A Voice over IP Quality Monitoring Architecture

Leandro C. G. Lustosa, Paulo H. de A. Rodrigues, Fabio David,
Douglas G. Quinellato

Laboratório de Voz Sobre IP, Núcleo de Computação Eletrônica,
Universidade Federal do Rio de Janeiro (NCE/UFRJ)*
Caixa Postal 2324, 20001-970, Rio de Janeiro, RJ, Brasil
{leandro, aguiar, fabio, douglasq}@nce.ufrj.br
http://www.voip.nce.ufrj.br

Abstract. A voice over IP quality monitoring architecture based on the development of the VQuality library is described. VQuality implements the E-model and its extensions for objective voice quality measurement. The library also supports generation of a customized voice quality CDR with extensions that permit transfer of call quality parameters measured over different instants of time, besides interacting with a RADIUS server for data collection. The framework is exemplified in the context of the fone@RNP VoIP service and VoIP clients incorporating the new functionality are shown.

1 Introduction

Voice over IP communication systems have been proliferating and represent a viable technological solution for voice provisioning. Aligned with this tendency, the Brazilian Education and Research Network (RNP) started to offer a VoIP service over its backbone. The service, named fone@RNP, is based on a heterogeneous (H.323 and SIP) VoIP architecture, which supports voice calls among institutional PBXs, call establishment with the public service telephony network (PSTN), and interaction with other international academic and research H.323 and SIP initiatives.

The environment of a fone@RNP institution partner is composed of servers (H.323 GnuGK [1] or SIP Proxy SER [2]) for VoIP client registration and location. Client authentication is done consulting an LDAP directory [3] via RADIUS [4] and call statistics accounting records are stored in an SQL database. An optional gateway can be used for interconnection with PBX, as shown in Fig. 1. Cisco routers and Asterisk [5] are used as PBX gateways, while Asterisk is also used as an H.323/SIP signaling gateway. Since E.164 addressing is directly supported by the H.323 architecture, SIP servers route E.164 calls to H.323 server, if the call is not destined to the local gateway, which means not destined to the institutional PBX or local PSTN. There is an extensive use of IP clients, like Openphone for H.323 [6] and X-Lite for SIP [7].

*Partially supported by RNP Advanced VoIP Working Group. Paulo H. de A. Rodrigues is also a professor in the Computer Science Department/IM at UFRJ.

Voice call quality statistics represent valuable information for system performance management, helping to validate admission control schemes and changes in network QoS configuration, besides enabling faulty service behavior detection and establishing parameters for monitoring user satisfaction.

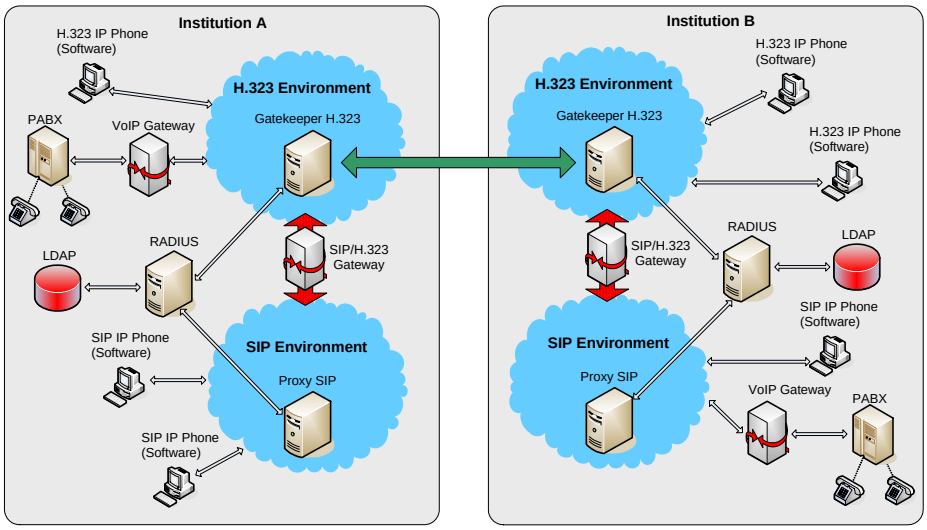

Fig. 1. fone@RNP service architecture

As shown in Fig. 1, accounting is performed with collection of call detailed records (CDR) generated by servers and gateways. However, only Cisco gateways give some call completion quality indication. Calls between clients (IP telephones or softphones) have no quality indication and the overall collected statistics are insufficient for precise system characterization.

