Techniques to Improve the Performance of TCP in a mixed Bluetooth and WLAN Environment

N. Golmie and O. Rebala
National Institute of Standards and Technology
Gaithersburg, Maryland 20899
Email: nada.golmie@nist.gov

Abstract—**A major challenge for the WLAN technology stems from having to share the 2.4 GHz ISM band with other wireless devices such as Bluetooth radios. The main goal of this paper is to investigate the use of techniques to mitigate the effects of interference for Bluetooth and WLAN and discuss the resulting performance trade-offs. We compare the performance of the Bluetooth and WLAN systems and evaluate how each technique improves or degrades TCP performance. Simulation results for selected scenarios and configurations of interest are obtained and the performance of Bluetooth and WLAN is measured in terms of packet loss, TCP throughput and delay.**

Keywords— **WPANs, Bluetooth, Interference, MAC scheduling, TCP performance.**

I. Introduction

Since the Bluetooth and 802.11b technologies use the 2.4 GHz ISM band, devices operating in close proximity may suffer from mutual interference and significant performance degradation in terms of packet loss, lower throughputs and higher delays.

Various techniques and algorithms aimed at reducing the impact of interference have been considered [1]. These techniques range from collaborative schemes intended for Bluetooth and IEEE 802.11 protocols to be implemented in the same device [2] to fully independent solutions that rely on interference detection and estimation [3].

In this paper, we investigate the use of several techniques to mitigate interference for Bluetooth and WLAN and focus exclusively on schemes that do not require changes to either specifications. We consider rate scaling in conjunction with adaptive filtering for WLAN, and interference aware scheduling for Bluetooth. We compare the effects of using these techniques on performance for different scenarios and traffic types. Performance is measured in terms of packet loss, TCP delay and throughput.

The remainder of this paper is organized as follows. In section II, we describe the techniques used to mitigate interference. In section III, we give simulation results and concluding remarks are offered in section IV.

II. Techniques to Mitigate Interference

In this section, we present two techniques that can be used to mitigate the effect of interference. For WLAN, we consider data rate scaling, which is a common technique used in many implementations today to reduce the data rate from 11 down to 1 Mbit/s in a WLAN system. For Bluetooth, we consider a scheduling algorithm that avoids transmitting data on channels used by other wireless devices.

A. Bluetooth Interference Avoidance Scheduling

In this subsection, we give a brief overview of the Bluetooth Interference Aware Scheduling (BIAS) algorithm [4]. BIAS consists of three main components, namely a channel estimation procedure, a credit function that allocates bandwidth to each device according to its service requirements, and a priority scheduling function. Channel estimation can be based on either explicit or implicit methods. Explicit methods include BER calculation, packet loss, or frame error rate measurements performed on each receiver (master and slave device). The measurements are then collected by the master device at regular time intervals. Alternatively, implicit methods do not require the master and the slave to exchange information about the state of the channel. This information is derived by the master upon receipt of a negative ACK. We note that either channel estimation method allows the master device, which controls all data transmissions in the piconet, to avoid data transmission to a slave experiencing a "bad" frequency. Furthermore, since a slave transmission always follows a master transmission, using the same principle, the master avoids receiving data on a "bad" frequency, by avoiding a transmission on a frequency preceding a "bad" one in the hopping pattern.

This simple scheduling scheme needs only be implemented in the master device and translates into the following transmission rule. *The master transmits in a slot after it verifies that both the slave's receiving frequency and its own receiving frequency are "good". Otherwise, the master skips the current transmission slot and repeats the procedure over again in the next transmission opportunity.*

Additional considerations including bandwdith requirements and quality of service guarantees for each master/slave connection in the piconet can also be combined with the channel state information and mapped into transmission priorities given to each direction in the master/slave communication. Details on assigning transmission priorities are given in [5].

The algorithm's general steps are summarized below.

