VOICE OVER IP PERFORMANCE MONITORING

COLE, R. G. AND ROSENBLUTH, J. H.
AT&T LABORATORIES
MIDDLETOWN, NJ
{rgcole, jrosenbluth}@att.com

ABSTRACT

We describe a method for monitoring Voice over IP (VoIP) applications based upon a reduction of the ITU-T's E-Model to transport level, measurable quantities. In the process, 1) we identify the relevant transport level quantities, 2) we discuss the tradeoffs between placing the monitors within the VoIP gateways versus placement of the monitors within the transport path, and 3) we identify several areas where further work and consensus within the industry are required. We discover that the relevant transport level quantities are the delay, network packet loss and the decoder's de-jitter buffer packet loss. We find that an in-path monitor requires the definition of a reference de-jitter buffer implementation to estimate voice quality based upon observed transport measurements. Finally, we suggest that more studies are required, which evaluate the quality of various VoIP codecs in the presence of representative packet loss patterns.

1 INTRODUCTION

There is great interest in supporting voice applications over both the public Internet and private intra-nets, i.e., Voice over IP (VoIP). Several popular Internet implementations are the Video Audio Tool (VAT) [1] and the Robust Audio Tool (RAT) [2], as well as a host of ITU-T H.323 implementations. An important aspect of VoIP is developing a performance monitoring capability to track the quality of the voice transport. In this paper, we discuss one approach to monitoring the performance of conversational voice applications over Internet transport. Specifically, we investigate the use of the ITU-T's E-Model[3] as a tool to relate several transport level metrics to an estimate of conversational voice quality. To accomplish this, we analyze the reduction of the existing E-Model in terms of transport-level metrics for the purpose of monitoring conversational voice quality. In the process, we discuss the advantages and shortcomings of our approach and identify a set of issues which we believe need to be addressed within the open literature.

The ITU-T's E-Model is a network planning tool used in the design of hybrid circuit-switched and packet-switched networks for carrying high quality voice applications. The tool estimates the relative impairments to voice quality when comparing different network equipment and network designs. The tool provides a means to estimate the subjective Mean Opinion Score (MOS) rating of voice quality over these planned network environments. We describe the E-Model in more detail in Section 3 below.

The specific method we advocate is to:

- Measure the low-level transport metrics (characterizing the channel), which impact voice performance, i.e., delay, delay variation and packet loss,
- Combine the packet loss and delay variation measurements, de-jitter buffer operations, packet size and coder frame size into an error mask (the exact sequence of good and bad coder frames) that can be characterized in a simple manner (e.g., average frame loss rate along with some measure of burstiness),
- Combine the characterized error mask with the coder and its frame-loss concealment algorithm via a look-up table (or curve fit) based on

subjective testing to produce an E-Model equipment impairment factor (I_{ef}), and

- Combine the I_{ef} with other E-Model low-level measurable elements, i.e., delay and echo, to produce a predicted opinion score on the quality of the voice conversation.

We illustrate this measurement and data reduction methodology in Figure 1 below. In this figure we capture the channel characteristics via a set of transport level measurements, e.g., packet loss and delay distributions. We combine this with various architectural characteristics of the VoIP gateways, specifically the de-jitter buffer implementations, the transport packet size and the codec frame size.

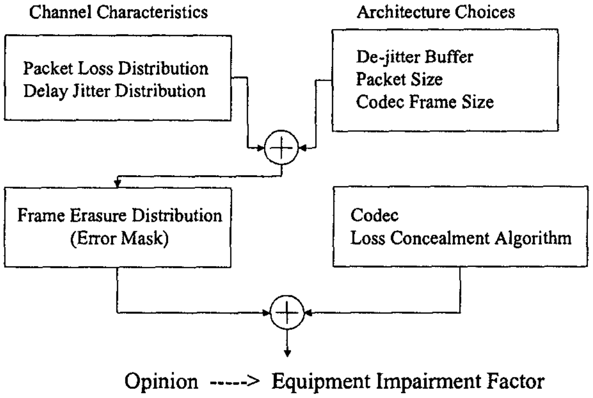

Figure 1: A measurement and data reduction methodology for VoIP quality monitoring, which highlights the equipment impairment factor elements.

The result of combining the channel characterization and the architectural characterization of the gateways is a *Frame Erasure Distribution* (or *Error Mask*). The Error Mask characterizes the salient features of the loss distribution as observed by the decoder. (Note: This loss distribution captures both the transport packet loss and the loss in the decoder's de-jitter buffer[1] due to late packet arrivals.) When the Error Mask is combined with the specific loss concealment algorithm implemented within codec, we generate an Equipment Impairment Factor, which captures the expected impairment of the codec under the above conditions. From this, the E-Model provides a means to estimate a quality score for the conversational voice application. We discuss the methodology in detail in Section 4 below.

Following this methodology, it is possible to build a relatively simple VoIP performance monitoring capability. It is a further goal of this paper to help foster an industry consensus around the specific methodology to follow in developing such a VoIP performance monitoring capability. Only then, would it be possible to obtain consistent quality estimates from this type of VoIP quality monitor.

The remainder of this paper is organized as follows: We next discuss the relationship of the E-Model approach we are advocating to other methods of monitoring VoIP performance. In Section 3, we present an overview of the ITU-T's E-Model. Section 4 covers our efforts to reduce the E-Model's formulae to transport-level, directly measurable quantities in an unambiguous fashion. Section 5 discusses several issues with this reduction and, in particular, discusses the issue of identifying a 'reference model' for performance monitoring. We follow this with a discussion of measurement methodologies and report on some field-work we have performed with this approach. We finish the paper with a section on our conclusions.

2 RELATIONSHIP TO OTHER METHODS

We know of a few commercial products, which implement monitoring approaches similar to the one we advocate within this paper. Also, the approach we advocate is not the only approach to monitoring the quality of VoIP applications. Other approaches range from "objective models of quality", involving the injection of sample speech segments across the network, to simple packet level measurements. In this section we discuss these alternatives.

We have run across several commercial products, which monitor VoIP quality in a fashion similar to the approach we advocate. These include the Cisco voice dial control MIB [4] and Telchemy's monitoring software [5]. The information on these products refers to the E-Model in the description of their approach. However, both of these products appear to rely on information extracted from within the voice gateways. As such, they require implementation within the VoIP gateways themselves. In this paper,

[1] The decoder must intentionally delay the variable delayed, arriving voice packets in its de-jitter buffer in order to reconstruct a synchronous bit stream. In some cases this de-jitter buffer delay is not large enough to absorb the transport delay variation. This results in de-jitter buffer losses as observed by the decoder.

we discuss a generalized approach, which is independent of the implementation location of the monitors. As such, we attempt to highlight the relative advantages of each approach.

An alternative method, commonly in use, requires the injection of sample speech segments across a voice transport path, i.e., the coder-to-network-to-decoder path. This method is often referred to as "objective models of quality". The models compare the output speech to the input speech, using psycho-acoustic fundamentals, to produce opinion without reference to the underlying channel conditions. However, it is still necessary to overlay the low-level transport measurements of delay and echo on top of this, using the E-Model, in order to capture the conversational speech impairments due to delay. The ITU-T has standardized one such model [6] and continues to investigate improved models [7].

