Abstract—The Bluetooth specification allows both asynchronous (ACL) and synchronous (SCO) links to be present in a piconet. However, the performance of ACL traffic rapidly deteriorates when an SCO link is present. This paper investigates the possibility of replacing the Bluetooth SCO connection with a QoS-constrained asynchronous link that uses multi-slot ACL packets. We have analyzed the performance of this scheme, dubbed pseudo-SCO, under limited service and exhaustive service scheduling. It was found that the pseudo-SCO scheme allows asynchronous traffic to experience lower delays than with the regular SCO connection, while supporting the bandwidth requirements of SCO traffic.

Keywords: Bluetooth piconet, scheduling, CBR traffic, limited service scheduling, exhaustive service scheduling

I. INTRODUCTION

Bluetooth is a wireless communication technology for short range ad hoc networks [1]. It uses a set of 79 (or 23, in some countries) RF frequencies in the ISM band around 2.4GHz [2]. Bluetooth devices are organized in piconets, star-like ad hoc networks that contain one master and several slave devices, up to seven of which can be simultaneously active at any given time.

Bluetooth provides two types of links between the master of a piconet and its slaves. Asynchronous data traffic is supported through the so-called Asynchronous Connectionless (ACL) links, in which the master is free to poll or not to poll the slave at will. The slave may talk back only when addressed by the master, and only immediately after being addressed by the master. (Two consecutive packets, the downlink packet and the immediately following uplink response, are often referred to as a frame.)

The other link type is known as Synchronous Connection-Oriented (SCO) link; it is specifically designed to support synchronous, constant bit rate (CBR) traffic such as voice [2]. In order to satisfy the bandwidth constraints for the transmission of one 64kbps voice channel, a strict timing scheme has to be observed. As a consequence, the performance of asynchronous data traffic deteriorates substantially; in extreme cases, such traffic cannot be supported at all.

Therefore, it would be of interest to find an alternative scheme that would allow simultaneous use of synchronous and asynchronous traffic, with as little performance degradation as possible. In this paper we show that a feasible solution, which will be referred to as pseudo-SCO or pSCO for short, may indeed be implemented using only the facilities already provided in the current Bluetooth specification [2]. The pSCO scheme allows the piconet to carry both types of traffic, whilst providing guaranteed performance for the synchronous traffic and acceptable performance for the asynchronous one.

It should be noted that Chawla et al. have investigated the possibility of bandwidth conservation using ACL packets for voice transmission [3], while Famolari and Anjum have analyzed the effect of different traffic priorities, noise, and interference, on voice traffic [4]. The work described in this paper, however, focuses on the performance of asynchronous traffic, and thus complements, rather than repeats, the results presented in these papers.

The paper is organized as follows: in Sec. II we describe the operation of the piconet with an ordinary SCO link. We present the limitations that render the piconet virtually or physically unusable for asynchronous traffic, and outline the pSCO scheme for supporting both types of traffic simultaneously. Sec. III presents measurement results which confirm that significant increases in asynchronous data rates can indeed be obtained with the pSCO scheme. In addition, we compare the performance of two intra-piconet polling schemes, limited service and exhaustive service. Finally, Sec. IV concludes the paper.

II. PICONE T OPERATION AND MODELING

As mentioned above, Bluetooth devices use a master-driven communication mechanism. Two Bluetooth devices establish a master-slave relationship, or link, through a defined inquiry and page procedure. The link is always established in ACL mode first. Different types of packets may be used, with different lengths (1, 3, or 5 slots) and different information-carrying capacity. Once the ACL link is established between the master and a slave, either of them may request that an SCO link be created. This link is operational until removed at an explicit request from either participant. A piconet may support up to three SCO links, and any given slave may participate in more than one SCO link [2].

A. Why the built-in SCO links are bad

SCO links can use three types of packets: HV1, HV2, and HV3, the characteristics of which are outlined in Table I. (A fourth packet type, known as DV, combines 10 bytes of voice data with up to 150 bits of other data; as its behavior is not different from that of the HV type packets, it will not be considered separately.) The SCO link is specifically designed to carry voice traffic at the standard, non-compressed rate of 64kbps. The SCO link can optionally use compression, but this does not change the timing scheme, nor does it affect the type of packets used. The SCO link reserves slots in master-slave communication, and the
SCO slave is allowed to send a packet in its reserved slot even if not addressed by the master in the immediately preceding master (even-numbered) slot.