Every Even TS_f // Master transmits on frequency f
  if $TS_f + l_{dn}$ is good // Master can receive in next slot
    {
    A^f_{data} = {set of slaves s.t. ((f "good") and ($qsize > 0$) }
    if ($A^f_{data} \neq \emptyset$)
      select slave i //according to a priority criteria
      transmit data packet of size l_{dn} to slave i
    }

where l_{dn} is the length of the packet from the master to the slave ($downstream$) and TS_f is the transmission slot using frequency f.

B. WLAN Rate Scaling

Rate scaling is used in most WLAN implementations in order to optimize the range performance since the 1 Mbit/s Barker code WLAN receiver performs better than the Complementary Code Keying (CCK) 11 Mbit/s [6] [7] [8]. The Barker code correlation effectively spreads noise or the interference signal while de-spreading the desired signal and leads to lower probability of bit error (BER) than CCK for the same signal-to-interference ratio (SIR).

While there is provision in the IEEE 802.11 standards [9] to implement a rate scaling algorithm, the details remain vendor implementation specific. In our study, we use a simple two-level threshold algorithm with some hysteresis margin in order to avoid unnecessary oscillations.

```
If SIR_measured ≥ SIR^High          // the interference is low
    PHY mode = 11 Mbit/s
If SIR_measured < SIR^Low           // the interference level is high
    PHY mode = 1 Mbit/s
```

Basically, $SIR_{measured}$ is based on the Received Signal Strength Indicator (RSSI). The assumption is when the RSSI is low, the interference level is high (or the desired signal is weak), and therefore, the receiver reverts to the 1 Mbit/s mode. We set SIR^{High} and SIR^{Low} to 6 and 2 db respectively based on the BER performance of each receiver. Above 2 dB the BER for the 11 Mbit/s is below 10^{-4} [7].

In addition, we use an adpative filter in our 1 Mbit/s WLAN receiver that is able to estimate and cancel the Bluetooth interference. This technique is based on recursive least-squares lattice (RLSL) filters and generally more effective for the 1 Mbit/s WLAN receiver. It is adaptive in the sense that it does not require an a priori knowledge of the Bluetooth hopping patterns. Additional details on this method can be found in [10] where the authors discuss its effectiveness for both the 1 and 11 Mbit/s WLAN receivers.

III. Simulation Results

In this section, we present simulation results to evaluate the performance of the two techniques discussed in the previous section. We use a detailed simulation environment consisting of the MAC, PHY and channel models for Bluetooth and WLAN as described in [11]. We use the topology illustrated in Figure 1. The Bluetooth master and slave are placed one meter a part at (-0.5,0) and (0.5, 0) meters respectively. The WLAN station is located at (0,15) meters, while the WLAN server is located at (0,d) meters, where d varies along the y-axis between 0 and 10 meters.

We consider two application profiles, namely, FTP, and HTTP. We use the TCP/IP stack implemented in the OPNET library and configure the application profiles as shown in Table I. The parameters used in the setup are summarized in Table II. The simulations are run for 500 seconds of simulated time. We run 10 trials using a different random seed for each trial. In addition, to plotting the mean value, we verify that that the statistical variation around the mean values are very small (less than 1%).

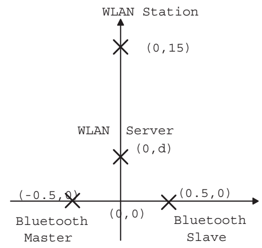

Fig. 1. Experiment Topology

The performance metrics include the packet loss, the average delay in seconds and the throughput in bytes/s. The packet loss is the percentage of packets dropped due to interference over the total number of packets received at the MAC layer. The average delay, measured at the TCP layer, indicates the time it takes to transmit a packet from the time it is passed to the TCP layer until it is successfully received at the destination. The throughput is the traffic received at the TCP layer and includes packet retransmissions.

