Research White Paper

WHP 170

October 2008

Standardising Audio Contribution over IP Communications

P. A. Stevens *and* M. Zemack

BRITISH BROADCASTING CORPORATION

BBC Research White Paper WHP 170

Standardising Audio Contribution over IP Communications

P. A. Stevens *and* M. Zemack

Abstract

Live audio contributions from reporters have traditionally been submitted using standard telephone lines via ISDN. With improvements in the Internet and other access IP technologies, such as wireless hotspots, reporters saw the potential for greater flexibility from where they could contribute their programme material. ISDN is gradually being replaced and so contributions would have to be transferred over IP. This brings its own problems, not the least being, that until recently, IP codecs[1] from different manufacturers would not communicate with other.

Based on an initiative from German vendors and broadcasters, the EBU[2] has created a standard for interoperability for live audio contribution over IP. One requirement was that manufacturer claims of interoperability could be independently tested. To achieve this, a reference implementation of the interoperability standard was developed jointly between BBC and IRT[3].

This paper describes an overview of the history and background of audio contribution over IP, together with the development of the reference implementation and some possible future developments within the audio over IP field. The reference implementation is based on open source code and is publicly available as an open source project.

This paper also includes a section on the use of the reference implementation in a 'Plug Test' that took place with manufacturers at IRT in February 2008.

Additional key words: RTP, SIP, SDP, UDP, VoIP, ACIP, NMC, aoip, G.711, G.722, MP2, PCM, PJSIP, Voice, Asterisk, Server, Outside Broadcast, TECH 3326, TECH 3329

---

[1] Codec – an amalgamation of audio (or video) coder and decoder, signifying dual purpose functionality.
[2] European Broadcasting Union, Geneva, Switzerland
[3] Institut für Rundfunktechnik GmbH, München, Germany

White Papers are distributed freely on request.

Authorisation of the Head of Broadcast/FM Research is required for publication.

BBC Research White Paper WHP 170

Standardising Audio Contribution over IP Communications

P. A. Stevens *and* M. Zemack

1 Introduction

Traditionally, live audio contributions from reporters based in the 'field' at sports grounds, churches and other locations away from fixed studios, have been submitted using (in the main) temporarily leased lines from a service provider over ISDN. In recent times, there has been a move towards using Internet Protocol (IP) technology as an alternative, due to its increased ease and availability of use and lower costs. This has presented both new opportunities and problems which are now being realised and overcome.

2 Background

2.1 ISDN closedown!

For the past 15 or so years, Integrated Services Digital Network (ISDN), a fully digital communication technology, has been the backbone of the world wide telephone network. As such, it uses standard telephone lines consisting of copper wire pairs, having converted the analogue signals into the digital domain, capable of transmitting audio, video and text data simultaneously. ISDN provides multiple channels operating concurrently on the same pair of wires, with each channel capable of transmitting at 64 Kbps.[4] It offers low latency point-to-point, bi-directional communication with guaranteed data and error rates over a synchronous circuit.

An Outside Broadcast (OB) vehicle transmitting live from a church somewhere in the UK may well feed the audio back to studio via an ISDN line. (Other methods such as microwave links may also be employed, if applicable, for example.). Alternatively, if a sports event abroad is to be broadcast, the main contribution is through ISDN.

ISDN lines are not expensive as such; a contribution feed usually costs about the same as two landline calls. If ISDN is used for contribution of an overseas tournament, the cost of the tariffs is high compared to the low cost of an ADSL or other broadband connection often already in place for other uses.

In the early days of ISDN, contribution equipment from different manufacturers would not communicate with each other over the ISDN network, despite there being a 'standard'. Today however, this is no longer the case and most ISDN equipment will interoperate.

Unfortunately for ISDN however, another revolution has been taking place with the massive increase in the use of computer networks operating over Internet Protocol (IP), with the development of the Internet and the use of satellite and wireless technologies. This has reduced costs by such a huge amount that countries that are still developing their communications infrastructure install only data cables, fibre and/or wireless technologies, with bandwidths of 100Mb/s and upwards. Countries with an existing communications infrastructure are taking advantage of the reduced costs by upgrading and moving away from ISDN, which has been gradually becoming more expensive to install, operate and maintain for operators and users alike. As a result, some countries are to cease using ISDN altogether. Those that are not ceasing to use it yet, the service providers are making it increasingly more expensive. Either way, users see the

[4] Once the signals reach an exchange, they are multiplexed with other user's signals and may well be carried over other physical medium, such as fibre within the core of the telephone network.

lower costs of IP and the variety of access methods now available more attractive and consider that this is the direction in which to provide audio contribution in the future.

Some manufacturers of audio contribution equipment have already produced codecs operating over IP. Until recently, these hardware codecs were not interoperable.