CESNET, the Czech education and research network, has also implemented a CDR accounting and monitoring structure using RADIUS and gateways [8]. In [9], the *Enterprise Call Analysis System* (ECAS) architecture is presented, also based on the analysis and collection of gateway CDRs. These and similar architectures suffer from the same limitation of not taking in account the quality of calls between VoIP clients.

In [10], to monitor the call quality offered by a network, distributed SNMP agents periodically generate simulated calls and collect packet losses, delay and jitter. This solution does not address the collection of user calls statistics and the sampling procedure may not reflect the real network impact on voice calls. In [11], as in many commercial products [12,13,14], a passive monitoring solution is used. Probes able of capturing and analyzing voice flows are strategically placed in the network and generate quality reports. This active scheme is not always successful, because secure communication may block packet interpretation and analysis, or the point to point voice traffic nature may force packets to be routed away from the probes.

An alternate solution is been pursued in the fone@RNP service, in order to enable a more extensive and trustable quality accounting. A library, called Voice Quality

(VQuality lib), capable of voice quality evaluation and quality CDR (VQCDR) generation, was developed and is being incorporated to softphones. This library implements the E-model [15] and its extensions [16,17]. Due to the use of the extended E-model, the generated voice quality indicators are extremely detailed and closer to human perception than those indicators generated by present VoIP gateways [18,19].

The remaining of this paper is organized as follows. In section 2, we describe the VQuality lib and corresponding VQCDR. In section 3, the architecture for CDR collection is presented. Finally, conclusions and future work are presented in section 4.

2 Voice Quality Library (VQuality lib)

One of the main efforts to support the deployment of fone@RNP has been the characterization and evaluation of voice call quality. Our first step towards this goal was the creation of an environment based on a module called MOBVEM (Modified OpenH323 Based Voice Evaluation Module) [16]. MOBVEM uses a modified Open H323 library for obtaining voice call detailed log with all the parameters needed for computing the E-model [15] and its extensions [17].

Although MOBVEM has all the necessary requirements for implementing a voice quality measurement tool, it was developed in Perl [20] and lacks the performance of a compiled language. Additionally, it does not offer an API to permit its integration to other software and is limited to H.323 signaling. Furthermore, MOBVEM needs to simultaneously analyze the logs generated by both ends of a call to estimate RTT, making real-time voice quality determination unfeasible, when one of the ends of a call is a client that does not use the modified OpenH323 library

To overcome MOBVEM limitations, VQuality library was developed. It is written in C++ and inherits the calculation of the extended E-model. Moreover, besides its superior performance, VQuality is flexible, extensible, portable, and its computation of received voice quality is independent of the other side of the call.

VQuality was developed under the oriented object paradigm and conceived for easy addition of new evaluation models or VoIP signaling protocols. Portability is achieved with a standard C/C++, except for TCP sockets and threads, which are implemented differently in each OS. We have implemented our own TCP sockets function, compatible with Windows and Unix systems. For thread handling, we use the Pthreads-win32 library [21], which allows code compilation based on Pthreads API, standard for Unix OS. The result is a code which is compilable in Linux, FreeBSD and Microsoft Windows, besides being portable to other architectures, if necessary.

A major challenge was to modify OpenH323 to make it full compliant with RTP and fill in all required fields. Integration of VQuality with OpenH323 lib permitted the creation of H.323 clients with VQCDR capability (see section 3.4).