TABLE I
Application Profile Parameters

Parameters	Distribution	Value
FTP		
Percentage of Put/Get		50%
Inter-Request Time (seconds)	Exponential	1
File Size (bytes)	Constant	2 M
HTTP		
Page Interarrival Time (seconds)	Exponential	10
Number of Objects per page	Constant	2
1st Object Size (bytes)	Constant	10000
2nd Object Size (bytes)	Uniform	(2000, 100000)

TABLE II
Simulation Parameters

Bluetooth Parameters	**Values**
ACL Baseband Packet Encapsulation	DH5
Transmitted Power	1 mW
Slave Coordinates	(-0.5, 0)
Master Coordinates	(0.5,0)
WLAN Parameters	**Values**
Transmitted Power	25 mW
Data Rate	11 Mbit/s if not rate scaling
Station Coordinates	(0,15)
Server Coordinates	(0,d)
PLCP Header	192 bits
Packet Header	224 bits

We run simulations for three different experiments where we vary the profiles used for the Bluetooth and WLAN applications as shown in Table III. In experiment 1, both WLAN and Bluetooth use the FTP profile, while in experiments 2 and 3, the WLAN (/Bluetooth) application uses FTP (/HTTP) and HTTP (/FTP) traffic respectively. Although a large amount of data was obtained at analyzed, due to space constraints, only a small subset of the results is shown here.

In the next two subsections, we discuss the performance of

TABLE III

EXPERIMENT SUMMARY

Scenario	WLAN	Bluetooth
1	FTP	FTP
2	FTP	HTTP
3	HTTP	FTP

TCP over WLAN and Bluetooth in terms of the techniques proposed. We compare the performance of WLAN and Bluetooth when rate scaling is used for WLAN and scheduling is used for Bluetooth. For each experiment, we run 4 simulations in order to identify the benefits of each algorithm and its interactions with other schemes. *None* refers to the case when no algorithm is used. *Rate Scaling* means that WLAN uses the rate scaling algorithm, while *Scheduling* means that Bluetooth uses BIAS. The case where WLAN uses rate scaling and Bluetooth uses BIAS simultaneously is refered to as *Rate Scaling + Scheduling*.

A. TCP over WLAN

Figure 2 (a) gives the packet loss with respect to the y-coordinate of the WLAN server, d, when both WLAN and Bluetooth use the FTP profile. When no algorithm is used, the packet loss can be up to 14% when the WLAN server is close to the Bluetooth piconet (d=0 meters). As the server moves away from the Bluetooth piconet, the packet loss drops to zero ($d \geq 5$ meters). When rate scaling is used, the packet loss drops to 5% when d=0 meters. This packet loss observed is due to the intermittent use of the 11 Mbit/s WLAN receiver before the 1 Mbit/s mode is used. While the adaptive filter used in the 1 Mbit/s receiver is able to reduce the packet loss to zero, the 11 Mbit/s receiver is less robust and yields a relatively high packet loss. Observe that the packet loss is zero when Bluetooth uses BIAS since the Bluetooth transmitter avoids using the same frequency used by WLAN.

Figure 2(b) illustrates the throughput of the WLAN server. When no algorithm is used, the throughput starts at 240 Kbyte/s when d=0 meters, and goes up to 350 Kbyte/s when $d \geq 5$ meters and the packet loss is zero. Observe that when BIAS is used, the throughput remains around 350 Kbyte/s since no packets are lost. Since rate scaling involves reducing the WLAN bit rate from 11 to 1 Mbit/s, this yields to reducing the throughput to 50 Kbyte/s. As expected, rate scaling can reduce the packet loss, at the cost of reducing the throughput.

Figure 3(a) and (b) give the WLAN packet loss and delay respectively for experiment 3. In this case, the WLAN uses the HTTP profile while the Bluetooth uses the FTP profile. The packet loss depicted in Figure 3(a)) is slightly less than when WLAN uses the FTP profile (Figure 2(a)), however it follows a similar trend. The packet loss with BIAS is around 1% when $d < 4$ meters.