Based on an initiative from German vendors and broadcasters, the European Broadcasting Union (EBU) has created a standard for interoperability for audio contribution over IP. An EBU group, Audio Contribution over IP (N/ACIP), under the auspices of the Network Management Committee (NMC) of the EBU, had its inaugural meeting in February 2006. The group's work has had many collaborative meetings with manufacturers. The two main objectives of N/ACIP are interoperability between audio contribution devices and advice on the utilisation of audio over IP systems.

2.2 IP Based Contribution

There are two main methods by which contributions may be made from remote sites or local offices into main studio centres:

- file transfer (store and forward)
- live contributions (streamed)

These contributions may be made either over:

- Well-managed private networks
- Un-managed public network – the Internet

Whichever network type is used, file transfer contributions can normally be guaranteed to maintain good audio quality and security through the use of File Transfer Protocol (FTP) and Transmission Control Protocol (TCP). Even on unmanaged networks, any delays in the transfer will not cause degradation in the audio quality.

However, live contributions need to be streamed across the network. In well managed internal corporate networks, possibly due to other data traffic, any delays and/or packet loss (or other network impairments) can be much better controlled than over un-managed networks, such as the Internet. This is one of the major challenges in the move to packet switched networks, such as IP, from the synchronous ISDN networks. Despite this, the Internet is being increasingly used for broadcast links and will continue to do so as Next Generation Networks (NGN) with higher bandwidths and improved services become more widespread.

3 Current standards for VoIP telephony

The Internet is of great importance for the telephony industry. Voice over IP (VoIP) usually denotes telephone quality audio over the Internet. The switchover from traditional dedicated phone lines is driven mainly by the fact that consumer prices are lower for Internet telephony.

The main challenges for VoIP:

- Available bandwidth
- Network latency
- Packet loss
- Jitter
- Echo
- Security
- Reliability

These have to be solved in order to deliver a performance equal to traditional telephony lines. In addition, deployment of VoIP on a network should also not adversely affect existing traffic, whether it is normal data business traffic or, as in the broadcast industry, live media transfer.

Initially VoIP was technically controlled by the International Telecommunications Union (ITU) standard H.323. Most carriers use H.323 for their backbone. Nowadays the Internet Engineering Task Force (IETF) standard Session Initiation Protocol (SIP) is mainly used for VoIP, especially for small customers setting up their own systems in homes or smaller businesses.

SIP is a signalling protocol that handles such things as;

- REGISTRATION at the “digital phone exchange” / SIP server.
- INVITE to a phone call.
- Hang up a phone call by using BYE.
- Tell other SIP users the OPTIONS you are using.
- ACKnowledge any SIP message.

SIP sends its commands using Session Description Protocol (SDP). The SDP is sent in clear text and could look as follows:

```
v=0
o=alice 2890844526 2890844526 IN IP4 host.anywhere.com
s=
c=IN IP4 host.anywhere.com
t=0 0
m=audio 49232 RTP/AVP 98
a=rtpmap:98 L16/48000/2
```

Exchange of SDP messages are done between the two parties of a call. A SIP call is usually set up through a series of phone exchanges, much like the old analogue phone call. In SIP, these exchanges are called SIP servers. A call setup through a SIP server might look as follows:

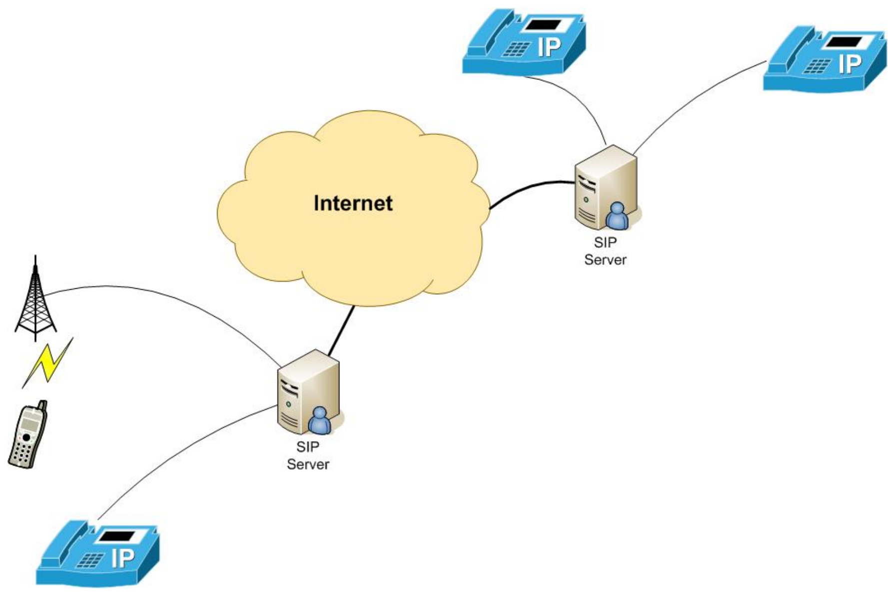

Figure 1 Possible VoIP Call Setup

4 Coding standards for VoIP

4.1 G.711

In ISDN, the Public Switched Telephone Network (PSTN) and VoIP, the de facto standard for audio coding is G.711. This exists in two flavours; µ-law and A-law. µ-law is mainly used in North America and Japan, whereas A-law is used in Europe and the remainder of the world. These standards encode 14-bit linear Pulse Code Modulation (PCM) samples to 8-bit logarithmic samples. The signal is sampled at 8 kHz which gives a bit-rate at 64 kbit/s. µ-law tends to give higher resolution at a high signal level, while A-law gives higher resolution at lower signal levels. Both these types of coding are easy for a computer to calculate.

