Application Note:

Experiences in Improving Voice Quality for VoIP using TI's PIQUA

by Jim Donovan

Jim@DonovanDen.com
(781) 721-1007

DocRev: 0.55
April 20, 2009

Abstract:

This Application Note describes specific voice quality enhancements in Texas Instrument (TI)Telogy's Software for Packet Based Telephony designs. The examples are discussed relative to TI's IP Phone SOC, System on a Chip and their Terminal Adapter communications processor SOC to highlight a few of the Texas Instrument's PIQUA concepts and capabilities. PIQUA is a family of technology innovations branded by TI as unique voice and video features working together to improve the quality and performance of communication delivered over a packet based networks.

The intended audience is any existing and/or prospective TI customers and/or TI partners who will evaluate VoIP software and hardware products to understand the PIQUA features of TI Telogy software modules operate within data networks and telephony systems to help quantify and improve the voice quality of a call.

Contents

Figures

1. *What is Texas Instrument's PIQUA all about?*

This application note explains various techniques for improving voice quality of packet based Telephony – specifically Voice over IP network telephone calls (more commonly referred to as VoIP). When two people engage in a conversation on a VoIP phone system they know they can recognize good voice quality; they subjectively assess the phone call as either acceptable or unacceptable. This app note explores a more objective means of assessing voice quality by using TI's PIQUA technology as built into the TI Telogy Software systems; and provides mechanisms to do something about improving the audio quality and overall reliability experienced in VoIP phone calls.

Since the early days of VoIP telephony systems there have been statistics such as a packet counter, packet loss indicators, jitter-buffer diagnostics, and others. These features have been built into voice gateways and voice endpoints and simply enabled counting the voice packets as they are being processed by a signal processor, DSP. Texas Instruments will combine the existing metrics along with proactive actions and the ability to report key satisfaction metrics within an umbrella of capabilities called PIQUA. The PIQUA diagnostics and statistics gathering are intended for real-time optimizations throughout voice networks and at individual endpoints. These PIQUA concepts can also be applied to video streams handled over packet networks.

PIQUA features are being introduced across a variety of VoIP products from TI. The main focus of this writing applies to IP Phones and Residential VoIP Gateways; but most of the concepts also can be applied to Medium Density Gateways as well as High Density Gateways and Media Servers.

1.1. Voice Quality within the Network Topology of VoIP

Primarily, we are discussing capabilities of VoIP endpoints, such as: IP Phones, Small Gateways. PIQUA features also are used in larger gateways and at concentration points along a call path. PIQUA concepts have been deployed in carrier class systems and do not necessarily require coordination with endpoints. The simplified view of the network topology portrays pairs of endpoints working together as PIQUA enabled endpoints communicating over a data network to construct a quality call path.

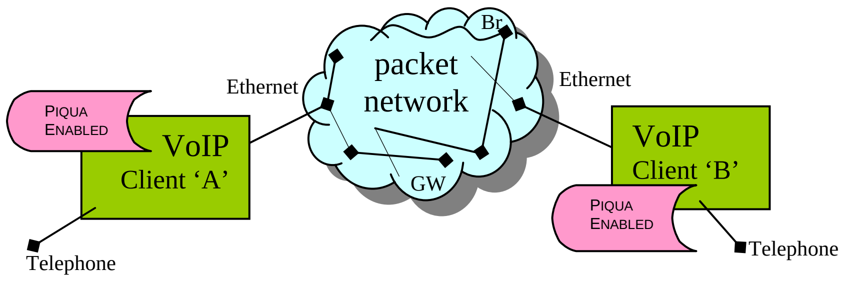


Figure 1 – Elementary VoIP, with PIQUA Enabled Endpoints

Variations of scenarios where a PIQUA enabled endpoint can communicate with a non-enabled endpoint are worthwhile in the bigger picture – but for initial understanding the most robust quality assessment set, it is easiest to just view the world of pairs of enabled endpoints sharing metrics between them. Client 'A' communicates with Client 'B' and they share quality data which indicates the quality of the voice transmission that is being experienced by both participants.

Some discussion of "*what's an endpoint?*" warrants attention. For the purposes of this writing we define an endpoint to be: where the digital signal processor (DSP) is utilized to adapt traditional time division multiplexed (TDM) voice data plus signaling information to create packet based network traffic (UDP/IP, or more generally just Ethernet). The transition from TDM to packet may occur midway along a voice path between two people speaking to each other – but "in path" gateways operate as though they are endpoints in the VoIP aspects of the call. In addition medium density gateways, and even large large carrier class gateways can operate like endpoints if that is where the VoIP portion of a phone conversation begins.

1.2. Voice Path, Making Phone Calls over an IP Network

The packet network provides the infrastructure over which the traffic for VoIP calls occurs and upon which the call control is directed, plus any analysis tools which provide management and monitoring features to administrators and designers of the VoIP based systems. It may be an over-simplification to describe the packet network this way; but to initially consider just a scaled up set of endpoints over an Ethernet packet network we can abstract the network and look more closely at the endpoints.

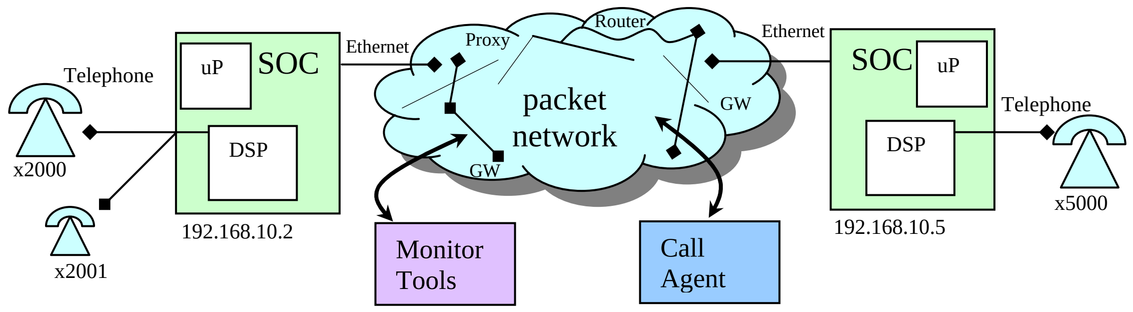


Figure 2 – System Diagram, Network Topology

As with most VoIP system this example topology includes the collection of endpoints, various data routers, and a call manager as well as accounting, management and control elements to make the whole system work. To understand TI's PIQUA features does not require understanding of all of those elements, but some experience with a full solution may help in considering how to best apply PIQUA's capabilities.

A call setup is generally performed with SIP messages being sent to the end-points and proxy servers over an IP Network. To gain visibility to these packets being exchanged between clients and servers the use of a packet sniffer or protocol analyzer is helpful to observe the call control data packets. Essentially the endpoints are coordinated with some means of translating an extension dialed into a destination IP address. To illustrate we use (see Figure 2) simple exchanges with phone numbers x2000, and x2001 – at IP Address 192.168.10.2, and another at 192.168.10.5 with the extension x5000. The example shown uses network values which are completely arbitrary – they are used with any phone numbers and respective IP addresses. We illustrate using specific addresses and telephone numbers to make the point that local packet traffic within a single gateway may result in completely internal packet flow. This would not give evidence externally of any voice path, nor even to indicate the call setup and configuration messages (the entire exchange may occur internally to the single gateway).

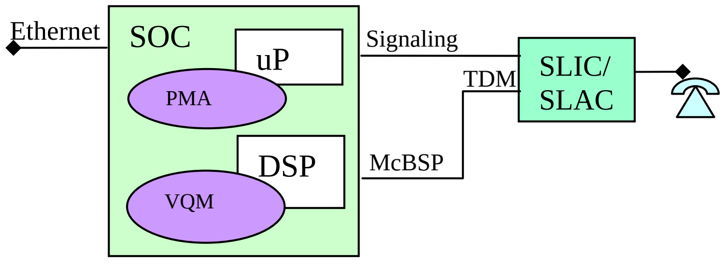


Figure 3 – Hardware: Interfaces and Internals of Telephony "SOC"

The design may be generalized as a common set of features for an IP Phone except that the voice path does not connect via a SLIC/SLAC but is rather connected directly to a hardware codec where the analog voice data is transferred directly into the McBSP for processing on the DSP – the control signals are handled by keypad presses, and on hook/off-hook hardware interfaces instead of the traditional analog telephone handling those events.

The software module on the microprocessor (referenced in
The design may be generalized as a common set of features for an IP Phone except that the voice path does not connect via a SLIC/SLAC but is rather connected directly to a hardware codec where the analog voice data is transferred directly into the McBSP for processing on the DSP – the control signals are handled by keypad presses, and on hook/off-hook hardware interfaces instead of the traditional analog telephone handling those events.
as PMA, PIQUA Multimedia Agent, in TI's Telogy software) is kept in sync with the data being collected by the DSP voice processing modules. Statistics and state machines represent an active call processing to report out the current data associated with calls. The DSP does the real-time voice processing, and periodically informs the micro of the latest values of data; it is prepared to report out at any time on the most recent statistics gathered (for example when the micro indicates that the call has terminated and the end-of-call statistics are transferred).

1.3. Architecture of TI's VoIP Solutions

Illustrating the internal operations of the DSP and it's relationship to the microprocessor and the rest of the network. The signal processor is logically the best source of the voice quality statistics the counters are a natural byproduct of handling the voice data. The packet processing is handled by a microprocessor in most designs – the data needs to be presented to the network. Given these two centers of computing resources, it's important to synchronize the data between the processing units accordingly. The following depiction explains how this is accomplished, the Markov model is a key element, in that collection about gaps, and bursty data loss in the voice stream is tracked and used to turn around results which indicate an 'rValue' and the other listener experience indicators.

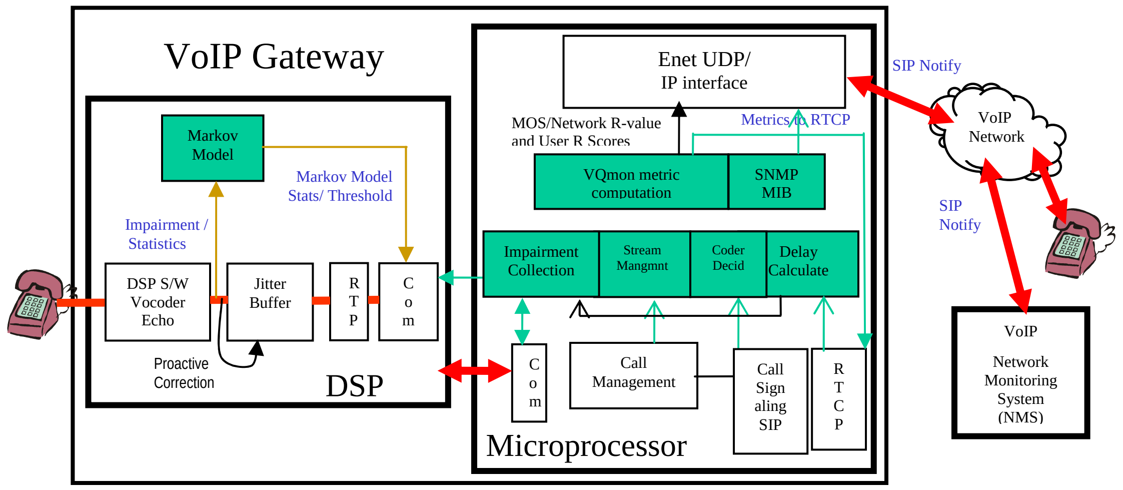


Figure 4 – Software Architecture of Telephony "SOC"

Outside of the VoIP gateway device packets containing statistical data provide the mechanism (here, shown as a "SIP Notify") to communicate quality measures to the outside world – this exchange occurs in the same stream of data as the voice payloads which were already being transmitted and received from each end point.

1.4. PIQUA – Gathering Statistics for Telephone Calls

When a VoIP Call is initiated, a call management protocol is used to coordinate the transactions between the end points and the server. Among the more popular methods today is Session Initiated Protocol, or SIP based servers in comparison to an IMS or H.323 type of call control. With core SIP capabilities a series of messages provide the control for setup and tear-down of the connection between two endpoints.