The advantages of objective models of quality are: 1) no knowledge of, or assumptions about, the underlying network is required (coder, de-jitter buffer, error mask, packet size), and 2) predicted opinion is based on fundamental psycho-acoustics rather than an interpolation of subjective testing results as with the E-Model, and thus the results may be more accurate and more robust. With regard to accuracy, the E-Model was intended to be used as a network planning tool, not a network maintenance tool. As such, it only needs to be accurate enough to distinguish between broad ranges of quality (see Table 1). On the other hand, objective models of quality can often distinguish between quality levels within a broad range. With regard to robustness, the E-Model cannot predict opinion for conditions that have not been previously scored by subjective panels. On the other hand, objective models of quality can predict opinion for such conditions, although their accuracy in such cases is not always good.

The disadvantages of objective models are: 1) they are complex and costly, 2) there are some conditions for which they are known not to be accurate (e.g., temporal clipping), 3) they are intrusive whereas the E-Model can be implemented as either intrusive or non-intrusive, and 4) they reveal nothing about the underlying cause of quality problems. Because we make low-level measurements with the E-Model, we have causality information.

Another method is to rely on direct packet level measurements or straightforward combinations of packet level measurements. Thresholds are then defined as to when the quality of voice conversations would degrade to a critical point. The advantage of this approach is that it is relatively simple to implement. Its disadvantage is that the thresholds it relies upon are somewhat arbitrarily chosen. Further, this approach does not attempt to combine the transport metrics in a meaningful way with respect to voice quality.

3 THE E-MODEL: AN END-TO-END VOICE QUALITY MODEL

The E-Model, defined in the ITU-T Rec. G.107 [3] as well as other associated ITU-T recommendations [8], is an analytic model of voice quality used for network planning purposes. Specifically, the E-Model presents a method for estimating the relative voice quality when comparing two reference connections [3]. According to [3], the E-Model has proven useful as a transmission planning tool, however further studies are underway to address the assumptions of the E-Model under specific parameter combinations. For a fuller discussion of the validity of the E-Model, refer to [3].

A basic result of the E-Model is the calculation of the R-factor, which is a simple measure of voice quality ranging from a best case of 100 to a worst case of 0. The R-factor uniquely determines the familiar Mean Opinion Score (MOS), which is the arithmetic average of opinion when "excellent" quality is given a score of 5, "good" a 4, "fair" a 3, "poor" a 2, and "bad" a 1. The R-factor is defined in terms of several parameters associated with a voice channel across a mixed Switched Circuit Network (SCN) and a Packet Switched Network (PSN). The parameters included in the computation of the R-factor are fairly extensive covering such factors as echo, background noise, signal loss, codec impairments, and others. An excellent discussion of the E-Model is found in [9].

The R-factor is related to the MOS through the following set of expressions:

$$\begin{aligned} &\textit{For } R < 0 : MOS = 1 \\ &\textit{For } R > 100 : MOS = 4.5 \qquad \textit{(Equation 1)} \\ &\textit{For } 0 < R < 100 : MOS = 1 + 0.035\,R \\ &\qquad + 7x10^{-6}\,R(R-60)(100-R) \end{aligned}$$

For reference, Eq.(1) is plotted in Figure 2.

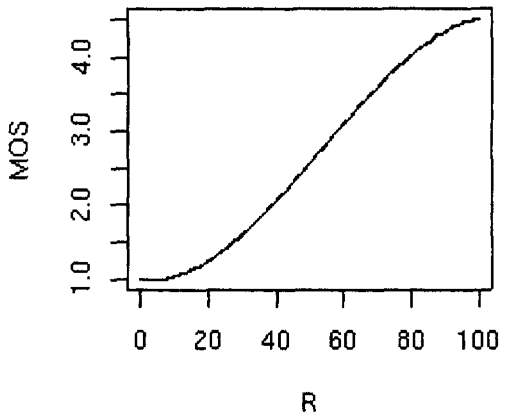

Figure 2: A plot showing the relationship between the R-factor and the MOS (see Eq.(1)).

Typically, the values of the R-factor are categorized as shown in Table 1 below. Here we see that connections with R-factors of less than 60 are expected to provide a 'poor' quality of service to users.

Table 1: R-factors, quality ratings and the associated MOS.

R-factor	Quality of voice rating	MOS
90 < R < 100	Best	4.34 – 4.5
80 < R < 90	High	4.03 – 4.34
70 < R < 80	Medium	3.60 – 4.03
60 < R < 70	Low	3.10 – 3.60
50 < R < 60	Poor	2.58 – 3.10

The R-factor is expressed as the sum of four terms:

$$R = 100 - I_s - I_d - I_{ef} + A \qquad \textit{(Equation 2)}$$

where I_s is the signal-to-noise impairments associated with typical SCN paths, I_d is the impairment associated with the mouth-to-ear delay of the path, I_{ef} is an equipment impairment factor associated with the losses within the gateway codecs and A is the *Expectation Factor.* An interesting aspect of the E-Model is that these terms, i.e., I_s, I_d, and I_{ef} are additive and further, that the delay and packet loss contributions are isolated into I_d and I_{ef} , respectively. This does not imply that delay and packet loss are un-correlated in the underlying transport media, but only that their contributions to the estimated impairments are separable.

The Expectation Factor covers those intangible quantities that are difficult (or impossible) to quantify. This term accounts for lowered customer expectations of quality because of, e.g., a cell phone user's tendency to tolerate lower quality in exchange for the convenience afforded by mobility, or in exchange for a lower price. For the most part it is difficult to estimate the Expectation Factor, although there appears to be some agreement that an Expectation Factor of around 10 for a cellular network is appropriate [10]. However, no such agreement has been reached for the case of lower prices as expected with some VoIP services. For this reason, we will drop the Expectation Factor from our future discussions of the R-factor.

The signal-to-noise impairment factor, I_s, is a function of several parameters, none of which are a function of the underlying packet transport. However, the ITU-T Rec. G.107 [3] recommends a set of default values for these parameters for planning purposes. Because this is not the focus of our discussion, and is dependent upon the method to access the VoIP network, we will rely upon the default recommendations for all but a few parameters, e.g., all except for the delay and packet loss parameters. For example, it is sufficient for our purposes to assume that echo cancelers are present and working properly (no echo). Choosing these default values, we can reduce the expression for the R-factor [3] to:

$$R = 94.2 - I_d - I_{ef} \qquad \textit{(Equation 3)}$$

Not only have we chosen the default values for the various SCN signal impairments, but we have also dropped reference to the Expectation Factor.

The delay components within the function I_d are: 1) T_a the average, absolute one-way mouth-to-ear delay, 2) T the average, one-way delay from the receive side to the point in the end-to-end path where a signal coupling occurs as a source of echo, and 3) T_r the average, round trip delay in the four-wire loop. Note that T_a, T_r and T represent various measures of delay from different points within a general reference connection. G.107 gives a fully analytical expression for the function I_d, in terms of T_a, T, T_r and parameters associated with a general reference connection describing various circuit switched and packet switch inter-working scenarios.

Table 2: Values of the delay impairment for selected, one-way delay values. (Note: The one-way delay is defined as mouth-to-ear delay.)