<table>
<thead>
<tr>
<th>Type</th>
<th>Slot(s)</th>
<th>Payload (bytes)</th>
<th>SCO interval (slots)</th>
<th>Data rate (kbps)</th>
</tr>
</thead>
<tbody>
<tr>
<td>HV1</td>
<td>1</td>
<td>10</td>
<td>2</td>
<td>64</td>
</tr>
<tr>
<td>HV2</td>
<td>1</td>
<td>20</td>
<td>4</td>
<td>64</td>
</tr>
<tr>
<td>HV3</td>
<td>1</td>
<td>30</td>
<td>6</td>
<td>64</td>
</tr>
</tbody>
</table>

SCO links, therefore, have to follow a very strict timing scheme, which reduces (and, in some cases, eliminates) the bandwidth available to asynchronous traffic. Consequently, the maximum achievable data rates for such will be reduced, and end-to-end packet delays will increase. As these delays are mostly determined by queueing delays [5], the increase in delays and the corresponding decrease of data rates will be disproportionally high compared to the reduction in available bandwidth.

On top of all this, additional limitations are imposed on allowable packet length, with additional deterioration in performance. Namely, the ACL traffic has to fit in the intervals between subsequent SCO frames; these intervals last for only two or four slots. Consequently, five slot packets cannot be used at all, while three slot packets can be used if and only if the other packet in a frame is a single-slot packet. In the extreme case of a single SCO link with HV2 packets or two SCO links that use HV3 packets, all subsequent ACL communications must use single-slot packets only. Note that the information-carrying capacity of one slot packets (17 or 27 bytes, depending on whether FEC is used) allows data rates of up to about 170kbps only, well below the theoretical maximum for Bluetooth networks.

To summarize, the performance of asynchronous traffic in the presence of an SCO link is far from satisfactory, and it is worth investigating whether an alternative arrangement that could be found.

### B. pSCO: a scheme for synchronous traffic

It turns out that such arrangement is indeed possible, as the Bluetooth Link Manager provides basic Quality of Service (QoS) capabilities [2]. Namely, the master and an ACL slave may set up the maximum polling interval \( T_{poli} \), i.e., the maximum time between subsequent transmissions. This polling interval is guaranteed in the active mode, except when there are collisions with page, page scan, inquiry, and inquiry scan. This mechanism may be used as the basis for an improved synchronous transmission mode, which we will call pseudo-SCO mode, or pSCO for short.

Basic requirements to be satisfied in the pSCO mode are related to the bandwidth and latency of the synchronous transmission channel. If the signal to be transmitted is voice, the bandwidth required is 64kbps without compression, and 32, 16, or even 8kbps with appropriate compression. (Issues related to compression are beyond the scope of this paper, and we will use 64kbps as the required bandwidth in subsequent analysis.)

Latency is not very important for unidirectional traffic (e.g., a live broadcast). However, for bidirectional traffic (i.e., ordinary telephone-like conversation), the end-to-end (round trip) delays of 100 to 300ms are noticeable but still acceptable [6]. Part of these will be incurred by the transmission and queueing delays, which may be controlled through the pSCO scheme. The rest includes the time for packetization, compression, and decompression of voice signal (each of which has to be performed twice in a two-way conversation), and possibly some time for buffering in order to compensate for the packet arrival jitter [7]. These additional delay times will not be negligible, the more so because the computational power of Bluetooth devices may be limited by the (more important) requirements of low energy consumption. No hard numbers can be given here, but in general we would like to keep the latency as low as possible.

The parameters to be adjusted in order to satisfy those requirements are the type of packets to be used and the duration of the polling interval \( T_{poli} \). The obvious task sequence would be as follows.

- First, choose the type of packet. Multi-slot ACL packets have larger payloads, which means they can be sent at longer intervals, which, in turn, leave more contiguous time available for asynchronous traffic. However, noise and interference conditions might dictate the use of FEC-protected packets (i.e., DM type), which carry somewhat smaller payloads than their DH counterparts.
- Then, calculate the polling interval so as to satisfy the bandwidth requirement. As the polling interval is expressed in Bluetooth time slots, \( T_{poli} \), it should be rounded to the nearest lower integer value.
- In fact, the polling interval may be increased if we allow multiple packets to be exchanged at once: e.g., if each exchange takes two packets instead of just one, the polling interval may be doubled. Too long polling intervals, however, might violate the latency requirement.

The corresponding values for different three- and five-slot ACL packet types, assuming all exchanges are single-frame, are given in Table II. Note that the round trip transmission delay will be twice the duration of the polling interval, plus the additional delays described above.