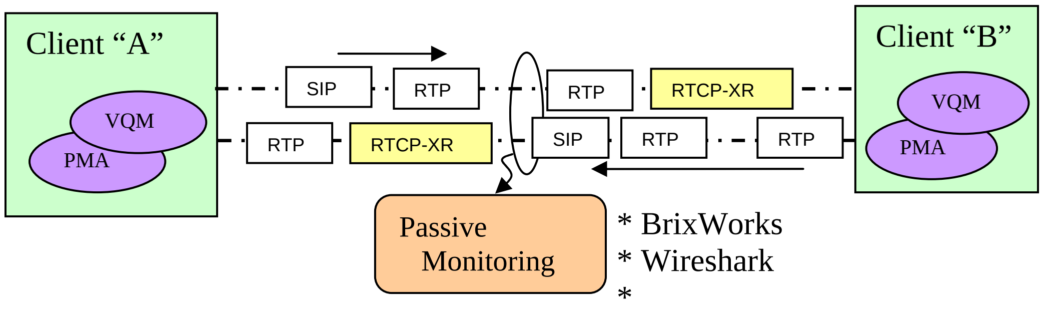


Figure 5 – Packet Exchanges between End Points

Upon completion of a call, there are again messages passed between endpoints and call control which can additionally report out on the final statistics on the call which took place. As depicted in Figure 6 the VoIP call is comprised of a stream of RTP data packets which usually contains 10ms or 20ms frames of voice samples. These series of packets are commonly referenced as the voice path – the primary data exchange which is involved in a VoIP call. In addition to the RTP payload, there are SIP control messages which are involved to get the call started, to complete the call, and other messaging which may occur while a call is in progress. Usually, the call control is responsible for only a small percentage of the data bandwidth for a given VoIP call.

With the advent of RFC3611 there are packets added to the SIP messages exchanged between endpoints – a SIP Notify is one mechanism, and the description of voice quality is the content to be conveyed within the proposed 3611 reccomendations. The block types are described as RTCP-XR packets (for extended reports) which have enabled the exchange of voice quality indices. The exchange is done at a call completion, but can also be sent at regular intervals while a call is in progress.

The SIP NOTIFY is subsequent to a SIP call control message BYE which indicates the call has completed – the notify is responded to with a 200 OK message. As an equivalent mechanism in Cable's product group, the MegaCO/MGCP protocols offer a SIP QoS Report which contains the equivalent relevant statistics gathered during the call is reported out as the end-of-call report.

The different types of analysis tools dictate the approach taken for collecting data: by passive observation of the IP streams, some monitoring tools can determine when calls happen and collect the parameters, statistics, and quality metrics from the data network directly. As illustrated in Figure 5 and Figure 6 there are various means of monitoring a network. When equipment can be located on the same subnet as the endpoints data can be derived just from RTP payloads (by seeing the destination addresses of UDP traffic), and with the use of the SIP messaging such as a call initiation and completion time-stamps.

Depending on the level of support for RTCP-XR or newer format described as RTCP-HR (for "RTCP High Resolution") there can be call quality as measured at either end or both ends of a VoIP conversation. The monitoring of packets to determine call quality is enhanced by standardizations such as RFC 3611 because an end-point is the location best extract the most accurate characterization of the voice processing. The generation of the RFC3611 is a direct result of having processed the actual voice payloads.

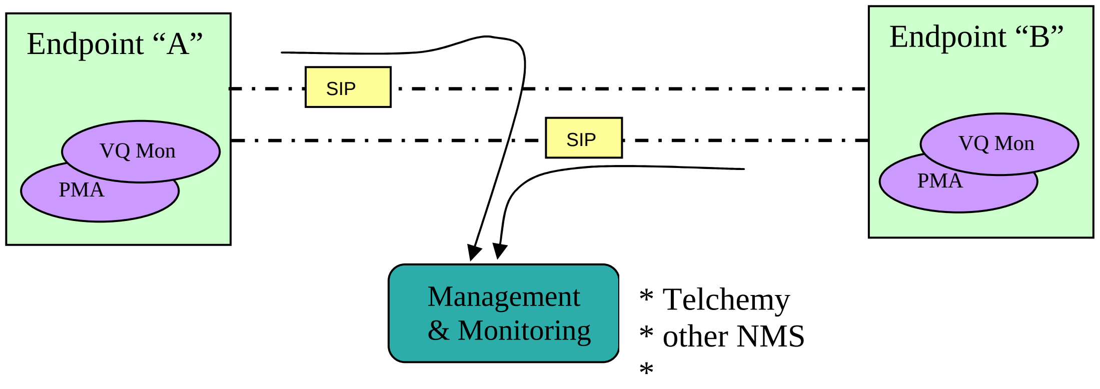


Figure 6 – SIP NOTIFY Message Directed to a Monitor upon Call Completion

The SIP Notify is directed to a server node with the right collecting agent. The advantage of using SIP NOTIFY message with the encapsulated XR data for the call statistics is that a server program (ex: SQ Mediator from Telchemy; or BrixWorks from Brix Networks) doesn't need to reside on the same subnet as the endpoint.   The SIP messages for notifying the call closure has occured can be directed to a specific IP address to be any remote location reached thru other gateways and routers. With the use of RTSP (Real Time Streaming Protocol) the packaging of the data gathered at a client can be organized and presented to a remote application by the client and can be transmitted at a regular interval such that the data is available for analysis by the client application. This is the idea of a session, for the streaming protocol; where a client invokes the session as a negotiated request for certain data.

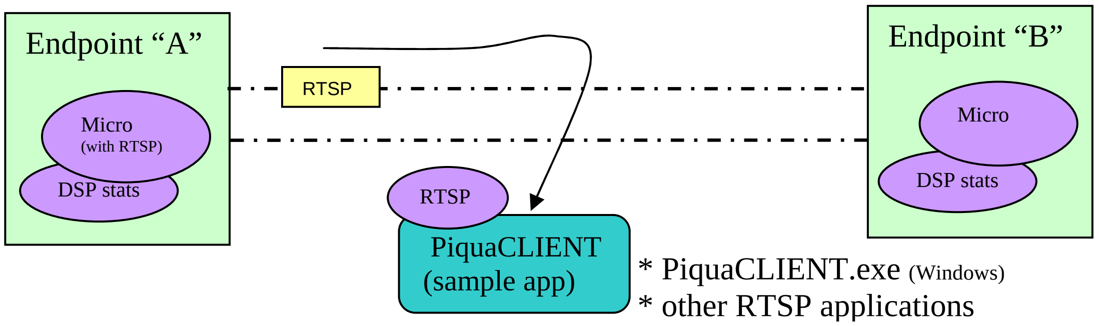


Figure 7 – RTSP used by Client App to request PIQUA data from a Gateway

The end-point in this approach is a TI voice gateway and the sample application is a client – so the end-point is servering data collected by the DSP and then managed by the end-point microprocessor and fed to the PIQUACLIENT program (a Windows/XP based application). This turns the nomenclature around as far as client and server: the end points act to serve the RTSP client for the PIQUA data it will coordinate to fulfill the requests.

The client sessions establish a connection with the Piqua enabled endpoint which makes requests resulting in data streams from the endpoint rather than just a single discrete update of statistics. This potentially results in a high bandwidth demand in contrast to the periodic XR packets being sent along with the voice packets.

With RTSP the implication on bandwidth should be considered carefully as part of the implementation to achieve voice quality improvements.

Whether managed by a session protocol, or just exchanged between a client and server via another mechanism, the ability to forward data from a client to represent what voice quality is being experienced by the listener is now possible. In the TI architecture the endpoint is responsible for the additional interfacing to PIQUACLIENT – it's an individual endpoint acting as the server here and provides the PIQUA data to the Windows/XP application for display and analysis.

1.5. IP Framing: RTP, UDP, RTCP, SRTP and RTSP

There are various packet framing levels of moving voice payloads thru a data network to map the parallel layers of protocol required to routing a series of data packets to their destination. To start, a packet's time stamp and a sequence number will be associated with each RTP packet. The RTP packet contains a 10, 20, or even 30 milliseconds or more of sample voice data. The voice sample may be wrapped with RTCP frame instead of RTP, but in either case it will be futher enveloped within a UDP frame to direct the packet across an Ethernet wide area network WAN. The math for packet rates is very straightforward: based on two end points generating a constant stream of 10ms packets, it means typical VoIP channels will generate 200 packets every second. The network processor must be capable of handling a worst case packet traffic multiplied by the number of voice channels. If there are conferencing features at a client endpoint, the worst case may imply doubling or tripling the packet rate for the additional destinations.

With RTP at the core of a packet’s content, the framing fields imply how the payload will traverse the data network – the routers do not need to look within the RTP packet. When it’s reached it’s destination the payload is processed; the fields around the actual voice sample serve to direct any final processing. For example: a time stamp, or sequence number provide processing hints to relate the current sample to those which had come before it; the hints include knowledge of gaps which occur, and even the relative size of the gaps to indicate if compensation should be made for the smoothing of the voice data in it’s playout processing.

A few additional comments regarding packet structures – first, the “SRTP” capability. To substitute an RTP frame with Secure RTP enables a DSP algorithm to encrypt the voice payload. Using Secure RTP requires coordination between endpoints to handle key exchanges and establishing the agreed upon encryption method. In conjunction with the payload encryption the packet framing can also be protected but SRTP only pertains to the voice samples being secured.

Also, to mention the RTSP, as a point of clarification: as a real-time streaming protocol, TI uses in PIQUA enabled endpoints as a means of streaming VoIP statistics and even traces or logging features to a diagnosis tool (named PIQUACLIENT). With RTSP a gateway can issue reports or send a data stream to be interpreted on a remote host to analyze the voice processing being performed by that gateway.

1.6. PIQUA Leverages RTCP-XR Block 8 Extensions

A typical VoIP conversation involves packets being exchanged between two end points. The packets contain RTP voice data payloads, encapsulated within a UDP frame. With the PIQUA approach to enhancing voice quality additional packets are sprinkled among the UDP payloads to convey metrics which describe the voice quality being experienced at each end of the call. This is not a new concept, the packet exchange to initiate calls occur as data packets exchanged between a call manager and end-points; it is also common for DTMF tones are transferred as data representations instead of actual audio samples. The usage of data payloads which summarize the characteristics of the on-going voice path between two people is just an extension of that concept. The extra payloads should represent a small fraction of the amount of data exchanged to maintain the conversation between two parties.

The XR fields as specified by RFC3611 include the structure shows in **Figure 8**; for each packet stream (i.e. two streams to constitute a bidirectional conversation) there are XR packets, known as block type 8 extensions to the RTCP framing, which contain additional data for all end points that support the generation of these packets.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           OVERHEAD                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Loss Rate   | Discard Rate  | Burst Density |  Gap Density  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|        Burst Duration         |          Gap Duration         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       Round Trip Delay        |        End System Delay       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Signal Level  | Noise Level   |     RERL      |      Gmin     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   R Factor    | ext. R Factor |    MOS-LQ     |     MOS-CQ    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   RX Config   |   reserved    |     Jitter Buffer Nominal     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Jitter Buffer Maximum       | Jitter Buffer Absolute Max    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 8 – Structure of an RTCP-XR Packet, Block Type 8

To highlight information which is included in the RTCP-XR packets, it includes:

- Peak-to-peak IP Packet Delay Variation (IPDV)
- Lost and Discarded Packet count
- Round Trip Delay (from RTCP SR/RR fields)
- Number of Jitter Buffer Adaptation Events
- Signal and Noise Levels

The peak-to-peak IP Packet Delay Variation (IPDV) is defined as:

- V_i: IPDV for packet "*i*" as the amount it has been delayed in transmission, minus the amount the first packet of the connection was delayed, in accordance with ITU-T Y.1540.

  $$V_i = (R_i - S_i) - (R_1 - S_1) = (R_i - R_1) - (S_i - S_1)$$

  Where:
  - R_i is the actual arrival time of packet "*i*" in timestamp units
  - S_i is the RTP timestamp of packet "*i*"

Lost and Discarded Packets, a series of variables which represent packet misplacement:

- The total time in sample periods affected either by packets unavailable due to network loss, or late delivery of packets, since the start of transmission.
- The total time in sample periods degraded by jitter buffer adaptation events e.g., where the jitter buffer either plays out a sample sequence not originating at the transmitter, repeats samples, or chooses not to play out sample sequence which was sent by transmitter.
- Cumulative seconds affected by "unavailable packets" events over the lifetime of the call.