One-way delay (msec)	I_d
0	0
25	0.9
50	1.5
75	2.1
100	2.6
125	3.1
150	3.7
175	5.0
200	7.4
225	10.6
250	14.1
275	17.4
300	20.6
325	23.5
350	26.2
375	28.7
400	31.0

Since the focus of this paper is on the development of an IP-based monitoring system, we choose to simplify the expression for I_d in three ways (and hence focus our discussion on IP-based transport and VoIP gateway issues). First, for the cases we are interested in, i.e., VoIP with no circuit switched network interworking, the various measurement points for the delay measures collapse into a single pair of points, such that

$$d = T_a = T = T_r / 2 \qquad \textit{(Equation 4)}$$

and that $I_d(d)$ is now a function only of the single delay measurement d. Second, we choose to use the default values listed in [3] for all terms in the I_d expression other than T_a, T and T_r. Third, we plot out the delay component and then fit the resulting curve to a simple expression for discussion purposes. For reference, the full expression for I_d, assuming only the default values listed in G.107, could be used for our purposes instead of our simplified expression derived below. But, we find it much more convenient for discussion and modeling purposes to use our simplified expression below. Table 2 above gives the values for the delay component for selected values of the one way delay [11].

In Figure 3, we plot these values and find that I_d has two roughly linear regions. A knee in the curve occurs at a delay of 177 msec. For one way delays less than 177 msec, conversations occur naturally, whereas at delays in excess of 177 msec conversations begin to strain and breakdown; often degenerating into simplex communications at high delay values.

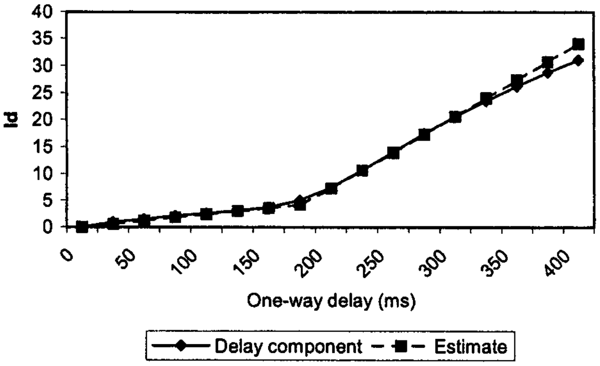

Figure 3: A plot of the I_d as a function of delay along with a simple fit.

Also on this plot, we fit the values of I_d to the expression:

$$I_d = 0.024\, d\ + 0.11(d - 177.3)\, H(d - 177.3) \qquad \textit{(Equation 5)}$$

Here *d* is the one-way delay (in milliseconds) and H(x) is the Heavyside (or step) function:

$$H(x) = 0 \quad \textit{if } x < 0, \textit{ else}$$
$$H(x) = 1 \quad \textit{for } x >= 0 \qquad \textit{(Equation 6)}$$

We can now express the R-factor in the form:

$$R \sim 94.2 - 0.024d + 0.11(d - 177.3)H(d - 177.3) - I_{ef}$$

(Equation 7)

All that remains is to find suitable estimates for the equipment impairment factors. Currently, no analytic expressions exist for the equipment impairment factors. Estimates must be extracted from subjective measurements. We address this in the next section.

4 EQUIPMENT IMPAIRMENT FACTORS AND THEIR REDUCTION (THOUGH MEASUREMENTS) TO BASE TRANSPORT METRICS

Figure 1 depicts how various elements lead to equipment impairment factors. Firstly, we have the channel packet loss and delay jitter characteristics. Secondly, the de-jitter buffer will smooth out the delay variation, at the expense of increased packet loss and delay. Thirdly, the resulting packet loss distribution is combined with the relative size of the packets and coder frames to produce an error mask (the exact sequence of good and bad coder frames). Fourthly, the error mask is combined with the coder and it's frame loss concealment algorithm to produce opinion. Lastly, opinion is converted to an equipment impairment factor. No analytic expressions are directly available for the I_{ef}'s. Instead, the I_{ef} must be obtained from subjective measurements of voice quality for each particular codec and the various operating conditions shown in Figure 1, e.g., the packet loss, packet size, etc.

What we require at this point, are studies of the quality of the various codec implementations for the expected operating conditions. As an example, Hardman, et. al. [12] and [13] have performed MOS and intelligibility tests for a novel loss concealment scheme for consideration within their Robust Audio Tool (RAT). For simplicity of discussion, we instead work with the results found in Appendix I in [8]. Here various I_{ef} values are directly given for several codec types as a function of average packet loss rate, packet loss burstiness and packet size based upon MOS testing. One such example is the G.729a coder [14], with a packet size of 20 msecs, and random packet loss of up to 16%. The reported results for the I_{ef} factor are plotted in Figure 4. The effect of packet loss is to increase the measured value of I_{ef}, and hence decrease the call quality, as one would expect. The minimum I_{ef} value is 11 for the case of zero packet loss. This is the measured impairment due to the compression algorithm within the G.729a standard. The measured I_{ef} is monotonically increasing as the average packet loss is increased. The largest measured value reported in [8] is 49 for the case of an average packet loss of 16%.

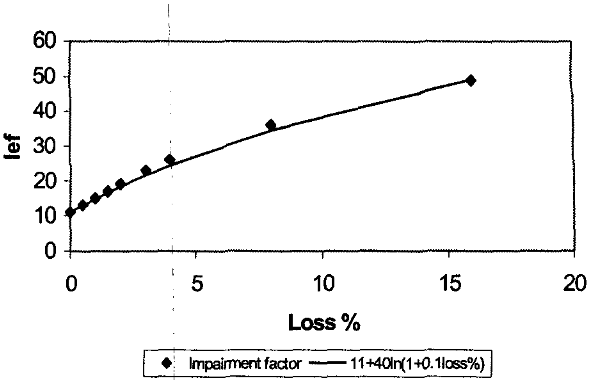

Figure 4: Measurements of I_{ef} for a G.729a codec under random packet loss conditions.

Also shown in Figure 4 is a curve fit of the measured values of I_{ef}. The curve is derived from an expression of the form:

$$I_{ef} \sim \gamma_1 + \gamma_2 \ln(1+\gamma_3 e) \qquad \textit{(Equation 8)}$$

where e is the total loss probability (i.e., e lies between 0 and 1) and the γ_i's are fitting parameters. In general, we have found that an expression of this form fits well the measured data on I_{ef} values for various codecs under conditions of random packet loss. As an example, the specific values used in Figure 4 are

$$I_{ef}(G.729a,\ random) \sim 11 + 40\ ln(1 + 10\ e) \qquad \textit{(Equation 9)}$$

Appendix I in [8] also gives measured values for the I_{ef} factor for the G.711 codec [15] implementing the packet loss concealment algorithm found in [16] in the presence of random and also bursty packet loss. We show these measured values in Figure 5. From the figure we see that at zero loss, the I_{ef} value is zero. This is a reflection of the fact that the G.711 codec adds no impairment to the observed quality of the speech in zero loss conditions. We also see that for low loss, i.e., e < 0.04, the measured values of I_{ef} (G.711 with packet loss concealment) under conditions of random and bursty packet loss are identical (within the accuracy of the measurement methodology). However, for larger loss values, i.e., e > 0.04, they differ dramatically. The bursty loss causes a marked degradation in the voice quality over the random loss measurements at comparable average loss rates. We also show the two curve fits we performed to the data. The lower curve was fit to the random packet loss data and is specifically given by the expression:

$$I_{ef}(G.711\,concealment,\ random) \sim 0 + 30\ ln(1+15\ e)$$

(Equation 10)

The upper curve was fit to the bursty packet loss data for loss values greater than 4%. The specific values used for the curve fit are given by:

$$I_{ef}(G.711\,concealment,\ bursty>4\%) \sim 0+19 ln(1+70\ e)$$

(Equation 11)

