Assuming that this waiting time can be reasonably approximated by a uniform distribution, the expected mean of the actual delay in ideal conditions (t'_R) has to include the effect of the polling process by adding an offset of $T_{poll}/2$ to the previously calculated delay:

$$t'_R(N) = t_R(N) + \frac{T_{poll}}{2} \tag{7}$$

b) Delay with packet retransmissions

The previous model presumes that BT transmissions take place in ideal conditions, that is to say, when no BT packet is lost and has to be retransmitted because of an unrecoverable error. However, BT technology operates in the unlicensed Industrial, Scientific and Medical (ISM) 2.4 GHz band. So, BT communications are exposed to the interferences of other BT, 802.15.4 (Zigbee), 802.11 (WiFi) interfaces or any other device with proprietary short range communications working in the same ISM band. Due to the popularity and extension of some of these technologies (especially BT and 802.11b), in most realistic scenarios where BT technologies may be of interest, BT communications will be accomplished in very noisy environments. In spite of the diverse mechanisms that BT implements to protect data integrity (frequency hopping, CRC checking, etc), the noise introduced by the interfering devices may induce unrecoverable errors in the BT packets. These errors will be detected in the receptor and will provoke the retransmission of the corresponding packets, which obviously will increase the transmission delay.

To compute the mean delay under noisy conditions (in a piconets of just one slave), we can extend the previous model if we incorporate a certain probability of retransmitting the packets in which the user data are segmented. The main goal of this extension of the model is to analytically describe the relationship between the delay and the noise of the environment, characterised by means of a single parameter: the mean the Bit Error Rate (BER) introduced by the noise in the BT transmissions.

If we approximate that bit errors (and consequently packet retransmissions) occur in an uncorrelated way, we can directly relate the BER with *p(N)*, defined as the probability that a BT packet of *N* user data bytes has to be retransmitted because of an unrecoverable error. In case that errors followed a correlated pattern (especially if they are normally grouped in long error bursts), erroneous bits would tend to concentrate in the same packets, so the number of packet retransmissions would most probably decrease. Consequently, the consideration of the bit errors as an uncorrelated process can be regarded as a worst case scenario for the packet delay.

Assuming that bit losses in the payload are independent of the losses in the header, we can calculate *p(N)* as:

$$p(N) = p_D(N) + p_H - p_D(N) \cdot p_H \tag{8}$$

where $p_D(N)$ and p_H are the probability of retransmitting a packet because of an error in the packet payload (of *N* bytes) or in the packet header, respectively. We distinguish these two variables as the header and the payload are protected against bit errors in a different way.

The BT packet header consists of 54 bits with FEC 1/3 protection. These 54 bits are organised in 18 groups of 3 bits. As one error in a group can be corrected, a packet must be retransmitted if more than one bit error occurs in any of these 18 groups. If the BER represents the Bit Error Rate of the transmission, the probability of having no error or just one error per group (p_{NEG}) is:

$$p_{NEG} = (1-BER)^3 + 3 \cdot BER \cdot (1-BER)^2 \tag{9}$$

Consequently, the probability of experiencing at least one unrecoverable error in the whole packet header with the 18 bit groups (so that the BT packet has to be retransmitted) can be calculated as:

$$p_H = 1 - p_{NEG}^{18} \tag{10}$$

In contrast with the packet header, the probability of an unrecoverable error in the data payload $p_D(N)$ depends on the packet type that is being utilised:

-In DH packets no FEC protection is employed. Thus, packet retransmission is now necessary if any of the payload bits is erroneous: $p_D(N) = 1-(1-BER)^{n_D(N)}$ (11)

where $n_D(N)$ is the numbers of the bits in the payload, which can be calculated as: $n_D(N) = \left(N + H_P(N) + O_{CRC}\right) \cdot 8$ (12)

where O_{CRC} corresponds to the 2 bytes of the CRC overhead while $H_P(N)$ are the number of bytes of the payload header. This overhead is 1 or 2 bytes long as a function of the number of slots required by the BT packet:

$$H_P(N) = \begin{cases} 1 & N \le L_1 \\ 2 & N > L_1 \end{cases} \tag{13}$$

-In DM packets, data are protected by 2/3 FEC (10/15 shortened Hamming Code). Five redundancy bits are added for each group of 10 data bits so that the algorithm permits to correct a single bit error in any 15 bit group.