- Cumulative seconds affected by severe “unavailable packets” events over the lifetime of the call. Host software configures/provides the SES threshold in RTP timestamp unit.

Round Trip Delay

- Round trip delay using procedure of RFC3550 for i^{th} measurement. This will be the round trip delay measured from the last received RTCP SR/RR packet.
- Minimum round trip delay experienced over the life of the connection
- Maximum round trip delay experienced over the life of the connection

Residual echo return loss (RERL) – The residual echo return loss value measured by the echo canceller. Note: This is line echo, not acoustic echo.

R factor – The R factor is a voice quality metric describing a segment of a call that is carried over an RTP session. The metric is defined per ITU-T G.107 and ETSI TS 101 329-5 dependent upon a number of factors which influence it’s value.

- *Burst density* – The fraction of RTP packets within burst loss periods since the beginning of reception that were either lost or discarded. A burst is a period during which a high proportion of packets are either lost or discarded.
- *Gap density* – The fraction of RTP packets within inter-burst gaps since the beginning of reception that were either lost or discarded. A gap is a period of low packet losses and/or discards.
- *Burst duration* – The mean duration, expressed in milliseconds, of the burst periods that have occurred since the beginning of reception.
- *Gap duration* – The mean duration, expressed in milliseconds, of the gap periods that have occurred since the beginning of reception.

Notice the distinction of MOS “listening quality” versus “conversational quality” which includes considering the effect of delay. As such, the MOS score is represented by two unique values:

- *MOS-LQ* – The estimated mean opinion score for listen quality is a voice metric on a scale from 1 to 5. This metric is defined as not including the effects of delay.
- *MOS-CQ* – The estimated mean opinion score for conversational quality is defined as including the effects of delay and other effects that would affect conversational quality.

There are also fields included in XR frames which express:

Signal Level – Voice signal level measurement

Noise Level – Noise singal level measurement

Jitter Buffer Adaptation Events

- Count of each occasion when jitter buffer delay is modified. Applicable to both fixed and adaptive jitter buffer.

The calculations are performed upon arriving packets as described below (see Figure 9). Because the DSP handles the voice processing it is the best place for collecting statistics. The consolidation of the statistics into coherent packet summaries also makes sense on the DSP or within close proximity of the DSP (i.e. the micro at the client endpoint).

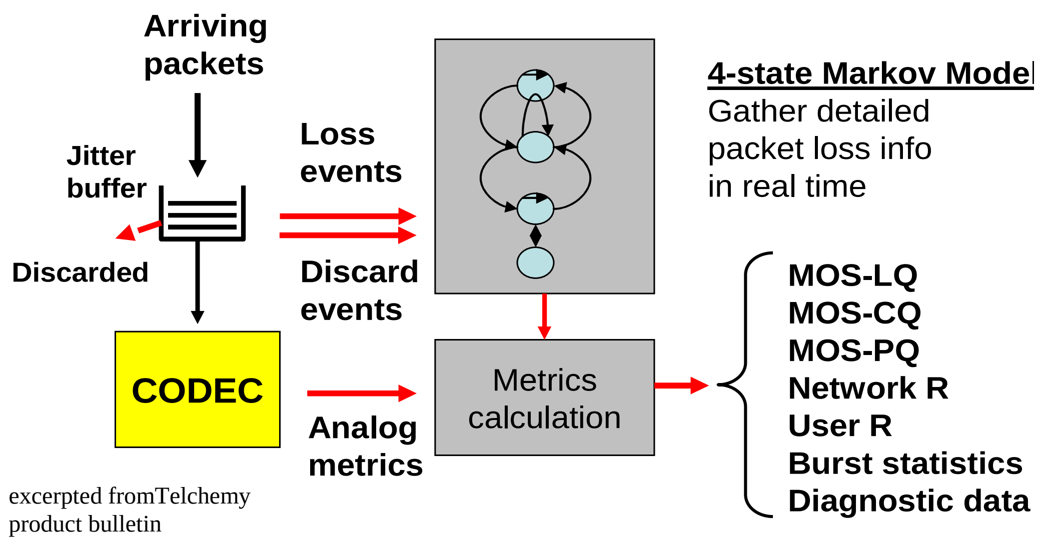


Figure 9 -- Quality Metrics Data Flow

Round trip delay – The most recent calculated round trip time between RTP interfaces, expressed in milliseconds.

End system delay – The most recently estimated end system delay, in milliseconds. End system delay is defined as the total sample delay in the send direction (including encoder and packetization delays) and receive direction (including jitter buffer and decoding delays).

Even more “others” we reference a few other values which may help in more fully describing the voice characteristics of a given call:

- Gmin configuration – Identified the value used for this report block to determine if a gap exists in the packet loss modeling.
- Packet loss concealment configuration – Identifies the packet loss concealment technique at the receiver.
- Jitter buffer configuration – Adaptive or fixed
- Jitter buffer rate – Identified the adjustment rate for a particular jitter buffer implementation at the receiver.

1.7. *Additional Info to Help Manage VoIP Networks*

To describe some other information which is not currently available, but for which it may be possible to derive details from statistics which are collected at VoIP endpoints.

One key concept would be to determine whether Fax and Modem data can be transferred with existing PCM-pass-thru mode of operating with existing networks. Meaning, if existing VoIP connection quality and robustness could be *good enough* to support the level of modem connectivity to enable the minimal 2400 baud connection common to DVR (like TiVO) or Home Security (ADT, Brinks, or such) where dial-up is still an important factor in getting connected to central servers for support, program listings, and program downloads.

Another key, is whether line condition is acceptable to produce the voice quality desired over the packet network; that is to say: garbage-in-garbage-out, if there's poor quality coming into a Residential Gateway then VoIP is not going to be able to produce good quality coming out over any Ethernet network. Of course, we don't want to discount the viability of noise cancellation algorithms, or the ability to boost the audio levels of human speech in the presence of poor SNR – but, that is a topic for some other paper.

The point to be made here, is just simply that with the right telephony interfaces there should be an ability to drive the right line current and the ability to receive voice data with acceptable signal-to-noise ratings is all part of what can deliver the best voice quality. In terms that PIQUA can address, it is feasible to relay this information to the remote end of a call and/or to forward it to an NMS which then can determine if the data is in an acceptable range for a good quality VoIP call.

1.8. More Features within Standards Based Structures

In addition to the default fields for voice quality the information there are enhancements needed for newer features added to VoIP Gateways and end points. For example, for expressing line quality, and other external influences to voice processing such as echo which is introduced to the network by the hybrid electronics (where the conversion occurs from four wires to two wires to handle both incoming and outgoing audio). There is data to help control the voice path – such as tone processing, or special tone sequences which may be flagged to transition into and out of FAX or Modem modes of operation. And, of course, the signaling information which provide the mechanisms for auxiliary features such as call forwarding, and call waiting, forwarding, or special voice mail and message indicators which have become part of the current telephone systems today.

1.8.1. Echo Quality Index

The TI-Telogy SW includes a metric for estimating the quality of the echo signal and echo conditions provided to a line echo canceller. This metric, called the Echo Quality Index (EQI) is intended to identify network conditions that may lead to potential line echo canceller performance degradations. The EQU metric is a number between zero and one; with low values meaning the canceller is not operating effectively, and a higher number to imply that the line echo (introduced by a hybrid) is being effectively removed from the listener's experience.

This EQI is being incorporated into newer TI Extensions to the RTCP-XR, RTCP-HR and SIP Notify end-of-call reports.

1.8.2. Line Condition Summaries

The end of the SDB the boot-up display includes the line data on the RS-232 terminal window display. There are also commands, in the NMM command line interface which can display further details (from the MXP> prompt, using “tiu test” commands.

Need to explain not just the codec capability, but what is different in double the sampling input rate (i.e. 3 kHz versus 8 – the rate of PCM data clocking, and the frequency response range of the handset electronics).

1.9. Video Quality Metrics Build Upon VoIP Techniques

For video systems transferring data on packet networks the issues are certainly different than the requirements of a VoIP transmission. However, the mechanisms for transporting the pairs of end-points can benefit from the experiences learned in VoIP. The question becomes which measure of video quality accurately represent how “good” the video output appears to a viewer. Even when a reliable transmission of data occurs and the processing of the data stream has occurred without fail, the displayed results may suffer from local hardware issues which are independent of the network considerations. The measures of quality we’re addressing is in the end-point processing and packet network based influences of the processing.

2. Software and System Requirements

TI’s Residential Voice Gateways are currently developed for an embedded Linux environment; although they can be ported to other operating systems it is important to understand that the direct support for the platform is Linux. Other TI VoIP devices are run on VxWorks and other kernels in various vertical markets such as: Medium Density Gateways, IP Phones or High Density Gateways -- those do provide formal support for non-Linux operating systems.

The various environments supported by the TI’s VoIP solutions include:

Device	**TI VoIP Release**	**Platform**	**Operating System**	**Version**
TNETV1060	EGW v10.x	Titan	VxWorks (Wind River)	R5.5
TNETV1050	IP Phone v10.3	Titan	Linux (Monta Vista)	R2.4 v17
TNETV1050	IP Phone v11.1	Titan	VxWorks (Wind River)	R5.5
TNETV1060	ResGW v11.3	Titan	Linux (Monta Vista)	R2.4 v17
TNETD5320	Cable	?501C?	VxWorks (Wind River)	R6.2
TNETD????	aDSL	UR8	Linux (Monta Vista)	R2.6 krnl
TNETV1061	ResGW v12.0	Petra	Linux (Monta Vista)	R2.4 v17
TNETV1051	IP Phone v11.2	Aries	VxWorks (Wind River)	R6.2
TNETV1061	ResGW v13.x	Petra	Linux (Monta Vista)	R2.6
TNETV2520	EGW v12.1.x	Apex	VxWorks (Wind River)	R5.5
TNETV3010	High Dens v11.	HD3	VxWorks (Wind River)	R5.5

Table 1 - Operating System and kernel support for TI’s VoIP Solutions

The operating system pertains to the microprocessor in the system; a DSP in each case is used for most of the real-time voice processing, and data is handed off to the local micro for being sent to the packet network. Only in the more high-end gateways do the DSPs deliver data directly onto the Ethernet packet network.

Not all of the platforms listed above are currently supporting all of the PIQUA features. But, as one example, the TI TNETV1061 SDB (software development board), can be used to get software development efforts underway, or to confirm product functionality and to verify voice quality or for interoperability with other hardware systems.

2.1. System Software for TI's Telogy SW Enabling PIQUA

Configuring TI's Telogy software development boards is usually done in conjunction with other systems – so, it's very likely that interaction with another board, a computer, or other gateways and IP devices will be necessary. Without trying to guess all of the options at every customer site our intent is to summarizing a few of the most likely situations. For basic configuration options, each unit is likely to need PC based software for a terminal emulator, a TFTP server, and maybe even a Ethernet protocol analyzer.