Therefore, our best estimate for the I_{ef} factor over the range of bursty loss values (combining the two previous expressions) is

$$I_{ef}(G.711\ concealment,\ bursty) \sim 30 ln(1+15\ e)\ H(0.04-e) + 19 ln(1+70e)\ H(e-0.04)$$

(Equation 12)

The G.711 data from [8] for bursty loss conditions shows almost a jump discontinuity between the measurements at e = 0.03 and e = 0.05. The reason for this is not apparent to us at this time; we can only speculate as to the underlying cause. Missing from [8] is a description of the algorithm used for generating the bursty packet losses. We also note that the discontinuity may be real or it may be an artifact of the measurement methodology, which is also not described in [8]. In the event that the discontinuity is real, it may be caused by the performance of the loss concealment algorithm [16] in the presence of bursty errors. This particular loss concealment algorithm performs extremely well in concealing single (isolated) packet losses, but hits a "cliff" for some number of consecutive packet losses. Beyond this point, its performance is comparable to other codecs and loss concealment algorithms. It maybe that at 3% or less packet loss (given the nature of the simulated loss patterns generated for the tests), the total loss rate is low enough that the tests do not often generate the "critical" number of consecutive losses. But, at 5% and higher, the tests do. Other coders and loss concealment algorithms may not have this discontinuity because they do not exhibit the same exceptional performance for single packet losses as the concealment algorithm described in [16]. We show these results only for illustrative purposes. However, we strongly believe that more measurement work in this area is warranted.

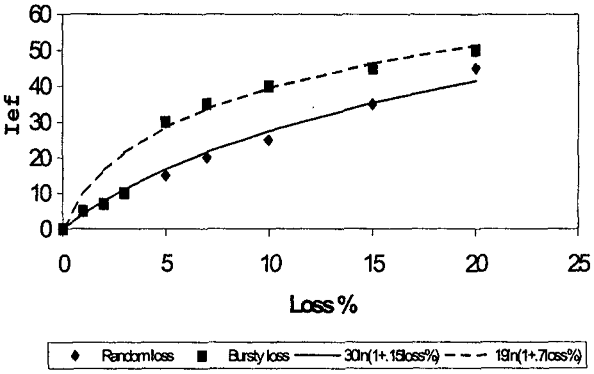

Figure 5: Measurements on bursty packet loss effects on I_{ef} for a G.711 codec with packet loss concealment (from Ref. [8]).

4.1 A FULLY ANALYTIC EXPRESSION FOR THE R-FACTOR

Ideally, we would like to be able to express the I_{ef} values for the various and interesting codecs in fully analytic form as a function of the average packet loss and some appropriate measure of loss burstiness. However, at this point in time not enough subjective measurements and their specifics are available in the open literature to accomplish this. The best we can currently do is to focus on the results from the measurements discussed above and draw some conclusions. For discussion purposes, we will limit our examples to the results for random packet loss measurements.

We are now in a position to summarize the results of the measurements and the curve fitting by writing down our relatively simple expression for the R-factor for various codec types. Based upon the discussion of this section and the previous one, we find for the R-factor the following expression (for random packet loss):

$$R \sim \alpha-\beta_1\ d-\beta_2\ (d-\beta_3)\ H(d-\beta_3) -\gamma_1 -\gamma_2\ ln(1+\gamma_3 e)$$

(Equation 13)

where

$\alpha = 94.2$

$\beta_1 = 0.024\ ms^{-1}$ $\beta_2 = 0.11\ ms^{-1}$ $\beta_3 = 177.3\ ms$

$\gamma_1 = 11\ (G.729a)$ $\gamma_2 = 40\ (G.729a)$ $\gamma_3 = 10\ (G.729a)$

$\gamma_1 = 0$ (G.711) $\gamma_2 = 30$ (G.711) $\gamma_3 = 15$ (G.711)

$$d = d_{codec} + d_{de\text{-}jitter\_buffer} + d_{network}$$

$$e = e_{network} + (1 - e_{network})\, e_{jitter\_buffer}$$

The results for the G.729a codec assumes a 20 msec packet size, while the G.711 results are for a 10 msec packet size. The results for both G.729a and G.711 listed above are limited to random packet loss. Other values need to be derived for other combinations of coders, packet size and error mask distributions.

In Equation (13), d represents the one way delay from mouth-to-ear. As we will see later in the discussion of the codec component models, d is composed of d_{codec}, the algorithmic and packetization delay associated with the codec and the IP packet processor, $d_{de\text{-}jitter\_buffer}$, the delay associated with the de-jitter buffer required to smooth out the delay variation in the arriving packet stream, and $d_{network}$, the one way transit delay across the IP transport network from gateway to gateway. Similarly, the loss probability is decomposed into a sum of two terms, the $e_{network}$, which is the loss probability due to the loss in the IP transport network, and $e_{de\text{-}jitter\_buffer}$, which is the loss due to the arriving packet stream underflowing or overflowing the decoder's de-jitter buffer.

These results are central to the theme of this paper. They will form the basis for the remainder of the discussion below. Before we go on, we highlight the assumptions and deficiencies in our above estimations:

- First, the volume of subjective test data is small. In the example of the G.711 codec, we were working with subjective measurements for only 8 different operating parameter settings. Clearly, more data is necessary to refine the curve fitting for the impairment factors of the various codec types.

- Second, the nature of the appropriate error masks is uncertain. There are a few examples in the literature where measurements of the packet loss probability over Internet paths have been made [17] and [18]. This body of work focuses on network induced packet loss. But we require some means to account for de-jitter buffer loss due to network transport delay variation in the error masks as well. Further work is required to reach consensus on the appropriate error masks to use for the subjective speech quality measurements. This implies a better understanding of packet delay variation across packet networks.

4.2 FURTHER REDUCTION TO TRANSPORT LEVEL METRICS

Equation (13) gives us a fully analytic expression for the R-factor in terms of total delay and packet loss. We directly measure the transport level metrics that contribute to total delay and loss, i.e., the one-way transport delay ($d_{network}$) and the network packet loss ($e_{network}$). The other metrics, required to complete the formulation of R, are d_{codec}, $d_{de\text{-}jitter\_buffer}$ and $e_{de\text{-}jitter\_buffer}$. As seen in Table 3 below, the delay attributed to encoding, compression and packetization delays are known quantities of the specific codec and IP packetization implementations. Unfortunately, the remaining two quantities, $d_{de\text{-}jitter\_buffer}$ and $e_{de\text{-}jitter\_buffer}$, are related to the design, implementation and performance of the de-jitter buffer at the receiver-side codec. They are also tightly coupled to the delay variation within the transport network. We now discuss in more detail these issues.

4.2.1 Delay Components within the Codec

First, we discuss the various delay contributions to the ear-to-mouth delay. In Equation (13) we break this quantity into three separate terms, $d = d_{codec} + d_{de\text{-}jitter\_buffer} + d_{network}$. In this section, we detail the calculation of d_{codec}. The next section details the estimation of $d_{de\text{-}jitter\_buffer}$ along with the de-jitter buffer's contribution to packet loss.

In Figure 6, we show a timing diagram for the case of a pair of G.729a codecs and a typical IP packet processor. Here it is assumed that voice is streaming from the left-hand side of the diagram to the right-hand side. Time is assumed to progress downward in the diagram. Packet insertion delays are indicated by the height of the packets and network propagation delay is accounted for in the downward slope of the lines that connect the left-hand side to the right-hand side. For the G.729a encoder, the PCM encoded voice stream is handed off in 10 msec blocks to the encoder. G.729a uses a 5 msec look ahead in order to encode the current 10 msec PCM encoded block, so the timing diagram shows a further 5 msec delay before encoding begins.