3 Collection Architecture

One of the most important features of VQuality is its ability to send Voice Quality Call Detailed Record (VQCDR). At call completion, the library computes the quality of the received voice media, processes identification parameters for the call and involved terminals, and reports this information to a centralized entity, responsible for collecting CDRs. VQCDR is sent using TCP/port 80, by default. TCP assures reliable transfer and port 80 (HTTP) facilitates operation behind firewalls, which rarely block this port number. VQCDR format and attributes are shown in Table 1.

Table 1. VQCDR fields

#	Field	Length (bytes)	Description
01	version	1 *	VQCDR version. Present version is 1.
02	signalingProtocol	1 *	Assigned: 0(H323), 1(SIP), 2 (MGCP).
03	IDSize	2 *	ID field length in bytes.
04	ID	0-65535	Call identification. In H.323 represents ConfID, while in SIP is Call-ID.
05	username	64 †	User identification.
06	model	1 *	Assigned: 0 (ITU-T E-model), 1(ETSI Extended E-model), 2 (UFRJ/UFAM Extended E-Model)
07	codec	1 *	Assigned: 0 (G.711), 1 (G.711 PLC), 2 (G.723.1 5.3kbps), 3 (G.723.1 6.3 kbps), 4 (G.726 16 kbps), 5(G.726 24 kbps), 6 (G.726 32 kbps), 7 (G.726 40kbps), 8 (G.728 16 kbps), 9 (G.729), 10 (G.729A), 11 (GSM FR (6.10)).
08	frameSize	4 *	Voice frame size in micro seconds.
09	framesPerPacket	1 *	Number of voice frames in IP packet.
10	IdEndOfCall	2 *	*Delay Impairment* (Id). Degradation indicator related to end to end delay and interactivity. Value times 100.
11	IeEndOfCall	2 *	*Equipment Impairment* (Ie. Degradation indicator related to high compression codecs, network packet losses and jitter buffer discards. Value times 100.
12	MOSEndOfCall	2 *	*Mean Opinion Score* (MOS) at call completion. Represents perceived human quality. Value times 100.
13	gapLossDensity	2 *	Loss density in isolated losses (gap). Value times 100.
14	burstLossDensity	2 *	Loss density in bursty losses (burst). Value times 100.
15	discardRate	2 *	Percentage of discarded packets in the jitter buffer. Value times 100.
16	lossRate	2 *	Percentage of loss packets. Value times 100.

17	RfactorEndOfCall	2 *	R factor at call completion (output from E-model). Value times 100.
18	duration	4 *	Call duration in seconds.
19	avgNetDelay	4 *	Network average delay in μ seconds. See note 1.
20	avgJitterBufferDelay	4 *	Jitter buffer average size in μ seconds. See note 2.
21	avgJitter	4 *	Average jitter in micro seconds. See note 3.
22	codecDelay	4 *	Coding and packetization delay in micro seconds.
23	packetsReceived	4 *	Total received packets.
24	packetsLost	4 *	Total lost packets.
25	packetsDiscarded	4 *	Total discarded packets in the jitter buffer.
26	mediaSource	256 †	Remote client address (IP or hostname), media sender.
27	mediaDestination	256 †	Local client address (IP or hostname), media receiver.
28	localAlias	64 †	Id of client emitting VQCDR. E.164 number or URI.
29	remoteAlias	64 †	Id of remote IP client. E.164 number or URI.
30	direction	1 *	Identification of call origin: 0 (called), 1 (calling)
31	appExtensionSize	1 *	Length of appExtension in bytes. 0 if field is absent.
32	appExtension	0-255 †	Application additional information.
33	VQLogSize	2 *	VQLog field length in bytes. 0 if field is absent.
34	VQLog	0-65535‡	Detailed quality log in VQLog format See sec. 3.5.

* Unsigned integer value (more significant byte first). ‡ Text file.

† ASCII coded text with delimitator (hex 00).