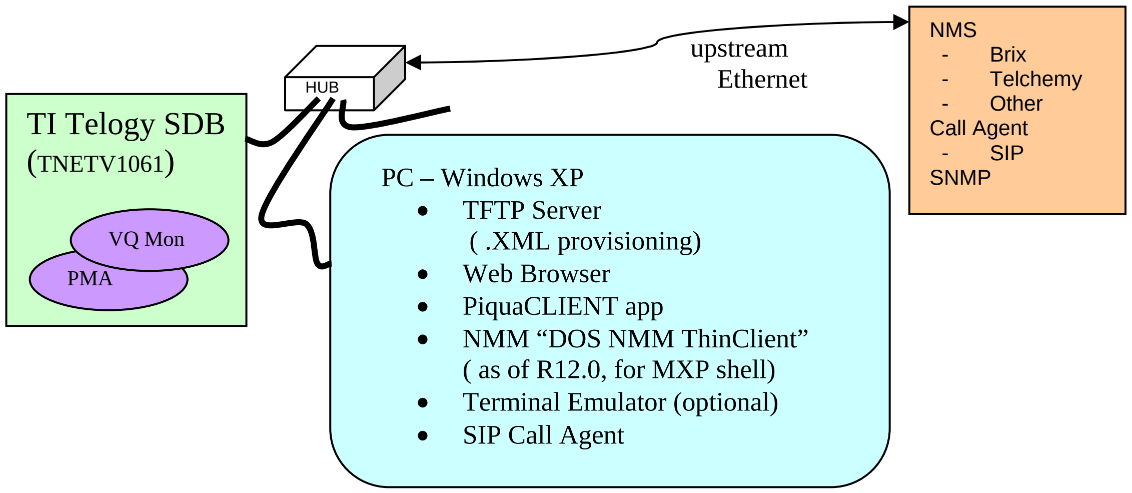


Figure 10 – System Software Elements: Windows/XP

The following list, is intended to provide a summary… it's not an exhaustive list and is not intended to indicate the only options which can be used.

This is meant as a quick reference guide:

Web Browser

A web browser is used for most of the Residential Gateway configuration; we have used Microsoft's IE6 and have no comments about use of other web browsers. If they work… great, if they don't TI does not expect to expend any development time or effort in the support group in understanding differences from the Internet Explorer, rev 6. (no stance has been taken yet about the use of Version 7 of MS IE Explorer).

Call Agent / Dial Plans

Basic dial plans are setup between a gateway and other client devices, such as another gateway or an IP Phone. This is dependent upon directing the SIP protocol to have calls go to the second board – but, a Proxy server is more often utilized for each customer's environment. To illustrate the use of a software based call manager, we've included sections on Proxy Server and used a free download evaluation SIP Server called OnDO from Brekeke Software. It can be downloaded from http://www.brekeke.com. For questions relating to that product, please contact Brekeke. Setup of the call manager is covered in section 3.5.11 and the placing of a basic call is described in section 3.1 (default configuration, is with a single box – directives to have the stand-alone board be able to just use it's own IP Address as the destination of the call manager); otherwise phone extensions are forwarded to a call manager to translate to an IP destination.

The local box can store dial-plans in the TI Telogy MXP environment, for temporarily adding, testing and debug of connectivity with other units. To set up dial-plans refer to the NMM User Guide; but, for now it will be presumed that a SIP proxy is being used.

Ethernet Protocol Analyzer

The R10.12 Ethereal is the one which understands RTCP-XR packets.
The URL is: www.Ethereal.com (that only has the older: R 0.99.0)

Another URL is: http://www.openxtra.co.uk/freestuff/ethereal-0.10.12-changes.php
The nice things about this newer version of Ethereal is that they display the fields of the XR block 8 extensions to comprehend the RFactor, MOS score, and other statistics. The latest version of Ethereal is called Wireshark; you should verify that Wireshark is the equivalent revision for the features we've referenced in this paper.

TFTP

The TFTP server we have used for configuring the boards is from SolarWinds. They have a few very useful utilities for Ethernet networks; you can download their software from: www.SolarWinds.net. Other TFTP servers would also work; we make no claims about the download from SolarWind's site – it's provided for your convenience only.

Download of provisioning configuration files

A TFTP server default login directory contains multiple XML scripts for a given subnet. The directory is specified in the ResGW Web GUI config screen, (under provisioning, in the Advanced tab) – the files in the directory need to have a specific name for each box which uses the TFTP server; the name is a combination of the prefix "ti" and the basename which based on the twelve digit MAC address of the WAN port for the box.

NMMThinCli.EXE

Any version of Residential GW after R12.0 will require a utility (which comes with customer's shipment) as needed for entering MXP command line interface. All of the "set" and "show" commands which provide temporary configuration changes in lieu of XML file being transferred by the TFTP server to a client.

The command line for initiating the MXP command shell must be run from a DOS prompt, and uses a "-s" option to specify the IP Address of the Petra SDB. There's no doubt that cut-n-paste, and script download is not as easy from a DOS shell compared with most terminal emulator programs – but the NMM commands should be minimal changes in configuration, the real changes should be made via the XML with TFTP.

PIQUACLIENT.EXE

A client, windows application, which establishes a RTSP and displays all of the RTCP-XR packet fields (each sub-window determines which statistics are requested, and then displays the values from the GW).

SoftPhone

A windows application for making a PC into an "IP Phone", this application can also work on a Linux PC, or even Mac or PDA/Cell phone; but, the purpose of being included here is simply to enable other "end points" for being involved with IP based voice traffic. If a softphone is not cabable of generating XR packets then there's no real relevance to the PIQUA capabilities. But, in any event, some examples of a softphone are available at: www.xten.com

NIST-Net

A network impairment tool which runs on a Linux PC… this program can be downloaded and built on various Linux machines… it is available from:

www.snad.ncsl.nist.gov/itg/nistnet

2.2. Alternative, System Software: Linux based example

Refer to Figure 11 of the same topology of TI VoIP boards with a Linux O/S as the Host.

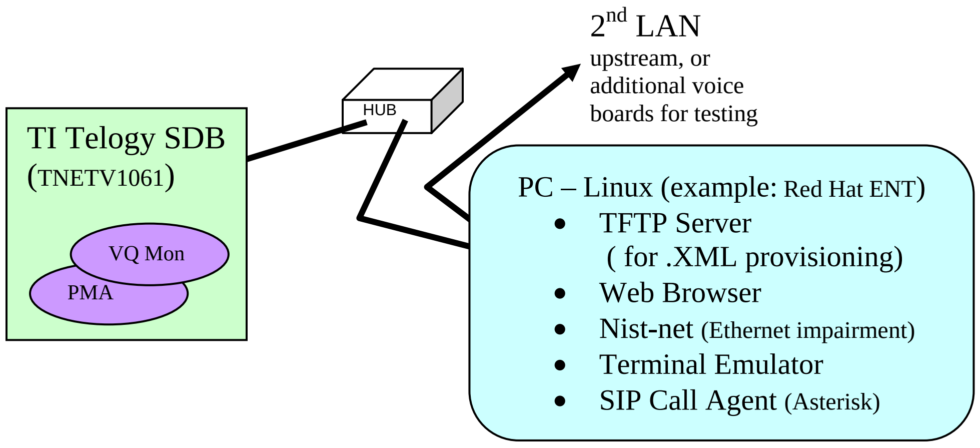


Figure 11 – Alternative Setup, System Software Elements with Linux

The difference in using a Linux host in conjunction with setup and configuration of the TI Telogy software boards is that some of the utilities are currently only available on

Windows – the "nmmthincli.exe" program on ResGW is not available on Linux so board configuration would need to be made by using TFTP of XML files for config changes [note: this only pertains to TI Residential Gateways].

Similarly the "PIQUACLIENT.exe" program is a Windows application. All of the other software needed can be used on either a Linux or Windows host except the "Nist-net" network impairment simulator and performance monitoring tools are only available on Linux machines.

2.3. Sending PIQUA Data to Network Management Server

The SQMediator software does not parse the IP data of SIP INVITES or even of the XR packets which are consistently sent during the calls. It is using the "SIP NOTIFY" at the end of the call which contains the XR records after the "SIP BYE" message is sent.

As described above some PIQUA capabilities can be performed by monitoring a local subnet; but, for some features of network managment a client must actively forward the information to a server – this is more flexible in that the server doesn't need to reside on the same subnet; but, a client must individually be configured to send the information out to the server.

3. System Logistics – A Single Board PIQUA Configuration

Illustrating PIQUA features can be done to a limited extent a call between two lines on a single TNETV1061 "Petra" development board. But no voice data traffic will be witnessed outside of the box (this example uses the residential gateway configured for local calls only) but the statistics of the call are available both during the call, and after a call is completed.

To start with the presumption that the call control uses SIP (for setup/teardown) on a basic reference design and that it is initialized on a local network with other IP units. The TI Telogy software is used with a stand-alone VoIP gateway (in other words the SIP Proxy is just referencing the single local node and all calls are kept within this gateway's IP Address). Even more fundamental is that the IP Address of this single board can be manually assigned as a fixed address (such that SIP messaging is easier to configure) or it may be connected to a DHCP server and also with a software call manager for initiating tests.

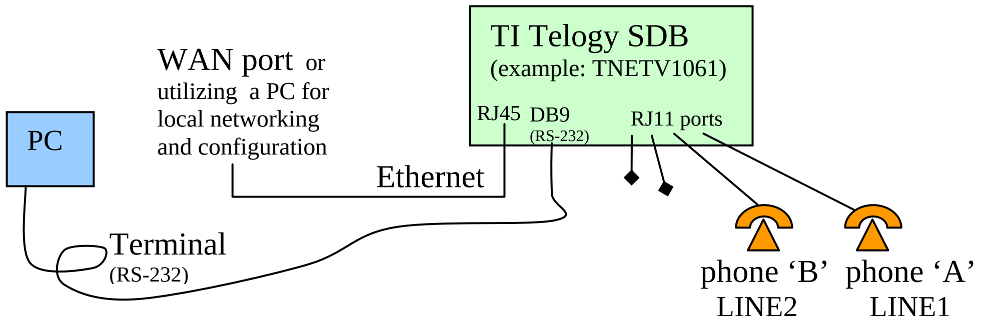


Figure 12 – Hardware: Basic Setup of a TNETV1061 Development Board

The box can be connected with another VoIP device over the WAN; the first preparation for voice path would be to confirm connectivity with a “ping” via Ethernet to be sure the units are communicating on the same subnet or that they can traverse over intermediate networks to exchange packets between them. Consider that even after a “ping” which confirms TCP traffic is working, the VoIP traffic is dependent on UDP based exchanges which may not navigate thru networks even if an IP “ping” is working.

After a “ping” has been confirmed to communicate to the board and the pair of VoIP boxes are coordinated with the same SIP proxy setup, you should be able to place basic calls between boards. With an Ethernet protocol analyzer the traffic can be monitored and by tracing SIP INVITE packets and the responses to those “invite” messages. An initiated call from one board will be acknowledged, and then connected to the second board. The same coordination occurs at the end of the call and a “tear down” occurs.

3.1. Verifying a Port-to-Port FXS call

After the first call has confirmed that hardware is working and is configured properly the voice path between two ports (even just on the same gateway) will provide DSP stats for helping to characterize the voice processing that took place. The statistics are, in most cases, preserved at the end of a call up until the next call restarts the counters in the DSP. It is important to understand that to collect information about the packet traffic the call must traverse the Ethernet; calls between two ports on the same gateway generally do not result in any data packets being observed outside of the box. But, to verify that a box is configured properly just calling from LINE1 to LINE2 within the same box can still help confirm basic connectivity

For this description, the LINE1 (we’ll refer to extension two thousand, or x2000) as been mapped to “tcid 0”, you can use an analog telephone plugged into the RJ-11 jack of “LINE 1” on the back of the box to call LINE2. Pickup the LINE1 phone – there should be a dial-tone heard in the telephone handset; dial “2001” to connect to the “LINE 2” in the same box. See Figure 2 – System Diagram, Network Topology, for a depiction of the basic network setup.

An important aspect of connecting two IP clients together with an Ethernet switch between them is that packets must be visible on the other ports of a switch if a protocol analyzer is expected to be used. Your Ethernet connection should be a “plain hub” [i.e. an old hub] or you need to be able configure the switch to allow promiscuous mode of sending all packets to all ports.

Most inexpensive four port hubs do NOT have this capability – you will not be able to capture the Ethernet traffic between two end points because the PC with Ethereal doesn’t see the packets; you ***must*** have a HUB to see the packet traffic.