If the probability (P_{BDM}) of having less than two errors in any group is:

$$p_{BDM} = (1-BER)^{15} + 15 \cdot BER \cdot (1-BER)^{14} \tag{14}$$

the probability of having at least one unrecoverable error in the payload can be computed as:

$$p_D(N) = 1 - (p_{BDM})^{n_{gDM}(N)} \tag{15}$$

where $n_{gDM}(N)$ is the number of 15 bit groups (with 10 data bits per group) in which the $n_D(N)$ user data of the BT packet are distributed:

$$n_{gDM}(N) = \left\lceil \frac{n_D(N)}{10} \right\rceil \tag{16}$$

Once the probability of a packet retransmission (*p(N)*) is known, we can derive the mean number of times ($\overline{N_{RT_X}}$) that a BT packet has to be transmitted. If a 'reliable channel' is assumed, the Bluetooth recommendation permits to fix no limit to the number of retransmissions of the same packet. As a consequence, the Baseband shall continue retransmitting an erroneous segment until it is properly acknowledged or a link loss occurs.

If a BT packet can be infinitely retransmitted, $\overline{N_{RT_X}}$ can be straight-forwardly computed as:

$$\begin{aligned} \overline{N_{RT_X}} &= 1 \cdot (1-p(N)) + 2 \cdot p(N) \cdot (1-p(N)) \\ &+ 3 \cdot p(N)^2 \cdot (1-p(N)) + \ldots = \\ &= (1-p(N)) \cdot \sum_{i=1}^{\infty} i \cdot p(N)^{i-1} = \frac{1}{1-p(N)} \end{aligned} \tag{17}$$

As the transmission between the master and the slaves in Bluetooth is governed by a polling process, when an unrecoverable bit error is detected in a Bluetooth terminal, the packet will have to wait a polling interval (T_{poll}) every time that it is retransmitted.

The calculation of the mean time for the transmission of a block of *N* user data bytes through SPP profile will have to take into account these retransmissions. In the previous section, Equations (5) and (6) compute the time to transmit the intermediate ($t_{ACK}(x)$) and the final ($t_{TX}(x)$) L2CAP frames when no bit error takes place. To cope with the retransmissions, we can modify these equations to add the mean expected delay of the retransmissions:

$$t_{ACK}(x) = \begin{cases} 0 & x = 0 \\ 2 \cdot T_S + \left(\overline{N_{RT_X}} - 1\right) \cdot T_{poll} & x \le L_1 \\ 4 \cdot T_S + \left(\overline{N_{RT_X}} - 1\right) \cdot T_{poll} & L_1 < x \le L_3 \\ 6 \cdot T_S + \left(\overline{N_{RT_X}} - 1\right) \cdot T_{poll} & L_3 < x \le L_5 \\ \left(6 \cdot T_S + \left(\overline{N_{RT_X}} - 1\right) \cdot T_{poll}\right) \cdot \left\lfloor \dfrac{x}{L_5} \right\rfloor & \\ +t_{ACK}(x \bmod L_5) & x > L_5 \end{cases} \tag{18}$$

$$t_{TX}(x) = \begin{cases} 0 & x = 0 \\ 2 \cdot T_S + \left(\overline{N_{RT_X}} - 1\right) \cdot T_{poll} & x \le L_1 \\ 3 \cdot T_S + \left(\overline{N_{RT_X}} - 1\right) \cdot T_{poll} & L_1 < x \le L_3 \\ 5 \cdot T_S + \left(\overline{N_{RT_X}} - 1\right) \cdot T_{poll} & L_3 < x \le L_5 \\ \left(6 \cdot T_S + \left(\overline{N_{RT_X}} - 1\right) \cdot T_{poll}\right) \cdot \left\lfloor \dfrac{x}{L_5} \right\rfloor & \\ + t_{TX}(x \bmod L_5) & x > L_5 \end{cases} \quad (19)$$