From Figure 6, we can easily extract d_{codec}. We see that for a G.729a codec, the best case estimate is 5msec + N x 10 msec, where N is the number of 10 msec voice frames packed into a single IP packet. Since it is assumed that N = 2 in Figure 6, the total codec encoding delay is 25 msec. We stated that this is best case because we have neglected all processing delays. For software based implementations, the processing delays may be significant. However, without access to specific implementation information and processing delay measurements, we have no way to estimate software-based codec processing delays. In general, however, we expect these delays to be on the order of a few msec given that most modern codecs are implemented in hardware.

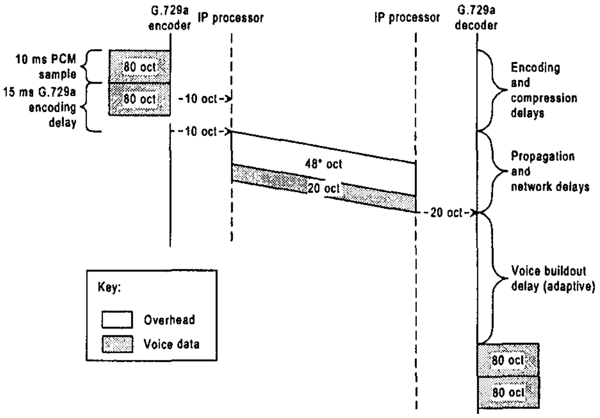

Figure 6: Timing diagram for the G.729a codec with a typical IP packet processor.

Similar timing diagrams can be developed for other codecs and their associated encoder delays can be estimated. In Table 3 we summarize the results for the value of d_{codec} for G.711 and G.729a codecs under the assumption that the IP processor acts on 10 msec units of speech.

Table 3: Estimated d_{codec} values for various codec types (N = the number of 10 msec frames per IP packet).

Codec type	Encoding delay estimate (msec)
G.711	10 x N
G.729a	5 + 10 x N

4.2.2 *Reduction of the De-jitter Buffer Delay and Loss Contributions*

The previous section evaluated the d_{codec} contribution to the delay. Other than the $d_{network}$ delay and $e_{network}$ loss, which are measured transport level quantities, we are left to deal with the $d_{de\text{-}jitter}$ contribution to the ear-to-mouth delay and the $e_{de\text{-}jitter}$ contribution to the total loss.

It turns out that given a specific de-jitter buffer implementation and network delay behavior, there exists a relationship between $d_{de\text{-}jitter}$, $e_{de\text{-}jitter}$ and $v_{network}$, the network delay variation. Therefore, $d_{de\text{-}jitter}$ and $e_{de\text{-}jitter}$ should not be discussed separately. However, due to the complexities associated with delay characteristics of packet networks and the range of implementation possibilities for the de-jitter buffer algorithms, the exact form of the above relationship is unknown. For a discussion of the complexities of modeling (or characterizing) de-jitter buffer implementations, see [19] and [20].

In the case where the monitoring points are placed within the voice gateways, it is conceivable that the monitors will have direct access to these quantities, i.e., $d_{de\text{-}jitter}$ and $e_{de\text{-}jitter}$. An example of this approach is discussed in [21]. For more general placement of monitoring points, these quantities must be inferred from the underlying transport measurements, e.g., $v_{network}$, and models of the de-jitter buffer behavior. We now discuss this latter approach and refer the reader to [21] for a discussion of the former approach.

For illustrative purposes, let us assume a simple (but common) model of the de-jitter buffer behavior and network delay characteristics. Then, from these assumptions we will derive a relationship between $d_{de\text{-}jitter}$, $e_{de\text{-}jitter}$ and $v_{network}$. From this expression, we will highlight several issues to resolve in order to develop a more general VoIP performance monitoring capability as we are suggesting.

Let us assume for the moment that a simple, static de-jitter buffer algorithm is implemented in the receiver-side decoder. In this example, we assume that the first packet (within a given talk spurt) is buffered until the *bth* and *bth+1* packets are received. Upon the receipt of the *bth+1* voice packet, the decoder will begin playing out the speech samples from the first voice packet. Typically, the total buffer size is given in average delay terms as *2bg* where *2b* is the total number of voice packets that could be buffered within the playout, and g is the mean interpacket arrival period to the de-jitter buffer. So, in this example, the first packet of a talk spurt is played out when the de-jitter buffer reaches half occupancy.

Let

- d_i = one-way delay of the ith packet in a talk spurt, and
- $l_i = d_{i+1} - d_i$

Let us further assume the following:

- A talk spurt consists of N voice packets, where N is large.
- The d_i's are assumed independent. The independence assumption is key to the result to follow. It is known that, in general, this assumption is not valid. However, the discussion here is meant only to illustrate several issues. It is not meant as a proposed method of modeling the de-jitter buffer behavior.
- $E(l_i) = g$, this implies that there is no loss within the transport network and hence no bandwidth limits.
- For the buffer model (discussed above), the late packets contribute to the de-jitter buffer loss but early packets do not, i.e., the implementation will buffer early packets. Not all implementations can dynamically allocate additional buffers for early packets, so this is an assumption. However, we would in general suspect that the majority of losses in a de-jitter buffer occur due to late (and not early) packets.

The loss due to the de-jitter buffer can be generally expressed as

$$e_{dejitter} = \left(\frac{1}{N}\right)\sum_{i=b+2}^{N} P\{d_i - d_{b+1} > bg\}$$

(Equation 14)

Figure 7 shows an idealization of the probability density function for the inter-packet arrival times, i.e., the l_i's.

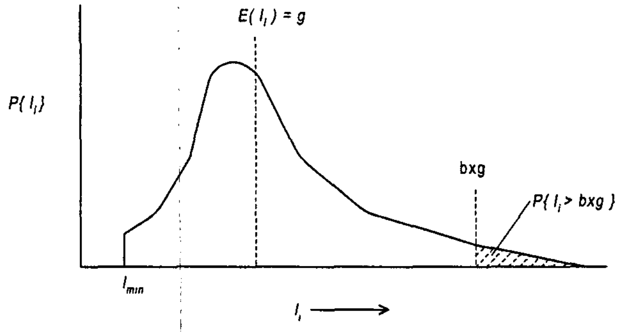

Figure 7: An idealized probability density for the interpacket arrival times.

Given our set of assumptions listed above, we can reduce the expression for the loss in the de-jitter buffer as follows:

$$e_{de\text{-}jitter\_buffer} \sim (N\text{-}b\text{-}1/N)P\{d_{b+2} - d_{b+1} > bg\}$$

$$\sim (N\text{-}b\text{-}1/N)P\{l_{b+1} > bg\}$$

$$\sim P\{l > bg\} \qquad \textit{(Equation 15)}$$

If the product bg is known a priori, then a measurement probe could directly measure the above quantity. However, it is reasonable to expect that a given Internet path would support a range of codec types, implementations and configurations. A given probe is not able to determine, by monitoring the protocol exchanges between the codec end-points, the de-jitter buffer size $(2bg)$. This information is not contained in the initial protocol setup and negotiation. Therefore, it is not reasonable for a probe to extract this information in most situations (unless it is co-located with the voice gateway).