3.2. Reporting VoIP Statistics per Channel

Associated with each call which is active or has been completed there are some basic statistics gathered by the DSP which the microprocessor has access to by simply querying the DSP.

Some of the functions to request the statistics are as follows:

- rxtxstat
- tlevel
- gains
- tstat
- vpstat
- call_record
- faxstat
- vadstat
- ec_debug_stat
- ecpath_coeff

The statistics have various means of output to the user or developer. In the following sections of this document we describe the various means of generating data output.

3.2.1. Basic Statistics from the Command Line Interface

The traditional interface for TI's development boards has been thru a command line interface with a command shell named "MXP" which evolved from the early days of Telogy Software platforms with a simple RS-232 "Craft Port". It utilizes any terminal emulator for entering commands, downloading scripts, and provided access to the "boot monitor" for making low-level board configuration to the hardware.

Here we provide an example output of the DSP statistics displayed in a terminal emulator window in any version of the TI Telogy software. The syntax is common across the platforms supported for other Telogy verticals from IP Phone thru High Density Gateways.

```
MXP> show rxtxstat 0
MXP> NMM: Display data, timestamp=3679290
NMM: 0, RXTX Stats
  Rx Voice Packets              = 4424
  Tx Voice Packets              = 4423
  Tx Silence Suppressed Frames = 0
  Rx Min Pkt Interarrival time = 0 (ms)
  Rx Max Pkt Interarrival time = 35 (ms)
  Rx RTP Avg Jitter             = 18 (pcm samples)
  Tx Grant Sync Dropped Frames = 0
  Tx Octets                     = 88460
  Rx Octets                     = 88480
  Coding Profile Changes        = 0
  Tx 2833/AAL2 Event Packets    = 0
  Rx 2833/AAL2 Event Packets    = 0
  SID Rx Packets                = 0
  SID Tx Packets                = 0
  TX Last Timestamp             = 687652588 (0x28fcbeec)
  TX Extended Sequence number  = 0 (0x0)
  TX Last Sequence Number       = 41266 (0xa132)
  TX Last Packet Type           = 0
  RX Last Timestamp             = 3320377828 (0xc5e8f1e4)
  RX Last SSRC                  = 2134506670 (0x7f39fcae)
  RX Extended Sequence number  = 0 (0x0)
  RX Last Sequence Number       = 11567 (0x2d2f)
  RX Last Packet Type           = 0
  Packets lost by Net           = 0
  Peer 2 Peer Packets to host  = 0
  Peer 2 Peer Filtered packets = 0
  Peer 2 Peer Squelched packets= 0
  Packets received from net     = 4424
  Packets transmitted to net    = 4423
  Rx Last Voice Prof Index      = 0
```

This output was captured directly from a terminal emulator connected thru the RS-232 cable into a UART from the TNETV1061. Starting in R12.0 and later versions of the

Residential Gateway software releases the DOS utility "nmmthincli.EXE" needs to be run to connect to the ResGW. The other alternative of using a windows based utility "PIQUACLIENT.exe" (see next section) which will connect to the gateway and display the session data (including PCM traces, rx/tx statistics, MOS data).

The basic information on packet counts, packet loss, time stamps, jitter and such have been available from the DSP since very early versions of the VoIP software capabilities; the more recent enhancements that are specific to PIQUA provide more computation with the statistics, such as showing "R-Factor" or "MOS rating" which indicate how the voice quality of a call is being experienced by the users at the end-points.

3.2.2. PIQUA Example Output: show vqmonstat 0

For specific values from the DSP which are used in deriving the more advanced indicators of voice quality the command line interface may be used, or more graphical interface with easier navigation of the multiple attributes on multiple channels to simplify the presentation.

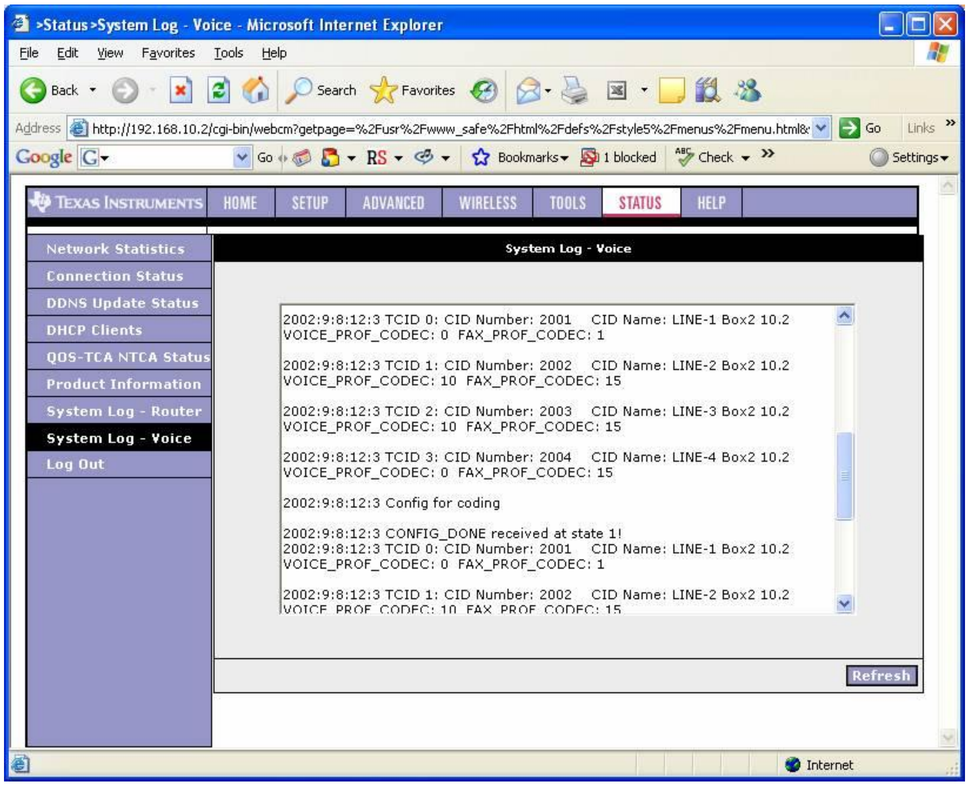
The statistics of an active call displayed on the terminal emulator:

```
MXP> show vqmonstat 0
MXP> NMM: Display data, timestamp=3800256
NMM: 0, Voice quality monitor Stats
  VQMon Version                     = 2
  Coding Type                       = 68
  Segment Duration in Half Ms       = 20
  Segments Per Packet               = 2
  Count of Bursts                   = 0
  Total Burst Received Segments     = 0
  Total Burst Lost Segments         = 0
  Burst Excess From Previous Event  = 0
  Total Gap Recieved Segments       = 69339
  Total Gap Lost Segments           = 6
  Gap Excess From Previous Event    = 0
  VP Concealment Type               = 2(G711A1)
  Num Segments Since Last Burst     = 65535
  VAD type                          = 1(Internal)
  Speech level                      = -44 (dBm)
  Noise level                       = -84 (dBm)
  nRERL_dB                          = 18 dB
  Total Discard Segments            = 0
```

These statistics show the VQ Mon metrics which are used in derivative calculations. Of the values above some of the key information to be used is the gaps and burst packet loss, which are most closely related to the actual listener perception of voice quality. The larger gaps, and increases in packet loss will lower the MOS score.

3.2.3. Monitoring Configuration, ResGW Web GUI Interface

This section will be enhanced to explain the details displayed under the "Status" form of the Web User Interface via a Browser. Currently it's not clear how to direct which information is displayed in this form – note: some of the same information is echoed out thru the terminal port (RS-232) and some of it's unique to the web interface.

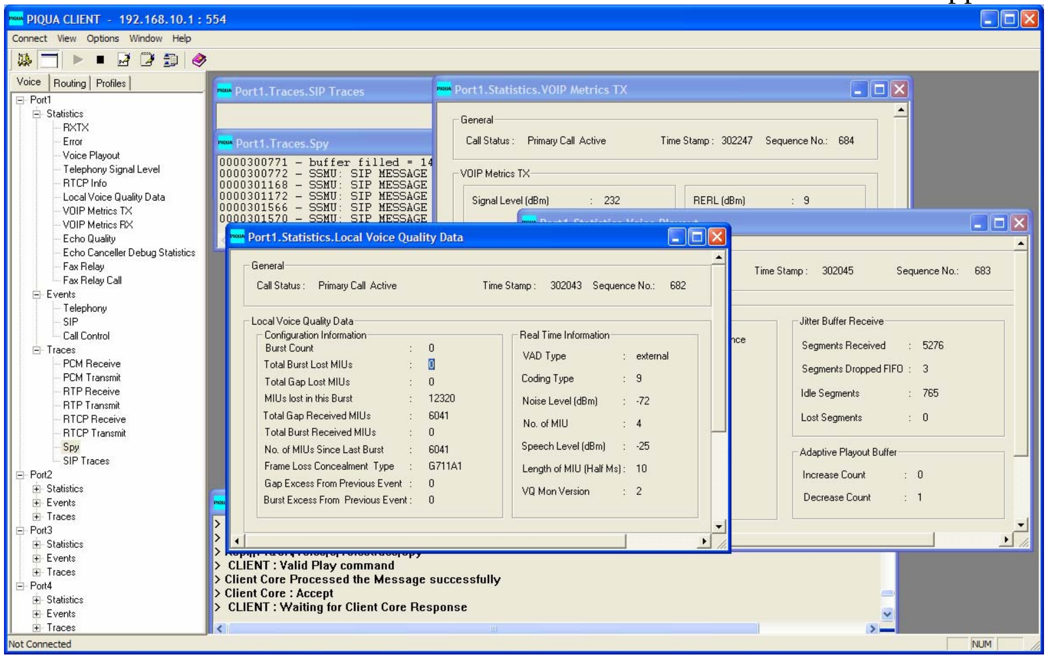


System Log - Voice

```
2002:9:8:12:3 TCID 0: CID Number: 2001    CID Name: LINE-1 Box2 10.2
VOICE_PROF_CODEC: 0  FAX_PROF_CODEC: 1

2002:9:8:12:3 TCID 1: CID Number: 2002    CID Name: LINE-2 Box2 10.2
VOICE_PROF_CODEC: 10  FAX_PROF_CODEC: 15

2002:9:8:12:3 TCID 2: CID Number: 2003    CID Name: LINE-3 Box2 10.2
VOICE_PROF_CODEC: 10  FAX_PROF_CODEC: 15

2002:9:8:12:3 TCID 3: CID Number: 2004    CID Name: LINE-4 Box2 10.2
VOICE_PROF_CODEC: 0  FAX_PROF_CODEC: 15

2002:9:8:12:3 Config for coding

2002:9:8:12:3 CONFIG_DONE received at state 1!
2002:9:8:12:3 TCID 0: CID Number: 2001    CID Name: LINE-1 Box2 10.2
VOICE_PROF_CODEC: 0  FAX_PROF_CODEC: 1

2002:9:8:12:3 TCID 1: CID Number: 2002    CID Name: LINE-2 Box2 10.2
VOICE_PROF_CODEC: 10  FAX_PROF_CODEC: 15
```

3.2.4. Streaming Statistics, in PIQUACLIENT.EXE WinApp

The statistic for reporting out the result of the DSP's gathered rValue, and MOS. Shown here from the "PIQUACLIENT.EXE" application in Windows. The same data is shown in the commands via a Terminal Emulator or from the "NMMThinClient.EXE" app.