In these new expressions the term $\left(\overline{N_{RT_X}} - 1\right) \cdot T_{poll}$ represents the mean delay introduced by the retransmissions as $\left(\overline{N_{RT_X}} - 1\right)$ is the mean number of times that a packet is retransmitted while T_{poll} is the fixed delay that a packet loss introduces. With these new definitions of $t_{ACK}(N)$ and $t_{TX}(N)$, the mean required time for the transmission of N bytes through SPP profile can be estimated as in the optimal case, by applying the equations (1) and (7).

3. EMPIRICAL MODEL EVALUATION

Validation without packet losses

We have checked the accuracy of the proposed model by measuring the end-to-end delay in systematic transmissions programmed for an actual Bluetooth network of two nodes employing SPP. As is it is indicated in the testbed sketched in Fig.2, both nodes (BT master and slave) were executed in the same equipment (a PC with two USB Bluetooth adapters) to avoid synchronization problems in the measurement of the delay. As BT adapters, we utilised different USB dongles with CSR Bluetooth 1.1 chipsets.

To optimise the transmission conditions and minimise any possible interference, both BT modules were located in a small metal-covered box. Power control by the BT adapters was also proved to remove any influence of the possible internal reflections. The communication between the master and the slave was established through two C programs that made use of the BlueZ protocol stack [4]. This stack establishes a value of 1008 bytes for the parameter *N1* while it fixes M_R to 1013 bytes. Each experiment consisted in the transmission through a BT socket of a user data block of a pre-determined size between 10 and 1500 bytes (this range was swept with increments of 10 bytes). The delay for each data block was computed as the time elapsed from the start of the data transmission to the reception of the last data bit in the slave. The delay was measured at both the application and HCI (Host Controller Interface, by means of *HCIdump* tool [6]) layers. Differences between these two measurement points were found to be always below 1 ms, which indicates that factors such as the Operating System and USB interfaces have a minor impact on the results.

Fig. 2 compares the results of the analytical model (t'_R) and the measurements on the real connections when both types of BT packets (DH and DM) are employed. For the actual BT transmissions, each point represents the mean value of 1000 different transmissions executed with the same data size. Figure 1.a shows the results when the minimum Poll interval (T_{POLL}) that the CSR BT module can guarantee (10 ms) is selected. The results in Figure 1.b corresponds to the case where the Poll interval is not explicitly defined so that it is set to a default value of 25 ms.