Further, there exist adaptive de-jitter buffer implementations for which a fixed buffer size does not exist. In the case of an adaptive de-jitter buffer, the algorithm attempts to control the loss probability at the de-jitter by dynamically adjusting the playout delay of the de-jitter buffer, i.e., bg. Hence, the loss probability is roughly fixed, but the delay variation in the transport network causes a variable de-jitter buffer delay. This is in contrast to a static de-jitter buffer, which fixes the buffer size, and hence the de-jitter buffer delay, at the expense of a variable de-jitter buffer loss rate due to the variability of the network transport delays.

A simple and fairly conservative relationship between the loss rate, the buffer size and the network transport delay variation for our simple model can be derived from Chebyshev's inequality [22]:

$$P\{\ |X-\mu|>t\ \}\ <\ v\ /\ t^{2} \qquad \textit{(Equation 16)}$$

Where $\mu = E(X)$, $v = Var(X)$. The inequality holds for general distributions, given the existence of the variance. Then, rewriting Eq.(15) for the de-jitter loss probability, we get

$$\begin{aligned} e_{de\text{-}jitter\_buffer} &\sim P\{\ l > bg\ \} \\ &\sim P\{\ l - g > bg - g\ \} \\ &\sim P\{\ |\ l - g\ | > bg - g\ \} \end{aligned}$$

$$e_{de\text{-}jitter\_buffer} <\ v(l)\ /\ (\ bg-g\ )^{2} \qquad \textit{(Equation 17)}$$

or dividing through by g on the RHS, and re-arranging, we get

$$(\ b-1\ )\ e^{1/2}{}_{de\_jiitter}\ <\ \sigma_l\ /g \qquad \textit{(Equation 18)}$$

where σ_l is the standard deviation of l. This is an extremely simple expression relating the key quantities of interest with respect to the de-jitter buffer performance. This expression sheds light on the tradeoff of increasing the mouth-to-ear delay by increasing the de-jitter buffer depth versus decreasing delay and increasing the loss probability. Although, we originally derived this expression to obtain the loss probability as a function of delay variation and buffer delay, we can just as easily turn it around and interpret it as an expression for the buffer delay (in an adaptive scheme) in terms of delay variation and target loss probability.

For the case of a static de-jitter buffer, we could use Equation (18) to estimate the de-jitter buffer loss by measuring the variation in the inter-packet delay, given the buffer depth. Alternatively, we could directly measure the de-jitter buffer loss through Equation (15). Because delay variation is very unpredictable on a given path and ranges greatly from path to path, it is unlikely that a static de-jitter buffer implementation is robust. It would have to be very large to cover the range of conditions it is expected to operate in, but this would cause large mouth-to-ear delays. Static de-jitter buffers may perform well only when the transport network is over-engineered or the voice packet transport were QoS enabled.

For this reason, it is more likely that adaptive de-jitter buffers will be relied upon. Several studies have recommended the use of adaptive de-jitter buffer implementations and studied their performance, for example [19], [20] and [23]. An adaptive scheme would attempt to control the de-jitter loss within a narrow operating range by adapting the mouth-to-ear delay through the depth of the de-jitter buffer. In this case, given the target loss probability that the de-jitter buffer is designed for, it is possible to measure the delay variation in the transport network and to estimate the de-jitter buffer delay contribution using the relationship in Equation (18).

5 Discussion of a Standard Reference Implementation for Consistent Evaluation

In either case, a static or an adaptive buffer implementation, information regarding the design of the de-jitter buffer is required in order to design a VoIP quality monitoring capability. This aspect of the decoder design is implementation specific and not addressed within any codec standard. If the monitor is implemented in the egress gateway, then this implementation information is potentially available. However, there are applications requiring a monitor to be physically distinct from the gateway, in which case this missing information becomes problematic.

One solution to this issue is for the industry to agree to a reference implementation for performance monitoring purposes. This would allow various implementations of a monitor to generate consistent results. Although the results from the monitor would not reflect the actual end-user experience in this case (because their de-jitter buffer implementations would differ from the reference implementation), at least the results would be consistent across various implementations of the monitor. Further, the results from the monitor would have a relative significance when evaluating the performance of the packet transport network.

In both cases, static and adaptive, if a reference implementation were agreed to, then a consistent and reproducible monitoring capability is possible. In a relatively straight-forward approach, the monitoring device could simply run the reference algorithm on the observed packet stream and compute $e_{de\text{-}jitter}$ and $d_{de\text{-}jitter}$ due to the accumulated delay variation in the path up to the location of the monitor. The reference algorithm for a static buffer would be fairly simple to define and execute. An adaptive reference algorithm would be somewhat more complex to define and execute.

6 MEASUREMENT METHODOLOGY

Once a consistent and generally accepted set of parameters characterizing the I_{ef}'s is developed, it is possible to build a performance monitoring capability characterizing VoIP quality. The method of monitoring can be categorized as *passive monitoring* or *active monitoring*. By passive monitoring, we mean a capability, which listens to traffic flowing across a wire and measures various transport metrics of interest. By active monitoring, we mean a capability, which actively injects test (or probe) packets into the network for the primary purpose of measuring various transport metrics of interest. There are distinct advantages and disadvantages of both passive and active performance monitoring. These two approaches are very complimentary in nature. Passive probes are, by their very nature, non-intrusive; they add no additional load on the network or service. Passive monitors can provide a more extensive measurement capability (not only the type of measurements but also the amount of samples collected). However, passive monitors do not control the generation of data for the measurement samples. In contrast, active monitors are intrusive; they add load to the network or service. Because they control the generation of the packets, they can control the quality of the sampling and hence the quality of the sampled statistics. They also control the volume of traffic they introduce. In general, it is not expected that the objectives for generating active probes would necessitate high volumes of traffic.

Work is currently underway within the Remote Monitoring Management Information Base (RMONMIB) working group at the Internet Engineering Task Force (IETF) to define application level performance monitoring capabilities [24]. Initially this monitoring capability will provide a passive measurement capability. But the overall performance management architecture is being designed to allow for the control of synthetic traffic sources, hence active monitoring. Some initial work on synthetic sources within this architecture has already begun [24]. Potentially, VoIP quality monitoring as discussed within this paper could be incorporated within this application level performance monitoring capability in the future.

If this were to happen, then the placement of the monitors, as discussed above, would influence the algorithms used to evaluate the conversational voice quality. If the monitors are placed within the packet transport network, then some form of reference de-jitter buffer implementation has to be assumed. There is no means within the call establishment protocols for such a mid-stream monitoring to determine the actual implementation of the de-jitter buffers on either voice gateway. However, if the monitors were coincident with the de-jitter buffers, i.e., the monitors were located in the gateways, then the monitors could have access to the actual loss statistics from the de-jitter buffer, i.e., the monitor could get direct measurements of *e*, the total packet loss. This would more accurately capture the conversational quality as perceived by the end user.

There is value in both types of deployment. Users will want access to measurements, which capture as closely as possible their experience. In this case, the natural deployment location of the monitor is in the gateway device. On the other hand, service providers are going to want a capability to monitor the 'quality' of their transport services with respect to the transport of VoIP applications. Their networks will necessarily support a broad range of codec types and implementations outside of their control and view. In this case, the service providers will want to deploy monitors on the edge of their networks. Because a standard reference de-jitter buffer model must be supported in this case, their measurements will not reflect actual end-user experiences. However, there is much value in the relative measurement results as a method to track changes in the behavior of their networks and its relative importance on conversation voice performance.