Port1.Traces.SIP Traces

Port1.Traces.Spy

```
0000300771 – buffer filled = 14
0000300772 – SSMU: SIP MESSAGE
0000301168 – SSMU: SIP MESSAGE
0000301172 – SSMU: SIP MESSAGE
0000301566 – SSMU: SIP MESSAGE
0000301570 – SSMU: SIP MESSAGE
```

Port1.Statistics.VOIP Metrics TX

General
Call Status : Primary Call Active   Time Stamp : 302247   Sequence No.: 684

VOIP Metrics TX
Signal Level (dBm) : 232   RERL (dBm) : 9

Port1.Statistics.Local Voice Quality Data

General
Call Status : Primary Call Active   Time Stamp : 302043   Sequence No.: 682

Local Voice Quality Data

Configuration Information		Real Time Information	
Burst Count	0	VAD Type	external
Total Burst Lost MIUs	0	Coding Type	9
Total Gap Lost MIUs	0	Noise Level (dBm)	-72
MIUs lost in this Burst	12320	No. of MIU	4
Total Gap Received MIUs	6041	Speech Level (dBm)	-25
Total Burst Received MIUs	0	Length of MIU (Half Ms)	10
No. of MIUs Since Last Burst	6041	VQ Mon Version	2
Frame Loss Concealment Type	G711A1		
Gap Excess From Previous Event	0		
Burst Excess From Previous Event	0		

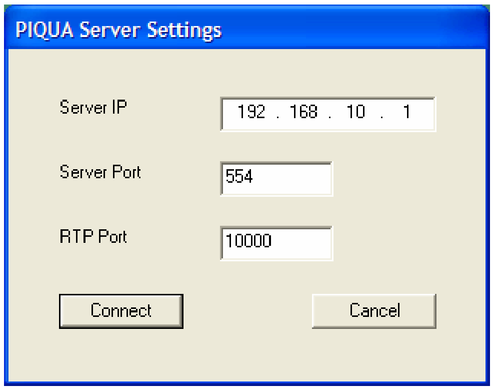
Time Stamp : 302045   Sequence No.: 683

Jitter Buffer Receive	
Segments Received	5276
Segments Dropped FIFO	3
Idle Segments	765
Lost Segments	0

Adaptive Playout Buffer	
Increase Count	0
Decrease Count	1

```
> CLIENT : Valid Play command
> Client Core Processed the Message successfully
> Client Core : Accept
> CLIENT : Waiting for Client Core Response
```

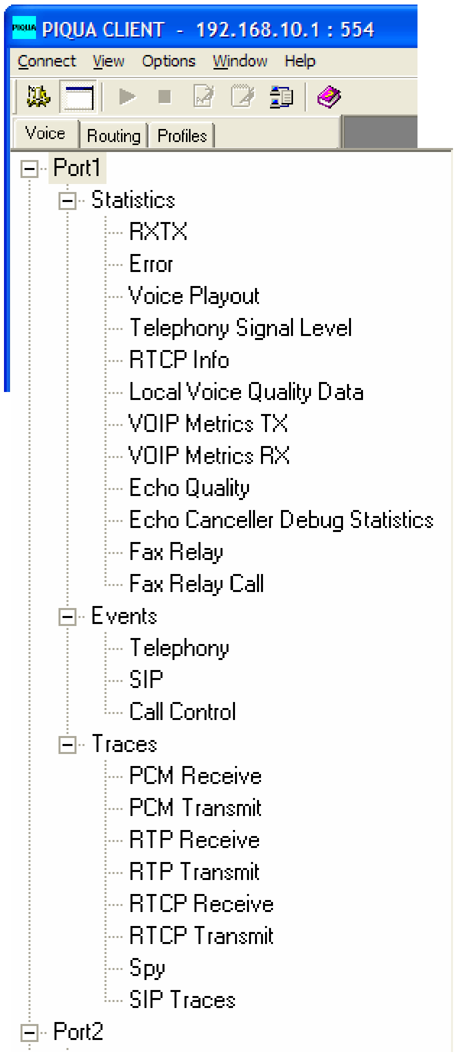
The initial screen for PIQUACLIENT appears with fields for the IP Address, the server port number and an RTP port. Enter the IP Address, and leave the ports at their default values.


PIQUA Server Settings

Server IP	192 . 168 . 10 . 1
Server Port	554
RTP Port	10000

The Server IP is the address of the Petra SDB Residential Gateway board's WAN port. From this dialog box, a full window with explorer style control appears on the left side of the page where the ports 1-4 can be selected to decide which statistics, which events, or traces are reported from the gateway to display in the client.

The mechanism for update to the PIQUACLIENT application is RTSP, and as such, the information being displayed is updated on the fly. All of the windows which are opened for each collection of statistics and voice related data will be changed as live calls take place.


PIQUA CLIENT - 192.168.10.1 : 554

- Port1
  - Statistics
    - RXTX
    - Error
    - Voice Playout
    - Telephony Signal Level
    - RTCP Info
    - Local Voice Quality Data
    - VOIP Metrics TX
    - VOIP Metrics RX
    - Echo Quality
    - Echo Canceller Debug Statistics
    - Fax Relay
    - Fax Relay Call
  - Events
    - Telephony
    - SIP
    - Call Control
  - Traces
    - PCM Receive
    - PCM Transmit
    - RTP Receive
    - RTP Transmit
    - RTCP Receive
    - RTCP Transmit
    - Spy
    - SIP Traces
- Port2

As shown below (Figure 13) the charting of output values over time is automated to give a sense of the trend of the statistics. The interval for updates can be controlled, as can the data ranges for the presented data.

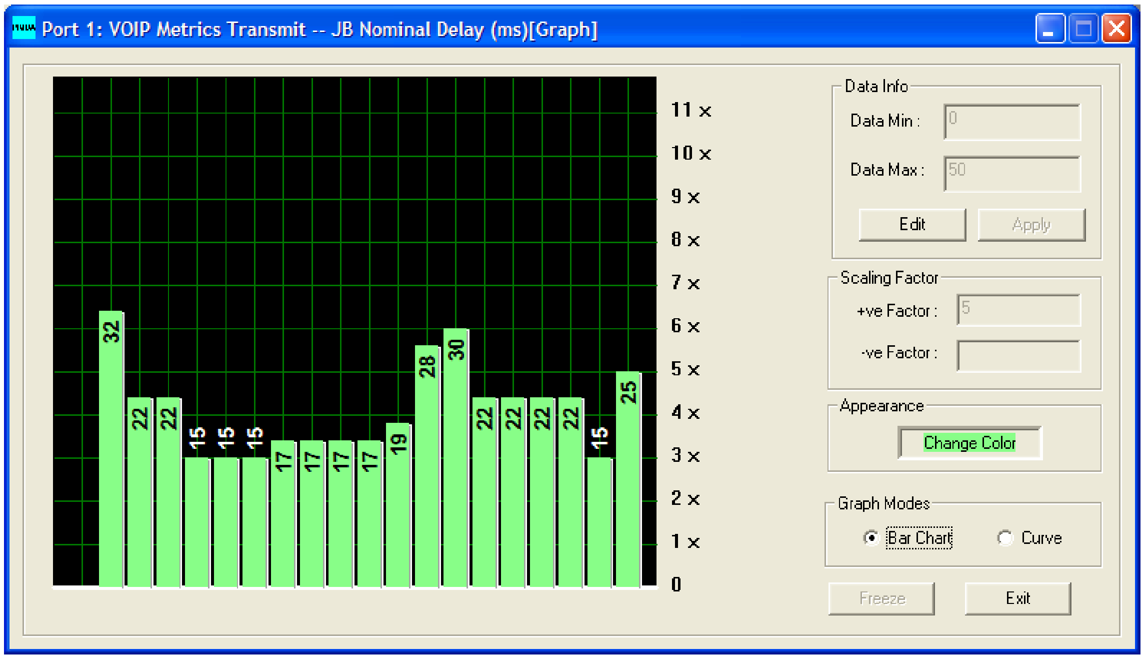


Figure 13 -- PIQUA CLIENT graphical output

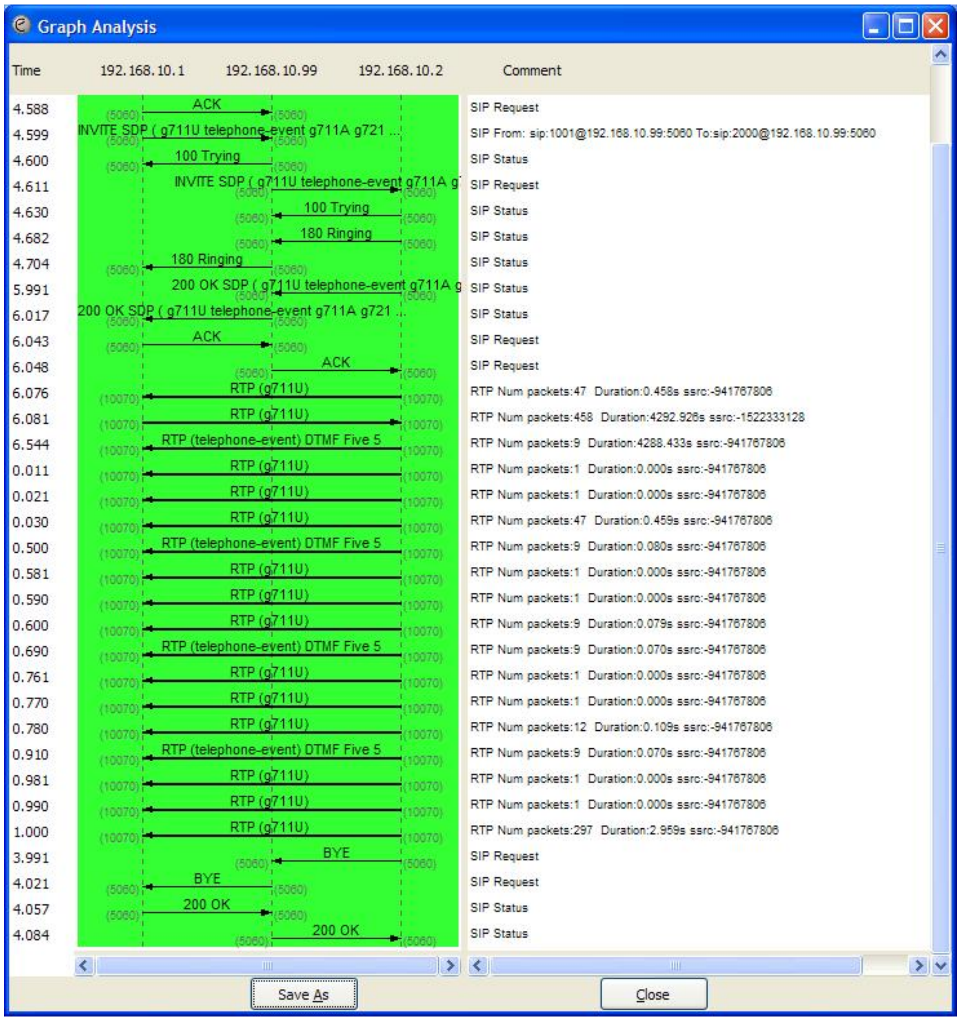
3.2.5. More PIQUA Statistics: show vqmon_metrics 0 local

Whether presented within the WinApp as part of the multiple screens of data, or from the terminal emulator software the statistic for reporting out the result of the DSP's gathered rValue, and MOS from the MXP command interface. The function allows getting the data from either the direct client endpoint or from the peer involved in the VoIP conversation; the VQ Mon metrics can be queried for "local" or for "remote".

```
MXP> show  vqmon_metrics 0 local
MXP> OK
NMM: Display data, timestamp=518839
NMM: 0, Voice quality monitor local metrics
nLossRate          = 0 (0%)
nDiscardRate       = 0 (0%)
nAvgBurstDensity   = 217 (85%)
nAvgGapDensity     = 0 (0%)
nAvgBurstDuration  = 50 ms
nAvgGapDuration    = 65535 ms
nRTDelayMs         = 5 ms
nEndSystemDelayMs  = 102 ms
nSignalLevel_dBm   = -44
nNoiseLevel_dBm    = -62
nRERL_dB           = 6 dB
nMinGapSize        = 16
nRFactor           = 91
nExtRFactor        = 127
nMOS_LQ            = 41
nMOS_CQ            = 41
fRXConfig          = 255
nJBNomDelayMs      = 22 ms
nJBMaxDelayMs      = 55 ms
nJBAbsMaxDelayMs   = 120 ms
```

3.2.6. *Decoding the SIP Call Control Messages*

For the next section before we look at a specific packet content within the Ethernet IP structure, a brief explanation of the packet formats is appropriate. When SIP messages are used to manage the control needed between end points, or between an end point and a call agent it is common practice to use SIP and extensions to SIP for the data packets. These packets can best be observed with the use of a Protocol Analyzer or “packet sniffer” on the same subnet (with a “plain hub” to assure all packets are observed by the analysis tools)… for example, watching the call sequence for a quick initial call see Figure 14.