The packet timing under the pSCO scheme is shown in Fig. 1, where the corresponding timing of the original SCO link with HV3 packets is included for comparison. (As before, gray frames correspond to pSCO frames, while the white ones cor-

---

**Table I**

<table>
<thead>
<tr>
<th>Type</th>
<th>Slot(s)</th>
<th>Payload (bytes)</th>
<th>SCO interval (slots)</th>
<th>Data rate (kbps)</th>
</tr>
</thead>
<tbody>
<tr>
<td>HV1</td>
<td>1</td>
<td>10</td>
<td>2</td>
<td>64</td>
</tr>
<tr>
<td>HV2</td>
<td>1</td>
<td>20</td>
<td>4</td>
<td>64</td>
</tr>
<tr>
<td>HV3</td>
<td>1</td>
<td>30</td>
<td>6</td>
<td>64</td>
</tr>
</tbody>
</table>

**Table II**

<table>
<thead>
<tr>
<th>Type</th>
<th>Slot(s)</th>
<th>Payload (bytes)</th>
<th>Polling interval (slots)</th>
<th>Data rate (kbps)</th>
</tr>
</thead>
<tbody>
<tr>
<td>HV3</td>
<td>1</td>
<td>30</td>
<td>6</td>
<td>64</td>
</tr>
<tr>
<td>DM3</td>
<td>3</td>
<td>121</td>
<td>24</td>
<td>64.5</td>
</tr>
<tr>
<td>DH3</td>
<td>3</td>
<td>183</td>
<td>36</td>
<td>65.1</td>
</tr>
<tr>
<td>DM5</td>
<td>5</td>
<td>224</td>
<td>44</td>
<td>65.2</td>
</tr>
<tr>
<td>DH5</td>
<td>5</td>
<td>339</td>
<td>67</td>
<td>64.8</td>
</tr>
</tbody>
</table>
As can be seen, the use of multi-slot packets for synchronous traffic means that less overall time is spent on synchronous exchanges on the average, and each interval between those exchanges lasts much longer. Asynchronous traffic will obviously suffer less disruption than in the original SCO scheme, and its performance should improve.

C. The system model

We consider the performance of a single Bluetooth piconet with 8 members, six of which are ACL slaves, and a single slave device with synchronous (SCO or pSCO) traffic. We assume that each ACL slave generates bursts (batches) of packets that follow a Poisson distribution with arrival rate $\lambda$. All packets within a burst have the same destination node, and the distribution of destination nodes is assumed to be uniform. The probabilities of ACL traffic packets being one, three, and five slots long are equal to $p_1 = p_3 = p_5 = 1/3$, respectively. The pSCO link generates packets of length $L_{sco}$ slots on every $T_{sco}$ time slots.

Note that the ordinary SCO connection with HV3 packets, as defined by the Bluetooth specification, may be described by the combination of $L_{sco} = 1$ and $T_{sco} = 6$.

Our main performance indicator will be the end-to-end packet delay $W_e$. We assume that all slaves generate ACL traffic with the same arrival rate, and that the master has no ACL traffic of its own. This is just a vehicle to simplify the derivations, rather than a conceptual limitation, and our analysis framework could easily handle the cases with non-uniform slave ACL traffic and/or ACL traffic to and from the master.

We will use two scheduling policies, limited service (in which the master polls each ACL slave for one frame only) and exhaustive service (in which the master stays with the ACL slave as long as there are packets to send in either direction). These policies were shown to achieve good results, while being simple to implement [8], [9], [5]. In fact, the Bluetooth specification does not prescribe any particular intra-piconet scheduling policy [2]. Unfortunately, scheduling policies which are optimal for polling systems, such as Stochastically Largest Queue (SLQ) [10], cannot be applied in Bluetooth – they require that “the lengths of all queues are known at all times” to the server, which is not feasible in Bluetooth networks [8]. Therefore, it may be more practical to focus on simpler policies, as was done in this paper.

![Fig. 1. Timing of different packet types with SCO and pseudo-SCO links.](image)

![Fig. 2. End-to-end packet delays for ACL traffic in the presence of a pSCO connection, under limited service scheduling.](image)
III. PICONET PERFORMANCE UNDER PSCO SCHEME

To assess the performance of the pSCO scheme, we have built a Bluetooth simulator using the object-oriented Petri Net-based simulation engine Artex by Artis Software Inc. [11] running on a Linux platform. The resulting analytical results, obtained under limited service scheduling of ACL slaves, are shown in Fig. 2, for pSCO connections that use packets of types HV3 and DH3; the use of other pSCO packets from Table II gives comparable results. The upper row shows analytical solutions, while the lower one shows simulation results. Limited service policy is used for intra-piconet scheduling of ACL slaves. We have assumed that $m = 8$, i.e., that the piconet has the maximum size allowed.