An important metric for HTTP is the delay to access data, therefore in Figure 3(b), we plot the TCP delay. Note that it is 15 ms when the packet loss is 12% (Figure 3(a)) and drops down to 2.5 ms when the packet loss is zero. Observe that when rate scaling is used the delay remains flat at 5 ms. On the other hand, when Bluetooth uses BIAS, the delay starts at 5 ms and drops down to 2.5 ms.

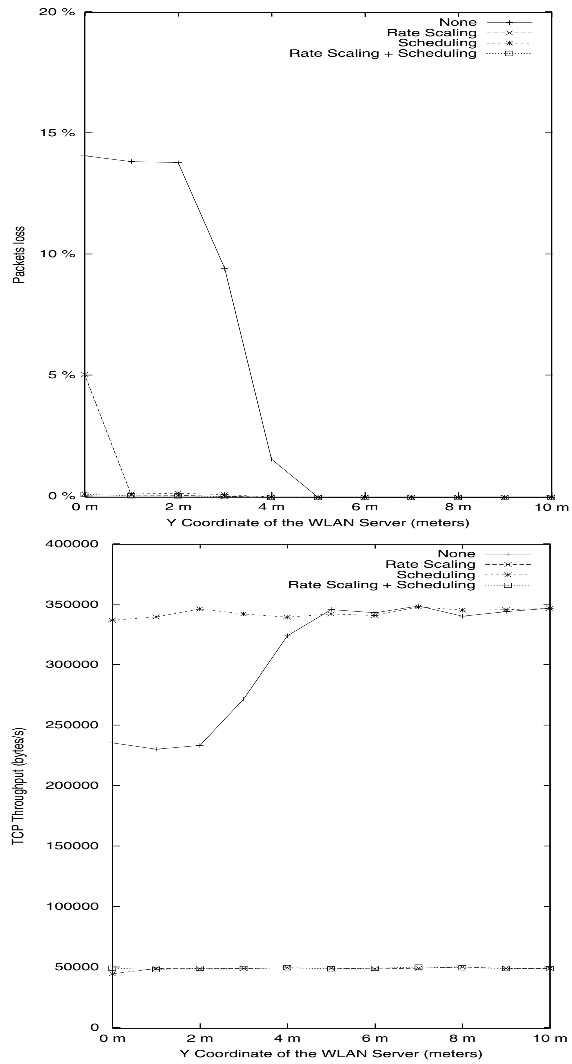

Fig. 2. $\frac{(a)}{(b)}$ Experiment 1. WLAN FTP Performance. (a) Probability of Packet Loss. (b) TCP Throughput

Overall, we note that the use of Bluetooth scheduling improves the WLAN performance and brings it closer to the ideal case when no interference is present. The use of rate scaling produces interesting but expected trade-offs. While the WLAN packet loss is reduced, the delay is increased and the throughput is reduced.

B. TCP over Bluetooth

Figure 4(a) gives the packet loss for the Bluetoth master device as a function of the WLAN server y coordinate, d. When no algorithm is used, the packet loss is around 10% for d =0 meters. When $2 \leq d \leq 6$ meters, we observe a spike with a peak of 17% at d=4 meters. This is due to the closed loop interference between the WLAN and Bluetooth systems. To better understand the interactions, we look at Figure 2(a). Since less WLAN packets are lost (more WLAN packets are transmitted), this causes more interference on Bluetooth and thus more packet loss. This trend is valid until d=5 meters and the WLAN packet