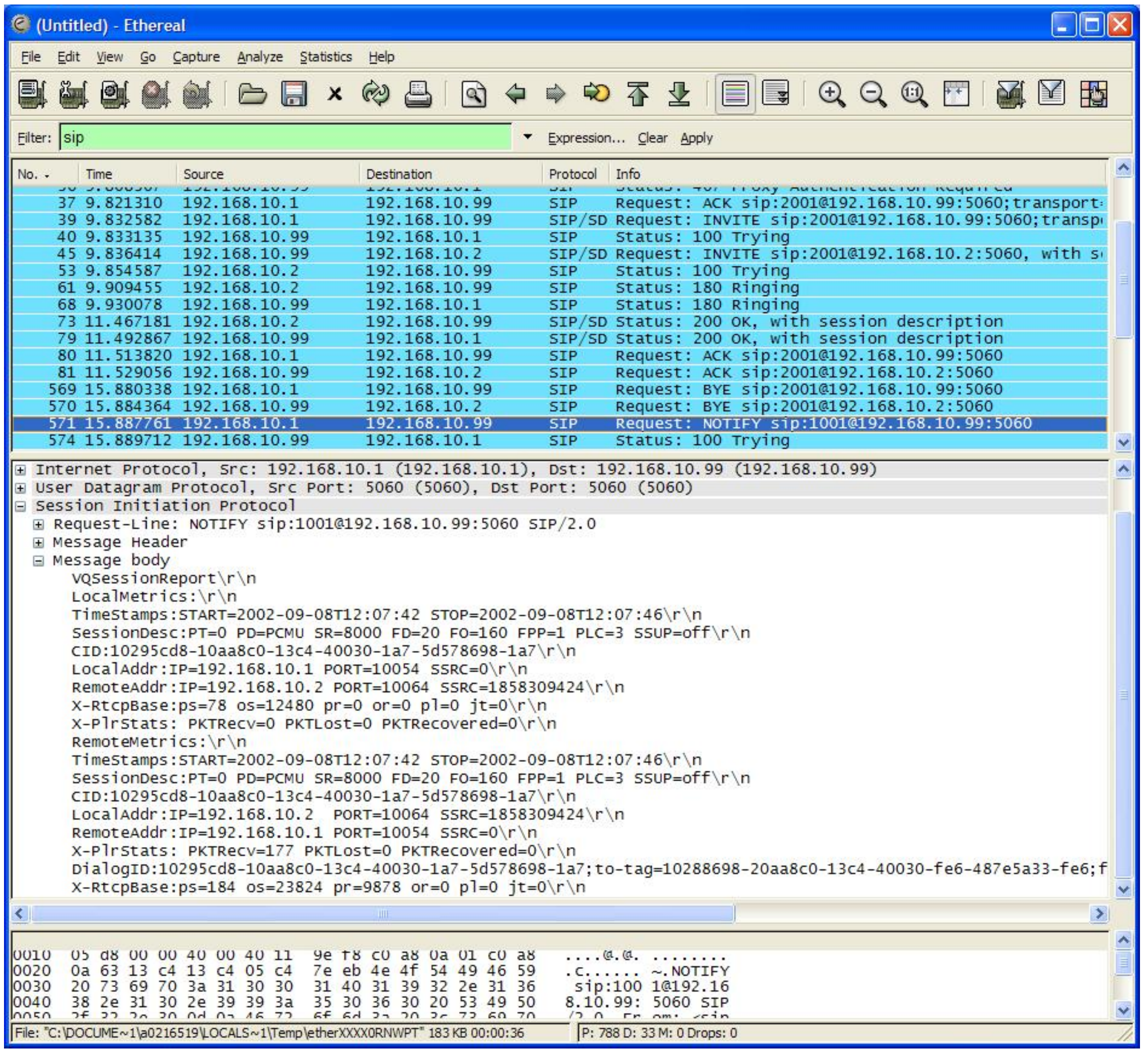


Time	192.168.10.1 192.168.10.99 192.168.10.2	Comment
4.588	ACK	SIP Request
4.599	INVITE SDP (g711U telephone-event g711A g721 ...	SIP From: sip:1001@192.168.10.99:5060 To:sip:2000@192.168.10.99:5060
4.600	100 Trying	SIP Status
4.611	INVITE SDP (g711U telephone-event g711A g	SIP Request
4.630	100 Trying	SIP Status
4.682	180 Ringing	SIP Status
4.704	180 Ringing	SIP Status
5.991	200 OK SDP (g711U telephone-event g711A g	SIP Status
6.017	200 OK SDP (g711U telephone-event g711A g721 ...	SIP Status
6.043	ACK	SIP Request
6.048	ACK	SIP Request
6.076	RTP (g711U)	RTP Num packets:47 Duration:0.458s ssrc:-941767806
6.081	RTP (g711U)	RTP Num packets:458 Duration:4292.926s ssrc:-1522333128
6.544	RTP (telephone-event) DTMF Five 5	RTP Num packets:9 Duration:4288.433s ssrc:-941767806
0.011	RTP (g711U)	RTP Num packets:1 Duration:0.000s ssrc:-941767806
0.021	RTP (g711U)	RTP Num packets:1 Duration:0.000s ssrc:-941767806
0.030	RTP (g711U)	RTP Num packets:47 Duration:0.459s ssrc:-941767806
0.500	RTP (telephone-event) DTMF Five 5	RTP Num packets:9 Duration:0.080s ssrc:-941767806
0.581	RTP (g711U)	RTP Num packets:1 Duration:0.000s ssrc:-941767806
0.590	RTP (g711U)	RTP Num packets:1 Duration:0.000s ssrc:-941767806
0.600	RTP (g711U)	RTP Num packets:9 Duration:0.079s ssrc:-941767806
0.690	RTP (telephone-event) DTMF Five 5	RTP Num packets:9 Duration:0.070s ssrc:-941767806
0.761	RTP (g711U)	RTP Num packets:1 Duration:0.000s ssrc:-941767806
0.770	RTP (g711U)	RTP Num packets:1 Duration:0.000s ssrc:-941767806
0.780	RTP (g711U)	RTP Num packets:12 Duration:0.109s ssrc:-941767806
0.910	RTP (telephone-event) DTMF Five 5	RTP Num packets:9 Duration:0.070s ssrc:-941767806
0.981	RTP (g711U)	RTP Num packets:1 Duration:0.000s ssrc:-941767806
0.990	RTP (g711U)	RTP Num packets:1 Duration:0.000s ssrc:-941767806
1.000	RTP (g711U)	RTP Num packets:297 Duration:2.959s ssrc:-941767806
3.991	BYE	SIP Request
4.021	BYE	SIP Request
4.057	200 OK	SIP Status
4.084	200 OK	SIP Status

Figure 14 -- Wireshark "call flow"

The screen capture from Wireshark clarifies which messages are involved with placing a regular phone call. This shows the “INVITE” and the “BYE” and a series of “RTP” packets (g711 uLaw) exchanged during the call (including some DTMF tones which were received from the far end). The “200 OK” messages are returned from most of the events and is passed thru by the call proxy to the other end point.

The final exchange of statistics takes place after the “BYE” message is received.


(Untitled) - Ethereal
File Edit View Go Capture Analyze Statistics Help
Filter: sip
Expression... Clear Apply

```
No. -  Time      Source          Destination     Protocol Info
 37 9.821310  192.168.10.1    192.168.10.99   SIP      Request: ACK sip:2001@192.168.10.99:5060;transport
 39 9.832582  192.168.10.1    192.168.10.99   SIP/SD   Request: INVITE sip:2001@192.168.10.99:5060;transp
 40 9.833135  192.168.10.99   192.168.10.1    SIP      Status: 100 Trying
 45 9.836414  192.168.10.99   192.168.10.2    SIP/SD   Request: INVITE sip:2001@192.168.10.2:5060, with s
 53 9.854587  192.168.10.2    192.168.10.99   SIP      Status: 100 Trying
 61 9.909455  192.168.10.2    192.168.10.99   SIP      Status: 180 Ringing
 68 9.930078  192.168.10.99   192.168.10.1    SIP      Status: 180 Ringing
 73 11.467181 192.168.10.2    192.168.10.99   SIP/SD   Status: 200 OK, with session description
 79 11.492867 192.168.10.99   192.168.10.1    SIP/SD   Status: 200 OK, with session description
 80 11.513820 192.168.10.1    192.168.10.99   SIP      Request: ACK sip:2001@192.168.10.99:5060
 81 11.529056 192.168.10.99   192.168.10.2    SIP      Request: ACK sip:2001@192.168.10.2:5060
569 15.880338 192.168.10.1    192.168.10.99   SIP      Request: BYE sip:2001@192.168.10.99:5060
570 15.884364 192.168.10.99   192.168.10.2    SIP      Request: BYE sip:2001@192.168.10.2:5060
571 15.887761 192.168.10.1    192.168.10.99   SIP      Request: NOTIFY sip:1001@192.168.10.99:5060
574 15.889712 192.168.10.99   192.168.10.1    SIP      Status: 100 Trying
```

```
⊞ Internet Protocol, Src: 192.168.10.1 (192.168.10.1), Dst: 192.168.10.99 (192.168.10.99)
⊞ User Datagram Protocol, Src Port: 5060 (5060), Dst Port: 5060 (5060)
⊟ Session Initiation Protocol
  ⊞ Request-Line: NOTIFY sip:1001@192.168.10.99:5060 SIP/2.0
  ⊞ Message Header
  ⊟ Message body
      VQSessionReport\r\n
      LocalMetrics:\r\n
      TimeStamps:START=2002-09-08T12:07:42 STOP=2002-09-08T12:07:46\r\n
      SessionDesc:PT=0 PD=PCMU SR=8000 FD=20 FO=160 FPP=1 PLC=3 SSUP=off\r\n
      CID:10295cd8-10aa8c0-13c4-40030-1a7-5d578698-1a7\r\n
      LocalAddr:IP=192.168.10.1 PORT=10054 SSRC=0\r\n
      RemoteAddr:IP=192.168.10.2 PORT=10064 SSRC=1858309424\r\n
      X-RtcpBase:ps=78 os=12480 pr=0 or=0 pl=0 jt=0\r\n
      X-PlrStats: PKTRecv=0 PKTLost=0 PKTRecovered=0\r\n
      RemoteMetrics:\r\n
      TimeStamps:START=2002-09-08T12:07:42 STOP=2002-09-08T12:07:46\r\n
      SessionDesc:PT=0 PD=PCMU SR=8000 FD=20 FO=160 FPP=1 PLC=3 SSUP=off\r\n
      CID:10295cd8-10aa8c0-13c4-40030-1a7-5d578698-1a7\r\n
      LocalAddr:IP=192.168.10.2  PORT=10064 SSRC=1858309424\r\n
      RemoteAddr:IP=192.168.10.1 PORT=10054 SSRC=0\r\n
      X-PlrStats: PKTRecv=177 PKTLost=0 PKTRecovered=0\r\n
      DialogID:10295cd8-10aa8c0-13c4-40030-1a7-5d578698-1a7;to-tag=10288698-20aa8c0-13c4-40030-fe6-487e5a33-fe6;f
      X-RtcpBase:ps=184 os=23824 pr=9878 or=0 pl=0 jt=0\r\n
```

```
0010  05 d8 00 00 40 00 40 11  9e f8 c0 a8 0a 01 c0 a8   ....@.@. ........
0020  0a 63 13 c4 13 c4 05 c4  7e eb 4e 4f 54 49 46 59   .c...... ~.NOTIFY
0030  20 73 69 70 3a 31 30 30  31 40 31 39 32 2e 31 36    sip:100 1@192.16
0040  38 2e 31 30 2e 39 39 3a  35 30 36 30 20 53 49 50   8.10.99: 5060 SIP
```

File: "C:\DOCUME~1\a0216519\LOCALS~1\Temp\etherXXXX0RNWPT" 183 KB 00:00:36 | P: 788 D: 33 M: 0 Drops: 0

3.2.7. PIQUA Statistics: SIP Notify Report

After a call completes, the error_log can be displayed as a means of serializing the statistics in a report format for capture in a terminal window.