7 AN IMPLEMENTATION AND FIELD MEASUREMENTS

We have implemented a VoIP performance monitoring capability as outlined in this paper. We refer to this monitoring tool as the WatchDog Internet Monitor. Our performance management application is written in PERL and PERL-TK for the graphical user interface. The PERL scripts write to the University of California – Davis' SNMP engine [25]. The PERL scripts configure active probes on remote Cisco routers through the Cisco enterprise rttMonMIB

[26]. For security reasons, we utilize the SNMPv3 capabilities of the UCD-SNMP engine and the Cisco IOS to strongly authenticate the SNMP_set commands used to configure the probes on the remote routers. The periodic SNMP_gets, issued by our tool to collect the data off the remote routers, are not authenticated. We have implemented our performance monitoring tool, scripts, and SNMP engine on a PC running Linux, Redhat version 6.1 [27].

The first version of the tool relied upon simple echo probes to estimate delay, delay variation and packet loss. The simple echo probes are relatively infrequently injected into the network, i.e., roughly once every 10 seconds. With the echo probes, we approximate the one-way transport delay measurement as one half the round trip delay and the one-way loss measurement as the loss probability of the echo probe. Obviously, the ability to perform true one-way measurements is desirable but not currently available within the implementation capabilities of our tool. Also, because we are sampling at a very infrequent rate, the accuracy of the delay variation measurement is questionable.

We have recently implemented a new capability relying on the 'jitter probe' defined within the Cisco enterprise rttMonMIB [26]. This probe can inject measurement packets into the network at a relatively high rate, e.g., once per 20 msecs. We are currently using this capability to emulate the injection of synthetic talk spurts, i.e., 20 packets of payload length 20 octets at 20 msec intervals similar to a G.729a codec packet injection process. We are using this 'heavyweight' probe to assess the difference in the reported delay variation measurements versus our earlier 'lightweight' probes based upon the echo probe capability in the Cisco enterprise rttMonMIB [26]. These results will be the subject of a future report.

A typical output from the WatchDog Internet Monitor is shown in Figure 8. This example is of a typical path spanning multiple, i.e., three, ISP networks and showing a moderate propagation delay. The figure shows measurements of the average delay and the standard deviation of the delay, measured over 5 minute intervals with 30 samples per interval. The delay statistics are plotted against the LHS of the plot and are in units of msecs. The loss probability and the R-factor estimates are plotted against the RHS of the plot. As expected, the R-factor is extremely sensitive to packet loss. These results are based upon our curve fitting for the G.729a codec in Section 4 above.

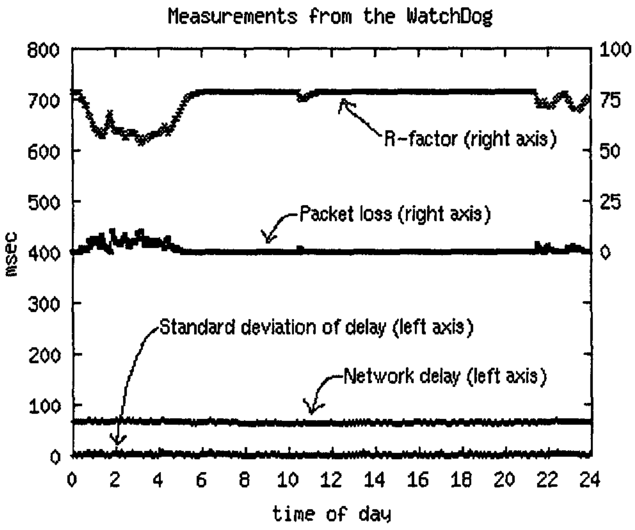

Figure 8: A daily plot from the WatchDog Internet Monitor showing delay, standard deviation of delay, loss and the estimated R-factor for a G.729a codec (time is GMT).

Let us illustrate an example of the calculations performed to estimate an R-factor in this case. Referring back to Equation (13), we see the estimated relationship between the R-factor, delay (*d*) and loss (*e*). The results following Equation (13) give us the values of the parameters characterizing the G.729a codec. Furthermore, these expressions relate the end-to-end measures of delay and loss to the component level measures of delay and loss. We get

$$R \sim 94.2 - 0.24\, d - 0.11(d-177.3)H(d-177.3) - 11 - 40\ln(1+10e) \qquad \text{(Equation 19)}$$

In this example, the measuring points are coincident with the codecs, i.e., the probes are generated from the same routers that contain the VoIP gateways. Thus, the difference between the end-to-end measurements and the transport level measurements are the delay and loss occurring within the gateways. Specifically, this is the encoding and packetization delays, the de-jitter buffer delays and the de-jitter buffer losses. From Section 4.2.1 above, we know that the encoding and packetization delays for a G.729a codec, which is generating a IP packet each 20 msec, is 25 msec. Further, in this example, we are assuming a static de-jitter buffer of 120 msec, i.e., we assume that the de-jitter buffer introduces another 60 msec delay on the end-to-end path. Thus we have that the end-to-end delay, *d*, is equal to the transport-level delay, $d_{network}$, plus 85 msec (the sum of the encoding and the de-jitter delays), i.e.

$$d = d_{network} + 85\ msec \qquad \text{(Equation 20)}$$

Inserting this expression into Equation (19), we get

$$\begin{aligned} R &\sim 94.2 - 0.024(d_{network} + 85) \\ &- 0.11(d_{network}-92.3)H(d_{network}-92.3) \\ &- 11\text{-}40ln[1+10(e_{network}+(1-e_{network})e_{de\text{-}jitter})] \end{aligned}$$

(Equation 21)

In this implementation of a measuring capability, we are injecting echo_probes to measure the round trip transport delays. Let RT_i, be the ith round trip delay measurement. Then, we use the following expressions to compute $d_{network}$, $e_{network}$ and $e_{de\text{-}jitter}$,

$$\begin{aligned} d_{network} &\sim \frac{1}{2}E(RT_i) = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{2}RT_i \\ e_{network} &\sim E(round\_trip\_loss) \\ &= \frac{1}{N}\sum_{i=1}^{N}H(RT_i - timeout) \\ e_{dejitter} &\sim \frac{1}{N}\sum_{i=5}^{N}H(RT_i - RT_4 - 3x20m\sec) \end{aligned}$$

(Equation 22)

Note, because we are making round trip measurements, we must infer one-way measurements (which of course we can not accurately do). So we have made the following choices. First, we take one half the round trip delay measurement to be the one way value. For loss, we take the one-way loss to equal the round trip loss (this over estimates the loss in any given direction, but is a conservative estimate). Clearly, the ability to make true one-way measurements is desirable.

So, assume that within our averaging interval, we obtain the following values

$d_{network}$ = 70 msec

$e_{network}$ = 0, and

$e_{de\text{-}jitter}$ = 0

We get for the R-factor from (Equation (21)), a value of 79. If instead we measured

$d_{network}$ = 80 msec

$e_{network}$ = 0.05, and

$e_{de\text{-}jitter}$ = 0.01

We get for the R-factor, a value of 60. Clearly the voice quality is extremely sensitive to loss.