Note 1: Arithmetic average of the 16 last network delay calculations (netDelay).

Note 2: Arithmetic average of the 16 last jitter buffer delay calculations. If the length of the jitter buffer is not dynamic, this value is constant.

Note 3: Arithmetic average of the 16 last jitter variations in RTP format [22].

2.1 Voice Quality CDR Server (VQCDR Server)

The VQCDR Server is the central entity responsible for collecting VQCDRs, checking its legitimacy and sending them for storage. Server interacts with a database or application to check if VQCDR was generated by a valid user/client. VQCDR server architecture is composed of three modules, as shown in Fig. 2.

Collector Module (CM): responsible for collecting and interpreting received VQCDR, and also responsible for activating the Authenticator Module (AM) and the Storage Module (SM).

Authenticator Module (AM): responsible for VQCDR validation. VQCDR Server can operate with zero or more AMs. For test or controlled environments, a simple IP address access list can be used to authenticate the VQCDR sender. However, in production environments, where a more sophisticated and flexible authentication procedure is required due to scalability reasons, use of an AM is more appropriate. When operating with SIP and H.323, for example, it may be necessary to access a specific AM for each signaling protocol. In this case, CM has to be capable of selecting the right AM. The VQCDR protocol field can be handy for this purpose.

Storage Module (SM): responsible for storing the collected VQCDRs in a database. This database can be, for example, an SQL database or a RADIUS server.

For the H.323 fone@RNP service, we have implemented a VQCDR Server with an AM specific for the GnuGK gatekeeper and an SM based on RADIUS. VQCDR generation in SIP clients have not been implemented yet.

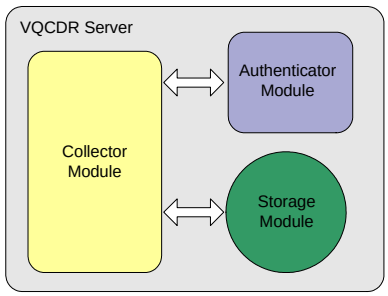

Fig. 2. VQCDR Server Architecture

3.2 GnuGK Authenticator Module (GnuGK AM)

GnuGK AM uses GnuGK [1] remote management port for validating VQCDR originator (see Fig. 3-A). This port implements a communication channel which allows checking registered users and accessing user information in the system, such as: user identification, registration IP address (public IP address used by the user during registration) and private IP address (present if user is behind a NAT box [23]).

For the authentication procedure, AM checks username (user id) and mediaDestination (private IP address if user is behind a NAT box) fields in VQCDR and also the IP address of the packet received with CDR (registration IP address). In case user is behind a NAT box, VQCDR will be validated if and only if user identification, private IP and registration IP addresses match a GnuGk user active record.

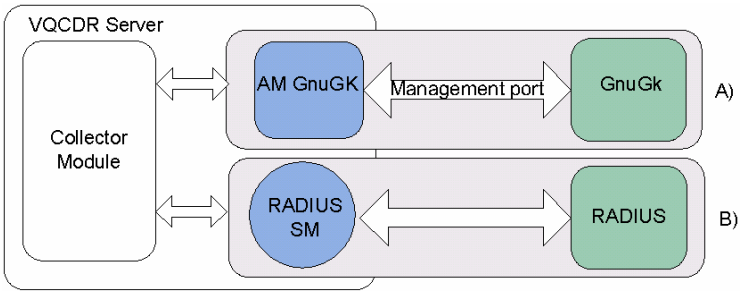

Fig. 3. GnuGK AM (A) and RADIUS SM (B)

3.3 RADIUS Storage Module (RADIUS SM)

RADIUS accounting offers a standard attribute set which is not enough to detail a voice call. However, RADIUS allows reporting extra attributes, application dependent, called VSAs (Vendor Specific Attributes). To use specific attributes, a Private Enterprise Number has to be solicited to IANA [24]. With this number, a vendor/application can define a data dictionary which allows RADIUS to interpret specific attributes. UFRJ VSAs were assigned data dictionary number 21715 by IANA.