<incomplete section>

3.2.8. PIQUA Statistics : Telephone Line Condition

Displaying the information about the telephony interface can be done from within the terminal emulator; Note: use the NMMThinCli.EXE program if the box has R12.0 software or later.

```
MXP> tiu test 0 start renld
MXP> OK

0014070965 - TIU: 0, Test renld result: REN LOW
0014070965 - Test renld raw data: 00000000
```

With no phones connected on LINE1:

```
MXP> tiu test 0 start renld
0014073555 - TIU: 0, Test renld result: Passed
0014119705 - TIU: 0, Test renld result: REN LOW
0014119705 - Test renld raw data: 00000039
```

With a single telephone connected on LINE1:

```
MXP> tiu test 0 start renld
MXP> OK
0000090300 - TIU: 0, Test renld result: Passed
0000090300 - Test renld raw data: 00000211
```

With five telephones connected on LINE2:

```
MXP> tiu test 1 start renld
MXP> OK
0000106814 - TIU: 1, Test renld result: Passed
0000106814 - Test renld raw data: 00000567
```

These commands executed with a Legerity (now, Zarlink) designed TID card (which uses a LE88241) with one telephone connected on LINE1:

```
MXP> tiu test 0 start femf
MXP> OK
0000097139 - TIU: 0, Test femf result: Passed
0000097139 - Test femf raw data: 00000000 00000000 00000000 00000000 00000000
00000000
```

The same command for the Legerity 88241 with five phones connected on LINE1:

```
MXP> tiu test 0 start femf
MXP> OK
0000100137 - TIU: 0, Test femf result: Passed
0000100137 - Test femf raw data: 00000000 00000387 00000387 00000000 00000000
00000000
0000100137 - TIU: 0, Test hazvolt result: Passed
```

Note: results from the TIU command are associated with the information which is displayed when the hardware is initially brought up at boot time, such as:

```
Hw Info for Fxs:
Type=1,Country=1,RingType=1,RingVolt=55,RingFreq=20
RingDCOfset=0,OnHookVolt=48,OffHookCurr=23
Hw Info for Fxo:
Type=2,Country=1,TipRingVolt=48,MinLoopCurr=20
Gain Info:
OnHookRx=-6,OnHookTx=6,OffHookRx=-6,OffHookTx=6
```

3.2.9. Seeing the effect of “G.711 App I and App II”

The ability to enable packet loss compensation, and also silence detection and comfort noise generation is build in with many of the vertical markets supported by TI’s Telogy software; it is “in the image” for most of the residential gateways builds and just be turned on using the appendices to G.711 with modifications in the provisioning.

4. Applying VoIP Statistics to Real World Scenarios

The benefit of having VoIP statistics is that an objective measure of voice quality can lead to diagnosis and repair of real problems experienced in VoIP Telephony systems. The most common problems can be summarized as:

- Garbled Voice
  - Mechanical Sound (unnatural playout)
  - Gaps in speech (lost data, dropped packets)
- Unnatural silence between voice transmissions
  - Voice Actitivity Detector (not working)
- Echo
- One-way audio
- Dropped calls

Some of the above list are slight variations of each other, and often a problem with the audio path will experience multiple problems which occur simultaneously – such that diagnosis is hard to describe accurately and succinctly. Of these, echo is the hardest to characterize, and the experience of acoustic echo being so common in IP Phones and cell phones that it distorts our impression of what causes echo. Line echo is specific to the electrical characteristics of the hybrid in the telephony network; acoustic echo occurs when the microphone picks up the audio transmission of the person speaking and re-sends it to the far-end listener – it is very different from the echo which is just inherent from the electronic components used in switching over to a four-wire circuit to send and receive the audio.

Associated with each of these subjective assessments of problems with VoIP can be multiple statistics which may be used to flag problems as they occur. In fact, as quantifing of the key parameters can be achieved even small degredations can be detected before the problem is noticeable to the human ear.

For example, for each of the general areas of problems experienced by the participants of a call the table summarizes which statistics would indicate the problems are being noticeable to the human ear.

Problem	**Explanation(s), of possible cause to the problem**
Garbled voice / Jitter	Packet arrival variation exceeds buffer size of "jitter buffer"
Garbled voice / DataLoss	Bandwidth available drops below real-time max thruput required based on vocoder
Unnatural silence	VAD not working
Echo	Line Echo canceller is mis-configured, or disabled

The thresholds at which each problem becomes detectable is as follows.

Problem	**Threshold**
Jitter	Packet Arrival Time Variation > Max Jitter Buffer
DataLoss	Throughput < 64kB/sec (G.711), 8kB/s (G.729AB)
Unnatural Silence	VAD (on/off)
Echo	EQI < 0.6

The remedies to each problem.

Problem	**Solution**
Jitter	Increase Jitter Buffer size Enable Adaptive Jitter Buffer Improve packet network
DataLoss	Switch to Low-bit-rate vocoders
Unnatural Silence	Disable VAD
Echo	Enable EC, and/or Optimize Echo Cancellation parameters

Illustrating the degredation, and application of the solutioin:

5. *Summary, conclusions, and practical guide to leveraging PIQUA features into a new VoIP design…*

Frequently Asked Questions

Here are few of the miscellaneous questions which have come up from working with customers doing designs for VoIP systems with PIQUA features.

FAQ #1 What does PIQUA stand for?

It doesn't stand for anything. It's not a derivative of some other word. It's just meant to be a unique easy to remember word rather than an acronym. Officially, it's properly used with all caps rather than with a capital "P" and lower case "iqua", i.e. PIQUA rather than Piqua. The first occurance of the trademark in any given document is meant to be used as "PIQUA^{TM} Software" from Texas Instruments.

FAQ #2 Are standards adhered to… or is PIQUA proprietary?

The RFC3611 specifies the "XR packets" and the structure of the packets which contain the statistics – how they get used and whether service providers, equipment manufacturers or software suppliers are responsible to handle and present the data is still within the realm of the control of the industry and the evolution of the VoIP market.

Additions are made, such as Echo Quality index, or PLC values which TI has defined the way to determining values for the additional fields within the standard packet structure. There are also many areas where PIQUA offers voice quality improvement which are not just limited to exchange of packets with statistics – some of these may be specific to a single vendor, or group of vendors. There are other existing standards already in place for telecommunications which PIQUA can work within, for example TR069 for line testing and control – PIQUA enables voice quality information to supplement the 069 monitoring.

FAQ #3 Is PIQUA a TI concept, or an industry term?

Although PIQUA is a trademark of Texas Instruments, the concepts are based on industry standards and TI's role in promoting the voice quality enhancements is specifically focused on embedded software systems and features which make it possible for service providers

FAQ #4 Which statistics are important?

There are a variety of statistics; some are more commonly cited that others – in fact, part of the problem with determining how to leverage the data available is that there are so many metrics to choose from. But, a MOS score gives direct association with a listener experience; the higher MOS correlates to a better quality call. Even here though, there are multiple choices – there are two MOS (or more) values which are commonly referenced: LQ and CQ, for Listener Quality and Conversational Quality – where Conversational Quality gives more of an indication of the ability to carry on a conversation, rather than just the instantaneous values of the LQ indicator. Note that CQ will quickly reflect a change in the MOS if there is latency in the network.

Additional statistics include: packet count and packet loss to indicate how many total packets are transmitted and what percentage are lost. In the TI implementation there are also metrics for the Echo Canceller’s Quality Indicator we refer to as EQI. The EQI indicates the effectiveness of the line echo canceller (locally) although it is more of an indication of voice quality which would be experienced by the far-end listener.

An “XR” report, or more formally RTCP-XR for eXtended Report is described in RFC3611. This includes MOS values, Delay measures, Packet loss rate and discard rates, as well as Burst densities and Gap densities for describing pattern of packet loss. There are also signal and noise levels, and echo characterization. All of the end-to-end detail of a VoIP call is encoded into an additional packet which is added to the voice stream between two end-points.

FAQ #5 How do endpoints know where to send the statistics?

Two end points in a VoIP conversation exchange XR packets between them. They just simply include XR packets periodically along with the other RTP payloads of the voice data. A monitor agent (like, from an NMS vendor) which resides on the same subnet can recognize the XR packets and summarize them and forward call details to a server on another network.

Additionally, a SIP Notify QoS Report can be sent to an IP / UDP address indicated by the SIP configuration at each endpoint; this is usually done at the end of a call. Service providers need to direct the client endpoints to use their own specific server destinations and a NMS will provide the facilities necessary to collect and interprete the information for a ISP of calls.

For monitoring systems which depend upon being present on the same subnet as the endpoint under observation it is not important to have endpoints configured to send info to them explicitly – they monitor the values of traffic transparently. The only down-side to this configuration is the proximity of the end-points to the monitor software.

FAQ #6 What is contained in a SIP Notify Report?

The end-of-call statistics are gathered by the microprocessor at an endpoint, or in any gateway along the voice-path, and the fields populated with the quality metrics which are available at that point in the voice path – likely the statistics are dependent upon information which was collected during the call by the DSP which does the voice processing.

FAQ #7 Do PIQUA features impact channel density of a gateway?

No, most of the processing of the call metrics is done in concert with the normal processing which is handling regular call traffic. The number of XR packets generated during a call is a small percentage of the total packet traffic, and the SIP Notify messages are only generated at the end of the call.

Although there are a certain number of MIPS consumed in calculating some of the metrics (like, the MOS score which is the result of executing the Markov model on collected DSP statistics), it is a relatively small computation which is incorporated into the full channel MIPS budget and fits with the total computation requirements for each channel. It varies by each application and device; but in almost every case the total channel density can be accommodated without adversely affecting the channel density.

FAQ #8 How much does a PIQUA feature cost per channel?

A TI Gateway device (such as the TNETV1061 device) is bundled with silicon and software into the same part – the PIQUA features are included.

FAQ #9 Do PIQUA enabled units work with non-PIQUA units?

The ideal configuration is for two enabled end points to communicate with each other; but, even with just local client providing call quality metrics the improvements can be done at least at one end – independent of the opposite end of the conversation.

Note: the RTCP-XR packet content allows local end-point to inform the remote unit of its current VoIP Quality, without these fields populated the paired endpoint is not aware of the remote unit’s data. However, the benefit can be achieved at least for the single client which has the PIQUA capabilities – it may be the case (just like for a line echo canceller in a VoIP gateway) that the remote party benefits from the PIQUA feature even if they don’t have PIQUA.

FAQ #10 Do XR packets get transmitted only at the end of a call?

RTCP-XR packets are transmitted according to the specification; normally it is at a regular interval, between five and fifteen seconds while a call is active. The XR packet is also normally sent at the end of a call; at the end-of-a-call the packet can be encapsulated within a SIP or MegaCo or other protocol as desired for the application.

FAQ #11 Do XR packets get transmitted by only one client?

RTCP-XR packets are transmitted from both clients if the end point has the XR feature enabled; it may appear that the data is only generated by one end if the route to the NMS is not visible to both clients, and/or if both clients do not have the ability to generate XR content. An XR packet is normally exchanged by the end points, if a “SIP NOTIFY” is used to forward the data to an NMS explicitly then the data may be routed to an address by one end point or the other.

FAQ #12 Is a Call Agent needed for the PIQUA features to work?

No, the call agent doesn’t need to get involved with quality assessment. In fact, most equipment manufacturers would want to deliver call control and PIQUA features as a separate hosted operation – they don’t need to relate to each other.