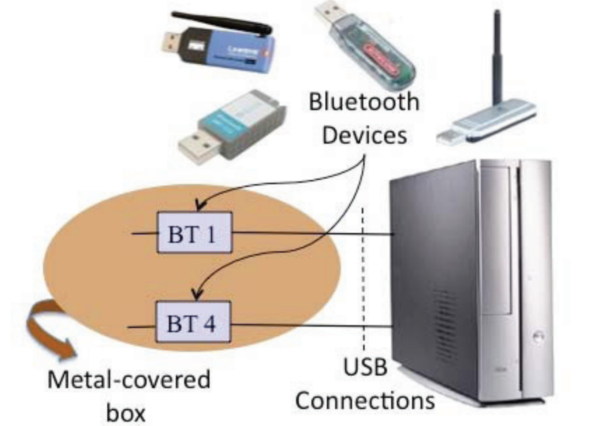
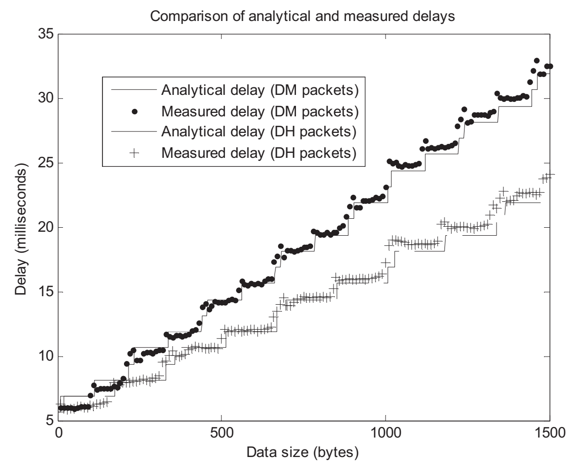
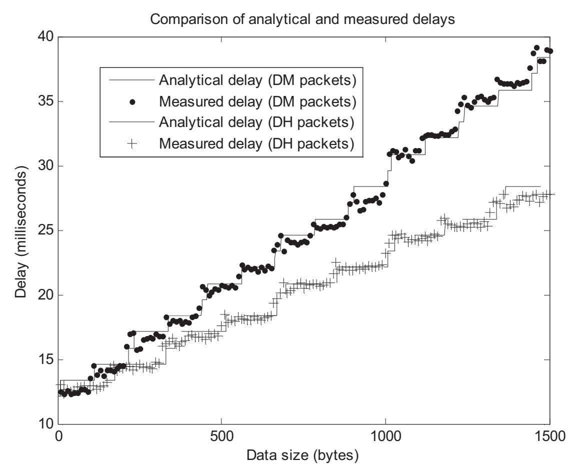


Figure 1. Testbed for the experiments in ideal conditions (without packet retransmissions)



a) T_{poll}= 10 ms



b) T_{poll}= 25 ms (default value)

Figure 2. Comparison of the theoretical minimum bound computed with the proposed model and the measured delay in the actual BT transmissions under ideal conditions

The measurements clearly verify the capability of the analytical model to characterise the end-to-end delay. In the figure the periodical 'steps' of the graphs correspond with the filling of 5-slot BT packets and the necessity of waiting for an acknowledgment to send the remaining data in a new BT packet. On the other hand, the values of $N1$ and M_R has been selected according to BlueZ implementation, so that when the maximum frame size is reached, three complete 5-slots BT (DH case) and four 5-slots plus 1-slot packets (DM case) are required at the Baseband layer. This prevents a segmentation mismatch to occur. So the figure shows that when data exceeds the MTU of L2CAP, the delay transmission is not especially increased.

Validation with packet losses

To evaluate the validity of the proposed model with packet retransmissions we utilised the testbed depicted in Fig. 3. When compared with the previous experimental environment for the connections without retransmissions, in this new testbed BT modules are removed out from the metal-covered box so that they get exposed to any other wireless communication running on the same frequency band.

In order to induce errors in the bits transmitted by Bluetooth and, consequently, packet retransmissions (in case that the errors cannot be recovered), we located an interfering WiFi traffic source in the vicinity of the BT interfaces. In particular an 802.11g connection (also operating in the same 2.4 GHz ISM band of Bluetooth) was established between the PC with the two BT modules under test and an 802.11g Access Point (also performing as IP router). For this purpose, an 802.11b PCMCIA interface is connected to the PC while the connection is accomplished through an UDP socket between the PC and another terminal connected via Ethernet with the Access Point.

To increase the noise introduced in the BT transmissions the Access Point was situated only 60 cm away from the BT interfaces. Similarly, the tests were executed under heavy traffic conditions in the interfering connections, with a bit rate between the second terminal and the PC of up to 10 Mbit/s. Using *iperf* tool [7], this background traffic was generated with a constant bit rate source emitting 1470–byte packets.