# UFRJ Vendor Specific Attributes				discardRate	15	integer	ufrj
VENDOR ufrj 21715				lossRate	16	integer	ufrj
# VQCDR attributes				duration	18	integer	ufrj
signalingProtocol	1	string	ufrj	avgNetDelay	19	integer	ufrj
callStart	2	string	ufrj	avgJitterBufferDelay	20	integer	ufrj
callStop	3	string	ufrj	avgJitter	21	integer	ufrj
ID	4	string	ufrj	codecDelay	22	integer	ufrj
username	5	string	ufrj	packetsReceived	23	integer	ufrj
model	6	string	ufrj	packetsLost	24	integer	ufrj
codec	7	string	ufrj	packetsDiscarded	25	integer	ufrj
frameSize	8	integer	ufrj	mediaSource	26	string	ufrj
framesPerPacket	9	integer	ufrj	mediaDestination	27	string	ufrj
IdEndOfCall	10	integer	ufrj	localAlias	28	string	ufrj
IeEndOfCall	11	integer	ufrj	remoteAlias	29	string	ufrj
MOSEndOfCall	12	integer	ufrj	direction	30	string	ufrj
gapLossDensity	13	integer	ufrj	RFactorEndOfCall	31	integer	ufrj
burstLossDensity	14	integer	ufrj				
			1/2				**2/2**

Fig. 4. VQCDR RADIUS data dictionary

SM interface with RADIUS (see Fig. 3-B) was implemented with use of the RadiusClient library [25], which is only available in Unix. In the future, this library will be modified to make it Windows compatible and allow running RADIUS SM on this OS. Our specific data dictionary is shown in Fig. 4.

3.4 Modified H.323 Clients

Using a modified OpenH323 library, for implementing a full compliant RTP, and VQuality lib, for voice quality evaluation and VQCDR generation, three clients based on open source implementations have been developed: VQOpenphone, from Openphone, a graphics Windows client, VQMeeting, from Gnome Meeting [26], a graphics Unix client and VQOhphone, from Ohphone, a textual multiplatform client (Windows and Unix). Fig. 5 shows a VQOpenphone and VQMeeting clients at the end of a call. In their status panel, MOS for the received call is displayed.