Note that the range of burst arrival rates in case HV3 packets are used, differs from that in case of DH3 packets. As the pSCO scheme uses mean burst sizes from 1 to 6, we have tried to adjust the mean burst size of the original SCO scheme so that we achieve similar data rates. Since mean packet length was $L = 3$, the average payload per packet was $27p_1 + 183p_2 + 339p_3 = 183$ bytes, or 6.77 times more than in the original SCO scheme. Using the same range of ACL data rates would exceed the network capacity and lead to queue instability. Therefore, slightly different ranges were used for the two diagrams, and the maximum value of $B$ in Fig. 2(a) corresponds to the minimum one in Fig. 2(b). Apparently, the pSCO scheme achieves much shorter delays for asynchronous traffic than the original SCO scheme.

To assess the performance of the pSCO scheme from another angle, we have also plotted the maximum achievable data rate with different packet types as a function of mean packet burst size and the corresponding polling interval $T_{sco}$. The burst arrival rates have been chosen to give the mean end-to-end packet delay of $400T = 0.25s$. The resulting diagrams are shown in Fig. 4, where data rates are expressed in bps (bits per second) per ACL slave. We note that the maximum data rates obtained with pSCO are much higher than those obtained with the original SCO. Moreover, the difference becomes more pronounced at higher values of mean burst size.

It should come as no surprise that the use of DM3 or DM5 packets gives inferior results compared to those obtained with DH3 or DH5. Namely, DM packets have lower information carrying capacity than their DH counterparts, and the polling interval has to be reduced. As a consequence, the portion of time that may be allocated to ACL traffic is lower. For example, DH3 packets use 6 out of every 36 slots, which leaves about $\alpha = 83.3\%$ of time free for ACL traffic. On the other hand, DM5 packets use 10 out of every 44 slots, so that ACL traffic is able to use only about $\alpha = 77.3\%$ of time, even though the polling interval is longer. The differences in $\alpha$ values are reflected in the differences in maximum available data rates.

Overall, the pSCO scheme is seen to offer much improved performance compared to the original SCO scheme. The improvement is due to the increase in time allocated to asynchronous traffic (i.e., $\alpha$), and the use of multi-slot packets with larger information-carrying capacity.

Whenever possible, DH-type packets should be used for the pSCO traffic, rather than the DM-type ones; in this manner, more time is left for the asynchronous traffic, and its performance improves. However, DM-type packets are more resistant to interference. On the other hand, the difference in achievable data rates from three- to five-slot packets (i.e., from DM3 to DM5, or from DH3 to DH5) is not high. Therefore, the choice of three- vs. five slots packets may be made on the basis of some other criteria, such as interference robustness, in which case shorter packets are preferable to longer ones [12].

We have also evaluated the performance of the pSCO scheme under exhaustive service scheduling; the resulting delays are shown in Fig. 3 as functions of mean burst arrival rate and mean burst size. As before, the pSCO scheme offers much reduced delays than the original SCO scheme at a comparable data rate. We note that exhaustive service intra-piconet scheduling gives slightly lower delays than limited service, but the difference is not high, and the end-to-end packet delays at the extreme end of the range of variables used on the diagrams are virtually identical.

We have also plotted the maximum data rate that can be
achieved with different packet types as a function of mean packet burst size and the corresponding polling interval, $T_{sco}$. As before, the burst arrival rates have been chosen to give the mean end-to-end packet delay of 400T. The resulting diagrams are shown in Fig. 4 for limited service and exhaustive service scheduling: in both cases, data rates are expressed in bps (bits per second) per ACL slave. As can be seen, the maximum data rates obtained with pSCO are much higher than those obtained with the original SCO, and the difference becomes more pronounced at higher values of mean burst size. The performance of exhaustive service scheduling is slightly better than that of limited service, in accordance with the observations made on the basis of end-to-end packet delays.

We note that similar results have been obtained with analytical modeling of the performance of pSCO scheme, using the the theory of $M^{[x]} / G / 1$ queues with vacations [13]. This analysis is presented in detail in [14]: however, it had to be omitted here because of space constraints.

IV. CONCLUSION

In this paper we propose an alternative scheme for supporting synchronous, CBR traffic over Bluetooth wireless channels, using longer, multi-slot ACL packets. It can easily support the bandwidth and latency requirements of synchronous traffic such as voice. The new scheme is able to provide satisfactory performance (defined in terms of end-to-end packet delay and/or corresponding data rate) for asynchronous data traffic between other slaves within the same piconet. Furthermore, exhaustive service intra-piconet scheduling was found to give slightly lower delays than limited service, whilst being equally simple to implement.

Further work in this area might concentrate on the definition and implementation of guaranteed quality of service requirements for asynchronous traffic.

Fig. 4. Maximum achievable data rates (in bps per ACL slave) as a function of polling interval and mean burst size.

REFERENCES