In our testbed, the BER of every experiment is indirectly calculated from the Link Quality (LQ) estimation which is provided through the HCI interface by the CSR Bluetooth Chipset [7] utilised by the BT devices. LQ is an integer (discrete) value between 0 and 255. According to CSR chipset specification (see [9] for more details), the BER can be estimated from LQ with the following formula:

$$BER = \begin{cases} (255 - LQ)\cdot 0.25\cdot 10^{-4} & 215 \le LQ \le 255 \\ 0.001 + (214 - LQ)\cdot 8\cdot 10^{-4} & 90 \le LQ < 215 \\ 0.1 + (89 - LQ)\cdot 64\cdot 10^{-4} & 0 \le LQ < 90 \end{cases} \quad (20)$$

As the chipset computes the LQ parameter only for the information contained in the BT packets that is protected with a FEC codification, the performed evaluation of our model is limited to DM packets. As FEC algorithm is only employed to protect the headers in DH packets, the value of the BER cannot be estimated from the estimation of LQ parameter in the case of DH type packets. However, we think that the conclusions obtained about the validity of the model with DM packets can be reasonably extended to DH packets.

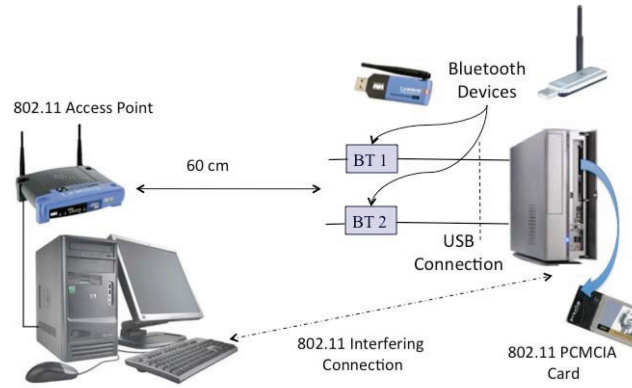


Figure 3. Testbed for the experiments with packet retransmissions

To check the accuracy of the analytical model we followed the same procedure as in the ideal case without packet retransmissions: a packet of predetermined size (between 10 and 1500 bytes) is sent from the BT master to the BT slave. For the test with each packet size (which is repeated 1000 times with an interval of 200 ms between two consecutive iterations) the transmission delay was computed in the reception point. After the corresponding 1000 transmissions of each size, a command is sent to the BT modules to compute the LQ and the corresponding BER.

The graphs in Figure 4 depict the measured mean delays for two different interference conditions (regulated by the traffic load generated in the interfering source). These noise conditions are characterised by the mean of the values of the BER estimated for the different packet sizes (the standard deviation of the estimated BER, considerably lower than the mean values, is specified in the figure captions). The graphs also show the delay that is computed by the analytical extended model when this mean BER value is considered as an input in the model to account for the effects of the retransmissions. The graphs illustrate that the analytical model can accurately fit the empirical results (a similar performance of the model has been detected for other interference conditions). Thus, the model could be appropriate to predict the performance (and even the usability) of Bluetooth technology for a communication application that have to be deployed in noisy environments.

4. CONCLUSIONS

This work has proposed and validated an analytical model to compute the delay in Bluetooth transmissions under the Serial Port Profile. In contrast with other studies, the model takes into account the headers and the segmentation introduced by the different communication protocols (RFCOMM, L2CAP, Baseband) in the Bluetooth stack.

The model is initially developed to characterise the minimum delay in ideal transmission conditions in which no packet has to be retransmitted. As in most practical applications BT connections will be interfered by other devices operating in the same band, the model is extended to cope with packet losses. Thus, assuming that the Bit Error Rate (BER) characterises the noise in the environment, the extended model determines the packet delay as a function of the probability of retransmitting a packet, which is in turn forwardly derived from the BER. Both for the cases with and without packet transmissions, the empirical evaluation with actual Bluetooth devices show the capability of the analytical model to predict the transmission delay and, indirectly, the applicability and viability of Bluetooth technology as a function of the application requirements (user data size) and environment conditions.