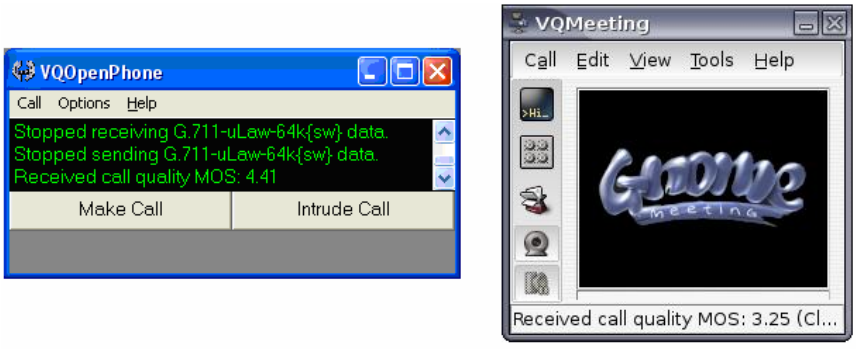

Fig. 5. VQOpenphone and VQMeeting clients

3.5 Voice Quality Log (VQLog)

VQCDR quality indicators offer a receiver perspective call quality summary, very adequate for a large VoIP system supporting hundreds or thousands of calls a day. However, this information does not allow an analysis of quality variation over time, which may be needed in tests or in more detailed and specific experiments.

```
! Example of VQLog file
G.711-uLaw-64k 1 20.00 40.00
13:41:00.625 6 0 0.00 98.00 13 116 0.15 22.76 70.50 3.62
13:41:01.547 3 0 0.00 98.00 18 116 0.15 23.62 69.60 3.58
13:41:02.469 3 0 0.00 97.50 15 115 0.15 20.46 72.80 3.73
13:41:03.375 5 0 0.00 97.50 18 115 0.15 20.37 72.80 3.73
13:41:04.391 0 0 0.00 97.50 14 115 0.15 20.37 72.80 3.73
13:41:05.391 0 0 0.00 97.00 15 114 0.15 20.37 72.80 3.73
13:41:06.375 0 0 0.00 97.00 18 114 0.15 20.37 72.80 3.73
13:41:07.365 0 0 0.00 97.00 18 114 0.15 20.37 72.80 3.73
2 0.00 57.50 14.00 40.00 367 17 0 0.00 11.26 0.15 20.42 72.80 3.73
0 D41F4102AAF118109FED0040A70 leandro 3377 3354 1 10.10.1.1 10.10.2.1 8
```

Fig. 6. VQLog file example

Need for more detailed measurements and motivation for sharing reports among different applications based on VQuality produced the VQLog format, which allows sending of extra information besides VQCDR indicators. To enable the timely analysis of a call, a file with values of main quality variables at different instants of time is included and sent with VQCDR.

A VQLog file is coded in ASCII and must have one or more codec identification lines, one or more quality indicators lines, one summary line and a call identification

line, necessarily in this order. Each line is formed by fields separated by blank spaces. During a call, one or more codecs can be used, needing one line for each. Each quality indicators line is associated with an instant of time, forcing sending many lines to characterize quality changes over time. A VQLog file example is shown in Fig. 6.

3.6 Voice Quality Plot (VQPlot)

VQPlot is an application which reads and plots VQLog information. Together with a client that uses VQuality, like VQOpenphone or VQMeeting, it becomes a very powerful analysis tool. At the end of a call, the user can run VQPlot. The application will automatically open the VQLog file and plot graphs showing the evolution of relevant parameters over time. These plots help to identify degradation factors for a call and its intensity. This analysis can help, for example, a technician give remote support to a user facing configuration problems with its network or personal computer.

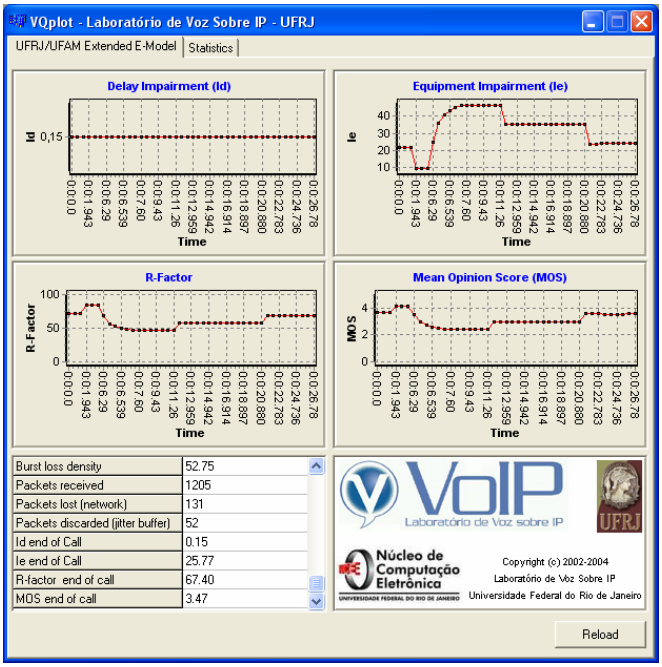

Fig. 7. Call quality over time

Fig. 7 and Fig. 8 illustrate VQPlot output for a call originated from a VQOpenphone running on a laptop with a 802.11b wireless lan connection.