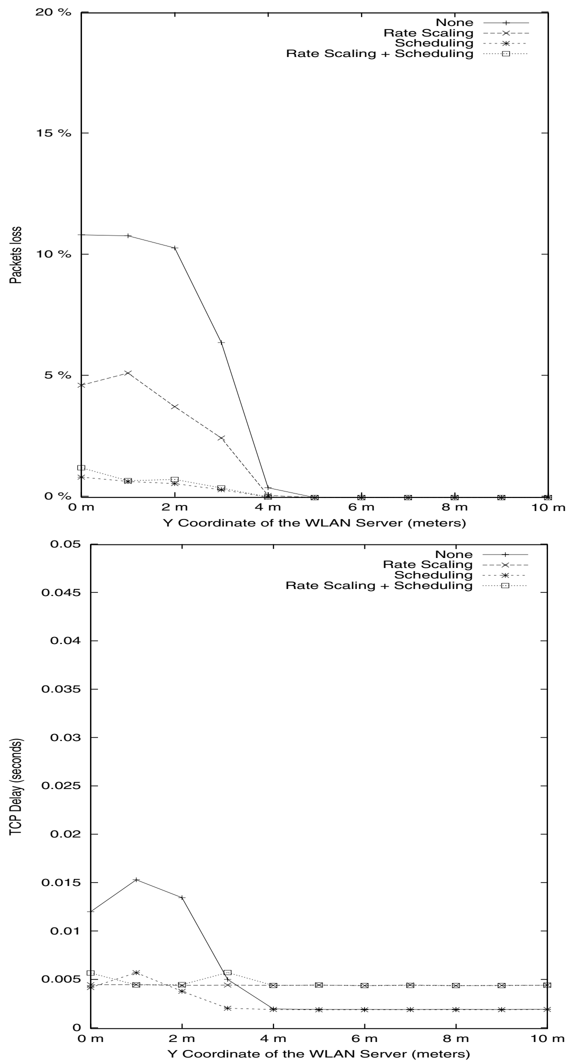

Fig. 3. $\frac{(a)}{(b)}$ Experiment 3. WLAN HTTP Performance. (a) Probability of Packet Loss. (b) TCP Delay

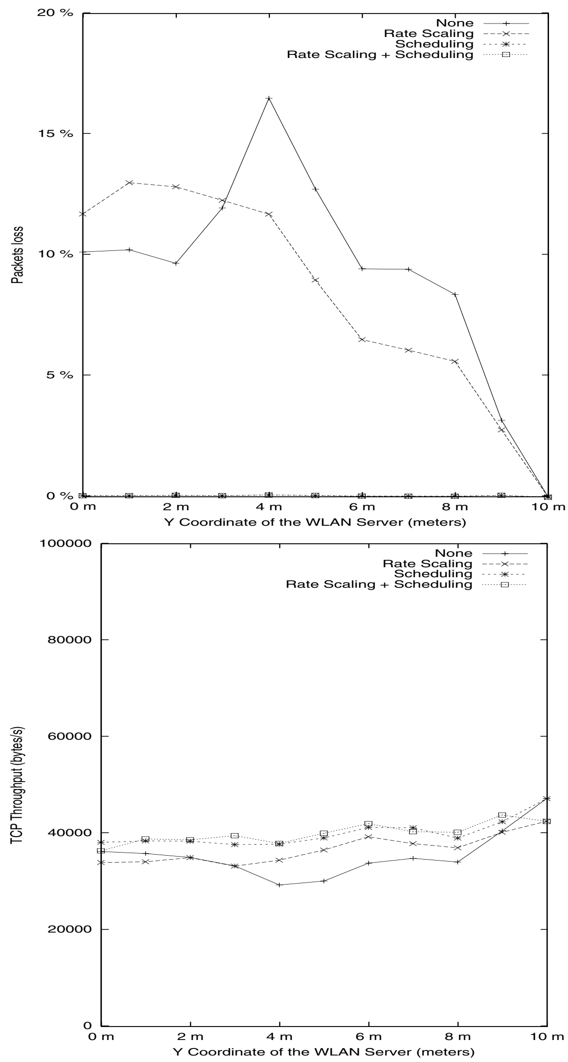

Fig. 4. $\frac{(a)}{(b)}$ Experiment 1. Bluetooth FTP Performance. (a) Probability of Packet Loss. (b) TCP Throughput

loss is zero. At that point, the Bluetooth packet loss start decreasing as the WLAN server moves further away. When rate scaling is used for the WLAN, we note a packet loss of 12% for Bluetooth at d=0 meters. The packet loss remains high until d=10 meters. This is due to the fact that rate scaling causes the WLAN to transmit packets at a lower rate, occupying more time in the air and causing more interference on Bluetooth. Note that when scheduling is used for Bluetooth, the packet loss is reduced to zero.