We have been testing this monitoring capability in various VoIP services and trials at AT&T. We have found this capability to be extremely useful in several areas, specifically:

- troubleshooting paths – pings can be used to identify that connectivity exists, but additional capabilities are required to determine the quality of the connectivity,
- circuit pre-test and turn-up - prior to turning up a capability or customer, there is much value in monitoring the quality of their path or service prior to putting the customer on-line (without the capability of generating probe traffic this can be problematic),
- fault management - allows determination of whether the application is operating at a sufficient quality level or not,
- base-lining enhancements - active measurements are used to base-line before and after a certain set of QoS policies are applied,
- potential service level agreement (SLA) monitoring – monitors similar to these may be used to track the appropriate quality metrics for the transport service.

8 CONCLUSIONS

We have described a method for monitoring the quality of Internet paths to support conversational voice. This method involves the use of the E-Model to combine low level transport measures in a manner relevant to voice quality. We use the R-factor, an output of the E-Model directly related to MOS, which is commonly used to rate call quality. The method involves reducing the expression for the R-factor to transport level, measurable metrics. However, we show in the paper that there exist several ambiguities in the method. We propose that these require industry-wide discussions and participation in order to resolve. Specifically,

- Because the E-Model impairment factors are typically based on subjective testing using the C-language implementations found in the standards describing the codecs and not actual field implementations, our approach may produce estimates of voice quality, which differ from specific implementations.

- There is an industry-wide need for more extensive measurements of E-Model impairment factors. Specifically,

    - ➢ not enough subjective data exists covering the wide range of coders, loss concealment algorithms and error masks, and

    - ➢ the most important, or typical, sets of error masks are yet to be determined and characterized. More measurements of Internet loss and delay variation behavior are required to better characterize the transport channel in order to determine the appropriate set of error masks to test codecs against.

- There is no standard reference model for de-jitter buffer implementations. Various proprietary implementations of de-jitter buffer algorithms exist and they will differ in their impact on the error mask and hence the voice quality. A standard implementation (or multiple standards) is required for our method to generate unambiguous results.

Beyond these ambiguities, we discuss issues regarding the design and the placement of voice quality monitors. We discuss the relationship to other methods of monitoring voice quality. We relate this work to more general work in the industry to develop a more general performance monitoring capability [24].

Finally, we briefly present our efforts at building a prototype monitoring tool along the lines that we have proposed. This tool relies on a set of active probes between routers to generate transport level statistics. From these, the tool uses the E-Model to analyze the results and generate an R-factor estimate of anticipated voice quality. We give an example of the type of analysis implemented in the tool and on future enhancements to the analysis. We finish with a list of uses we have made with such a monitoring capability.

9 ACKNOWLEDGEMENTS

This work has benefited from numerous discussions with several individuals. In particular, we would like to gratefully acknowledge the contributions of Mark Perkins, Lawrence Brody, Carl Kalbfleisch, Dan Romascanu and Rao Raparla to this work. Also, we would like to thank the referees for their insightful comments, which contributed significantly and positively to the final version of this paper.

10 REFERENCES:

1. Jacobson, V., "*VAT manual pages*", Lawrence Berkeley Laboratory, Feb. 1992; www.nrg.ee.lbl.gov/vat/ .

2. Hardman, V., Kinstein, P., Sasse, A. and I. Kouvelas, "*Robust Audio Tool (RAT)*", www-mice.cs.ucl.ac.uk/multimedia/software/rat/index.htm.

3. ITU-T Recommendation G.107, "*The E-Model, a computational model for use in transmission planning*", December 1998.

4. "*The Cisco-Voice-Dial-Control-MIB*", CiscoSystems, Inc., ftp://ftp.cisco.com/pub/mibs/v2 .

5. "*The Telchemy monitoring agent*", Telchemy, Inc., www.telchemy.com .

6. ITU-T, Recommendation P.861: "*Objective quality measurement of telephone-band (300-3400 Hz) speech codecs,*" August 1996.

7. Dr. John G. Beerends, Mr. Antony Rix, Dr. Andreis P. Hekstra and Dr. Mike Hollier, "*Proposed Draft Recommendation P.86x: Perceptual Evaluation of Speech Quality (PESQ), an Objective Method for End-to-end Speech Quality Assessment of Narrowband Telephone Networks and Speech Codecs,* " ITU-T Delayed Contribution 12-D.140-E, May 2000.

8. ITU-T Recommendation G.113, "*General Characteristics of General Telephone Connections and Telephone Circuits – Transmission Impairments*", February 1996.

9. Perkins, M, Dvorak, C., Leisch, B., and J.A. Zebarth, *"Speech Transmission Performance Planning in Hybrid IP/SCN Networks"*, IEEE Communications Magazine, July 1999.

10. Private communications with Mark Perkins of AT&T.

11. Annex C of Reference [3] contains code to compute the R-factor and obtain the expressions shown in Equation (3) and *Table 2*.

12. Hardman, V.J., Sasse, M.A., Watson, A. and M. Handley, "*Reliable Audio for use over the Internet*", In Proceedings of INET95 (Honolulu, Oahu, Hawaii, Sept. 1995); info.isoc.org/HMP/PAPER/070/html/paper.html .

13. Hardman, V., Sasse, M.A. and I. Kouvelas, "*Successful Multiparty Audio Communication over the Internet*", Communications of the ACM, Vol. 41, No. 5, May 1998.

14. ITU-T Recommendation G.729a, "*Coding of Speech at 8 kbit/s using Conjugate-structure Algebraic-Code-Excited Linear-Prediction (SC-ACELP)*", March 1996.

15. ITU-T Recommendation G.711, *"Pulse Code modulation (PCM) of voice frequencies"*, November 1988.

16. ANSI T1.521-1999, "*Packet Loss Concealment for use with ITU-T Recommendation G.711*", December 1999.

17. Bolot, J.-C., "Characterizing end-to-end packet delay and loss in the Internet", Journal of High Speed Networks, vol. 2, pp 305-323, 1993.

18. M.S. Borella, D. Swider, S. Uludag, G.B. Brewster, "*Analysis of end-to-end internet packet loss: Dependence and asymmetry*", Technical Report AT031798, 3COM Advanced technologies, March 1998.

19. R. Ramjee, J. Kurose, D. Towsley, H. Schulzrinne, "*Adaptive Playout Mechanisms for packetized Audio Applications in Wide Area Networks*", Proc. IEEE Infocom, 1994.

20. S.B. Moon, J. Kurose, D. Towsley, "*Packet Audio Playout Delay Adjustments: Performance and Algorithms*", ACM/Springer Multimedia Systems, 5:17 - 20, January 1998.

21. DTS/TIPHON 5008 VO.2.5, "*Quality of Service (QOS) measurement methodologies*", July 2000 and private communications with A. Clark.

22. Feller, W., "*Introduction to Probability Theory, Vol.1*", John Wiley & Sons, New York, 1968.

23. Kouvelas, I. And V.J. Hardman, "*Overcoming Workstation Scheduling Problems in a Real Time Audio Tool*", Usenix Conference, June 1997.

24. The RMONMIB Home Page at http://www.ietf.org/html.charters/rmonmib-charter.html/

25. The UCD SNMP Project, see http://ucd-snmp.ucdavis.edu/, the University of California, Davis, Calif.

26. The Cisco rttMonMIB, http://www.cisco.com/public/mibs/v2/CISCO-RTTMON-120 5 T.my , Cisco Systems, San Jose, Calif., USA.

27. Redhat Linux 6.1, see http://www.redhat.com/, the Red Hat, Inc., Research Triangle Park, North Carolina, USA.