The model has been developed for the version 1.1 of the Bluetooth standard but it can be easily extended to the recent 2.0 and 2.1 versions.

ACKNOWLEDGMENTS
This work was supported by Project No. TEC2006-12211-C02-01.

REFERENCES

[1] Bluetooth Special Interest Group (SIG), “Specification of the Bluetooth System vol. 2: Profiles”, Version 1.1, February, 2001.

[2] Huang, P., Boucouvalas, A. C., “Bluetooth L2CAP Layer Modelling and Performance Analysis”, Mediterranean Journal of Computers and Networks, Vol. 2, No. 1, January 2006, pp.41-46.

[3] Misic, J., Misic, V. B., "Performance analysis of Bluetooth piconets with finite baseband buffers", Wiley Wireless Communications and Mobile Computing (WCMC), Vol. 5, No. 8, December 2005, pp.917-925.

[4] M.J. Moron, R. Luque, E. Casilari, A. Diaz-Estrella, “Minimum delay bound in Bluetooth transmissions with serial port profile”, *Electronics Letters*, Vol. 44, Issue 18, August 28 2008, pp. 1099-1100.

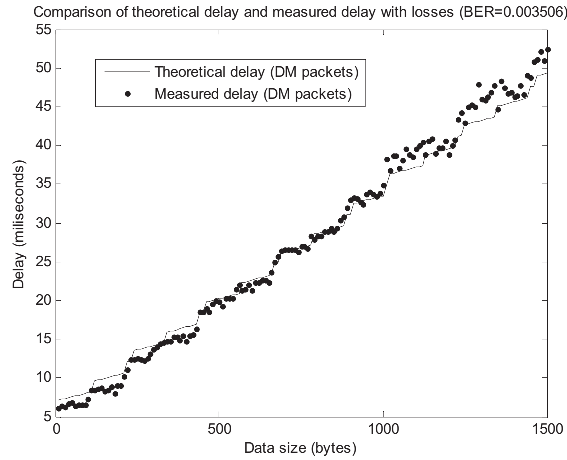
[5] Bluetooth Special Interest Group (SIG), “RFCOMM with TS 07.10, Serial Port Emulation”, Bluetooth Specification Version 1.1, June 2003.

[6] BlueZ, Official Linux Bluetooth Protocol stack, http://www.bluez.org/

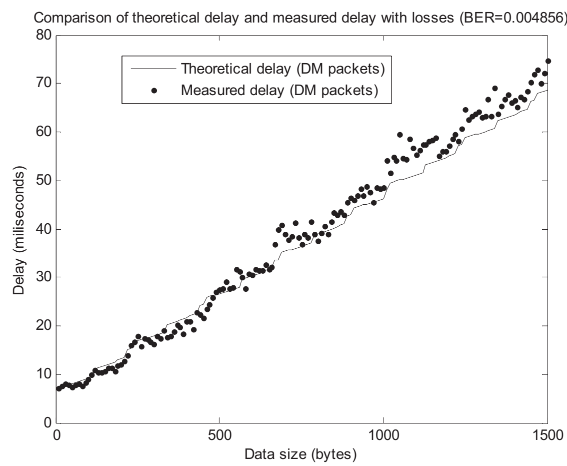
[7] Iperf - Internet Protocol bandwidth measuring tool: http://sourceforge.net/projects/iperf/?abmode=1

[8] Bluetooth Special Interest Group (SIG), «Bluetooth Core Specification Version», February 2001.

[9] CSR, “Cambridge Silicon Radio Plc. BlueCore Bluetooth, chipset”, URL: http://www.csr.com



a) BER=0.003506 (std. deviation=0.001395)



b) Mean BER=0.004856 (std. deviation=0.00188)

Figure 4. Comparison of the theoretical delay computed with the model and the measured delay in the actual BT transmissions under an interfering traffic source