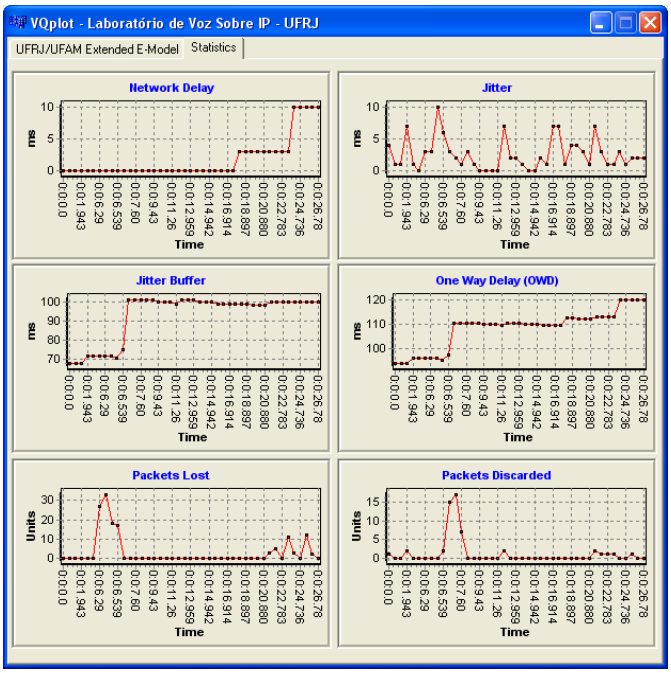

Fig. 8. Call statistics over time

3.7 Visualization Environment

All CDRs, generated by a server, a gateway or IP client (VQCDR), are sent to a RADIUS server and stored in an SQL database. Soon after, CDRs that refer to the same call are consolidated in one single record. A Web based interface (Fig. 9) generates reports such as: call distribution over a day, number of calls over a day period, call quality along a day, number of simultaneous calls, among others.

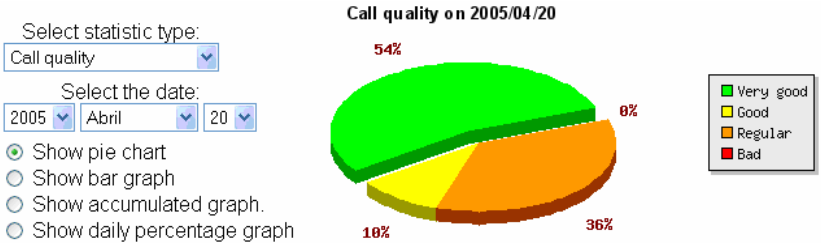

Fig. 9. Visualization Web environment for statistics reports

4 Conclusions and Future Work

In this paper, we presented a voice quality monitoring architecture which process voice quality CDRs (called VQCDR) generated by the VQuality library. VQuality implements objective voice quality evaluation based on the E-model and its extensions. This library, written in standard C/C++, is portable and supports different VoIP signaling protocols. Besides generating its VQCDR and sending it via RADIUS, it has extensions for sending complementary information as a textual file. An application called VQPlot is able to read this complementary information and plot graphs displaying parameter variation over time. Using VQuality and a modified OpenH323 lib, three clients, VQOpenphone, VQMeeting and VQOhphone, capable of generating voice quality CDRs, were developed.

One future step will seek the integration of VQuality lib into open source SIP client or in IP telephone under partnership with vendors. Another possible line of action would be the incorporation of VQuality to a simulation tool, as the Network Simulator [27], what could be helpful for the design and evaluation of complex VoIP systems.

When using VQCDRs and full compliant RTCP reports, it is possible to determine delays in each direction of a call. From the statistics database, as a future work, we could derive information about traffic asymmetry in the underlying network.

The lack of clients with the ability of measuring call quality should foster partnerships with vendors and providers, specially because of the increasing interest in VoIP and need for assuring and measuring user satisfaction.