The TCP throughput depicted in Figure 4(b), closely follows the packet loss curves in Figure 4(a). When no algorithm is used, the throughput is 38 Kbyte/s when d=0 meters, 35 Kbyte/s when d=5 meters, and 45 Kbytes/s when d=10 meters, which clearly reflects a 12%, 17%, and 0% packet loss respectively. As expected, when rate scaling is used the throughput is about 10% lower than when scheduling is used reflecting the 10% packet loss observed in Figure 4(a).

The results for packet loss and delay when Bluetooth uses the HTTP profile (experiment 2), are illustrated in Figures 5(a) and (b) respectively. The packet loss when rate scaling is used is slightly higher (11%) than when no algorithm is used (8%). The packet loss is zero when scheduling is used.

The TCP delay in Figure 5(b) starts at 33 ms when rate scaling is used at d=0 meters. It is 7 ms and 12 ms when scheduling and no algorithm are used respectively. When no interference is present (d=10 meters), the delay is around 6 ms. Thus, the scheduling algorithm yields a slight increase in delay (around 1 ms) while reducing the packet loss to zero.

In summary, the main advantages of using scheduling in terms of the Bluetooth performance, are to reduce the packet loss to zero at almost no cost to either thoughput or delay. On the other hand the use of rate scaling for WLAN leads to higher packet losses for Bluetooth, including higher delays and lower throughput.

IV. Concluding Remarks

In this paper, we study the performance of TCP over Bluetooth and WLAN in a mutual interference environment consist-

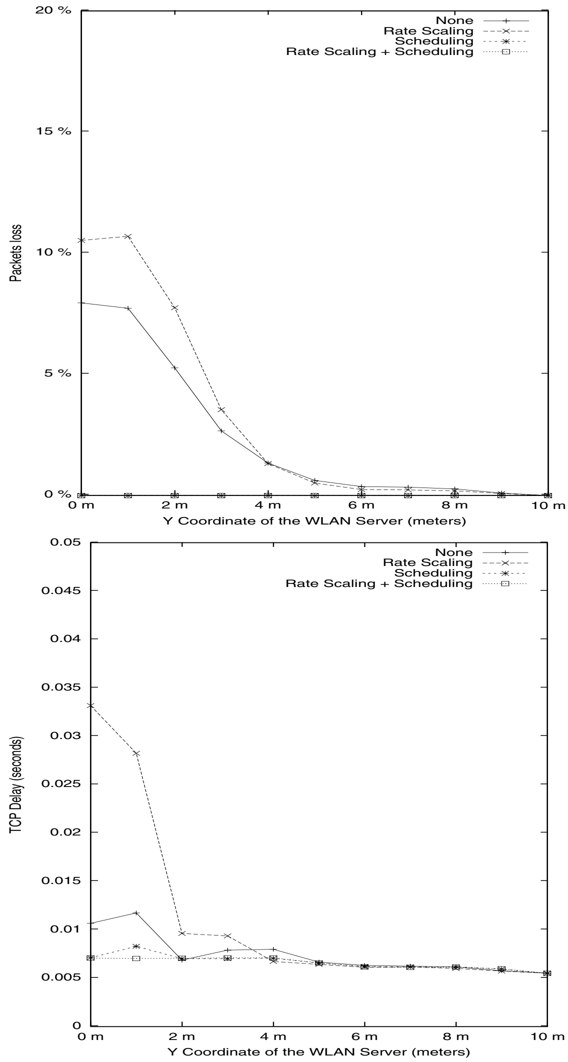

Fig. 5. $\frac{(a)}{(b)}$ Experiment 2. Bluetooth HTTP Performance. (a) Probability of Packet Loss. (b) TCP Delay

ing of two Bluetooth and two WLAN devices operating at the same time. We consider two application profiles, namely HTTP and FTP.

We investigate the use of two techniques to mitigate the effects of this mutual interference. Both techniques rely on detecting the presence of other wireless systems and adapting to the interference environment. For Bluetooth, we use a scheduling scheme that consists of avoiding to transmit a packet on a frequency used by the WLAN system. On the other hand, for WLAN we use rate scaling which consists of reverting to the more robust 1 Mbit/s mode. We also include in the 1 Mbit/s receiver used, an adaptive filter that can notch out the Bluetooth signal. Both techniques do not require any changes to either the Bluetooth or the IEEE 802.11 specifications.

Our simulation results indicate that the use of Bluetooth scheduling improves both the Bluetooth and WLAN systems' performance. The packet loss is reduced to zero, while the throughput is increased, and the delay decreased. On the other hand, the benefits of using rate scaling in the WLAN system are clearly less pronounced. While the packet loss is reduced for WLAN due to the the use of a more robust receiver and an adaptive filter, the performance of Bluetooth is degraded due to the increase of the WLAN packet transmission. As a result, the probability of a packet collision in time and frequency is much higher leading to higher packet loss and delays, and lower throughputs.

Finally, we note that these observations apply to either FTP or HTTP traffic. While the exact performance results depend on the parameters of the application profile used, the general trends hold in most cases studied.

ACKNOWLEDGEMENTS

The authors would like to thank Amir Soltanian for his help in the PHY layer simulation models and his assistance in making the 1 Mbit/s WLAN receiver with the adaptive filter available to use in the combined MAC and PHY simulation framework.

REFERENCES

[1] Carla F. Chiasserini, and Ramesh R. Rao, " Coexistence mechanisms for interference mitigation between IEEE 802.11 WLANs and bluetooth ," in *Proceedings of INFOCOM 2002*, 2002, pp. 590–598.

[2] J. Lansford, R. Nevo, E. Zehavi, "MEHTA: A method for coexistence between co-located 802.11b and Bluetooth systems," in *IEEE P802.11 Working Group Contribution, IEEE P802.15-00/360r0*, November 2000.

[3] B. Treister, A. Batra, K.C. Chen, O. Eliezer, "Adapative Frequency Hopping: A Non-Collaborative Coexistence Mechanism," in *IEEE P802.11 Working Group Contribution, IEEE P802.15-01/252r0*, Orlando, FL, May 2001.

[4] N. Golmie, N. Chevrollier, and I. Elbakkouri, "Interference Aware Bluetooth Packet Scheduling," in *Proceedings of GLOBECOM'01*, San Antonio, TX, November 2001.

[5] N. Golmie, "Bluetooth Dynamic Scheduling and Interference Mitigation," in *ACM Mobile Network, MONET*, 2002.

[6] B. Sklar, *Digital Communications: Fundamentals and Applications*, Prentice Hall, 1997.

[7] A. Soltanian and R. E. Van Dyck, "Physical layer performance for coexistence of Bluetooth and IEEE 802.11b," in *Virginia Tech Symposium on Wireless Personal Communications*, June 2001.

[8] N. Golmie, R.E. Van Dyck, A. Soltanian, A. Tonnerre, and O. Rebala, "Interference Evaluation of Bluetooth and IEEE 802.11b Systems," in *ACM Wireless Network, WINET*, 2002.

[9] IEEE Std. 802-11, "IEEE Standard for Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specification ," June 1997.

[10] A. Soltanian, R. E. Van Dyck, and O. Rebala, "Rejection of Bluetooth Interference in 802.11 WLANs," in *Proceedings of IEEE VTC, Fall 2002*, September 2002.

[11] N. Golmie, R.E. Van Dyck and A. Soltanian, "Interference of Bluetooth and IEEE 802.11: Simulation Modeling and P erformance Evaluation," in *Proceedings of the Fourth ACM International Workshop on Modeling, Analysis, and Simulation of Wireless and Mobile Systems, MSWIM'01*, Rome, Italy, July 2001.