References

1. OpenH323 Gatekeeper. The GNUGatekeeper: In://www.gnugk.org/. Accessed in May 2005
2. iptel.org SIP Server: SIP Express Router. In://www.iptel.org/ser. Accessed in May 2005
3. Hodges, J., Morgan, R.: Lightweight Directory Access Protocol (v3): Technical Specification . RFC 3377 (2002)
4. Rigney, C., Willens, S., Rubens, A., Simpson, W.: Remote Authentication Dial In User Service (RADIUS). RFC 2865 (2000)
5. Asterisk™ – The Open Source Linux PBX: In: //www.asterisk.org. Accessed in May 2005
6. OpenH323 Project: Available in http: //www.openh323.org/. Accessed in December. 2004
7. Xten X-Lite: Available in http://www.xten.com/. Accessed in May 2005
8. Ubik, S.: IP Telephony Accounting and WAN Deployment Experience. IPTEL 2001, USA (2001)
9. Lin, M., LO, C., e W., S.: Design and Implementation of an Enterprise Call Analysis System for VoIP Deployments. 2003 Australian Telecommunications, Networks and Applications Conference (ATNAC) (2003)
10. Huang, C., Chao, C., e Liu, A.: A Distributed Management Framework for H.323-Based VoIP System. Communications in Computing (CIC) 2003. USA, (2003)
11. Broom, S., Hollier, M.: Speech Quality Measurement Tools for Dynamic Network Management. Measurement of Speech and Audio Quality in Networks (MESAQIN) 2003 (2003)
12. Telchemy: SQmon - Service Quality Monitoring for Voice over IP. Available in: http://www.telchemy.com/sqmon.html. Accessed in May 2005

13. Empirix: Hammer XMS. Available in: http://www.empirix.com/Empirix/Network+IP+Storage+Test/hammer+xms.html. Accessed in May 2005
14. Brix Networks: Advanced VoIP Test Suites. Available in: http://www.brixnet.com/ products/voip_testsuite.html. Accessed in May 2005
15. ITU-T Recommendation G.107: The E-Model, a computational model for use in transmission planning. Switzerland (2003)
16. Lustosa, L.C.G., Carvalho, L.S.G., Rodrigues, P.H.A., Mota, S. E.: Utilização do Modelo E para avaliação da qualidade da fala em sistemas de comunicação baseados em voz sobre IP, XXII Simpósio Brasileiro de Redes de Computadores. Brazil (2004)
17. ETSI TS 102 024-5 v4.1.1. Telecommunications and Internet Protocol Harmonization over Networks (TIPHON) Release 4: End-to-end Quality of Service in TIPHON systems; Part 5: Quality of Service (QoS) measurement methodologies. France (2003)
18. Cisco Syst., RADIUS VSA Voice Implementation Guide: In: //www.cisco.com/univercd/cc/td/doc/product/access/acs_serv/vapp_dev/vsaig3.pdf. Accessed December 2004
19. Cisco Systems, Managing Voice Quality with Cisco Voice Manager (CVM) and Telemate: In: http://www.cisco.com/warp/public/788/AVVID/cvmtelemate.pdf. Accessed in Dec. 2004
20. The Perl Foundation. Perl Directory at Perl.org: In://www.perl.org/. Accessed in May 2005
21. PThreads-Win32. Open Source POSIX Threads for Win32: Available in http://sources.redhat.com/pthreads-win32/. Accessed in May 2005
22. Schulzrinne, H., Casnet, S., Frederick, R., Jacobson, V.: RTP: A Transport Protocol for Real-Time Applications. RFC 3550 (2003)
23. Srisuresh, P., Egevang, K.: Traditional IP Network Address Translator (Traditional NAT), RFC 3022 (2001)
24. Reynolds, J. e Postel, J.: Assigned Numbers, RFC 1700 (1994)
25. RadiusClient: Available in http://freshmeat.net/projects/radiusclient/. Accessed May 2005
26. Gnome Meeting: In://www.gnomemeeting.org/. Accessed in May 2005
27. The University Of Southern California: Network Simulator – ns2. Available in: http://www.isi.edu/nsnam/ns/. Accessed in May 2005