4.2 G.722

Newer VoIP handsets sometimes include this codec. Some handset manufacturers call this feature "HD". G.722 samples the audio at 16 kHz (though the IANA[5] records, due to a historical error, wrongly specify that G.722 has a sample rate of 8 kHz[6]). In audio contribution for broadcast over ISDN, G.722 has been extensively used. This codec also exists in newer versions, G.722.1 and G.722.2.

In addition to this list, other coding algorithms are also used, such as speex, iLBC, etc.

5 Audio Contribution over IP – EBU TECH 3326

An important distinction between VoIP and Audio over IP should be understood by the reader.

VoIP is generally used for relatively short phone calls using low audio (telephony quality, ~8kHz) bandwidth, whereas Audio over IP for live contribution is generally for longer periods using higher quality codecs or linear PCM audio, intended for broadcasting to the general public.

The widespread nature of VoIP engenders – even among technical staff – the feeling that VoIP is the same as Audio over IP. While they are related, they are definitely not the same. This is due to psychoacoustic particularities rather than the underlying technology. Indeed voice is very forgiving because of the inherent redundancy of speech. Artefacts and frame losses in music contributions immediately lead to the perception of bad quality.[1]

Codecs intended for contribution over IP from differing manufacturers did not historically intercommunicate. Based on an initiative from German vendors and broadcasters, the EBU, together with manufacturers, created a standard for interoperability for audio contribution over IP. The EBU group, N/ACIP published the Requirements for Interoperability document EBU TECH 3326[2], together with a Tutorial on Audio Contribution over IP, EBU TECH 3329[2]; a guide to support EBU members in this technology.

The interoperability specification is based on protocols used for VoIP, with RTP over UDP for the transport of audio and SIP for signalling.

The standard describes a minimal set of MUSTs when implementing an Audio over IP device (AoIP). It also describes a set of RECOMMENDED and OPTIONAL additions. Portable AoIP devices do not have to include all the features needed in none-portable devices.

6 Coding standards for Audio over IP

The G.711 and G.722 coding algorithms used in VoIP are mandatory within the Requirements for Interoperability document for Audio over IP, as are the following additional algorithms.

[5] Internet Assigned Numbers Authority
[6] See RFC3551, section 4.5.2 for clarification

6.1 Linear PCM

This is not an audio coding as such, but the straightforward mapping of the analogue audio signal to data representation. In broadcasting, the sample rate is usually 44.1 kHz or 48 kHz using 16-bits (12, 20 and 24-bit also used).

6.2 MPEG-1 Audio Layer II (MP2 or Musicam)

MP2 is a psychoacoustical compression algorithm. It splits the audio into 32 sub-bands and if any sub-band is considered to be imperceptible, that sub-band is then not encoded.

The MP2 audio codec is used in some broadcast environments. For example, Swedish Radio stores the major part of their audio in this format at the highest available bit rate of 384 kbit/s. This codec is chosen in the broadcast community because as early as 1992 it already had a sound quality that was considered acceptable. This codec is also considered more error resilient than newer codecs such as MP3.

Further details of the following set of codecs that are included as recommendations or optional, can also be found in the EBU TECH 3326 document.

6.3 Recommended audio codecs

ISO MPEG-1/2 Layer III

MPEG-4 AAC, MPEG-4 AAC-LD

6.4 Optional audio codecs

Enhanced APT-X

MPEG-4 HE-AACv2

Dolby AC-3

AMR-WB+

6.5 SIP

As with VoIP, session management in AoIP is provided by the use of SIP, along with SDP (Session Announcement Protocol – SAP, must also be supported). So far the SIP standard has not clashed with the needs of the broadcasters. Some of the parts in SIP are well defined, especially how communication with SIP servers is performed. But, as the SIP community is rapidly growing, at times, ad hoc rules are in force. How the EBU standard should use these ad hoc rules are debatable, but so far the main way forward has been: “If we can communicate with standard SIP phones and SIP software phones using G.711, we’re probably safe with the worldwide SIP community”.

The screen shot below shows a typical SIP contribution link setup and teardown, describing the messages passing between the two endpoints, or User Agents (192.168.191 and 192.168.1.71) and the SIP server in the middle (192.168.1.50). Once the connection is established, the audio stream passes directly between the two User Agents.

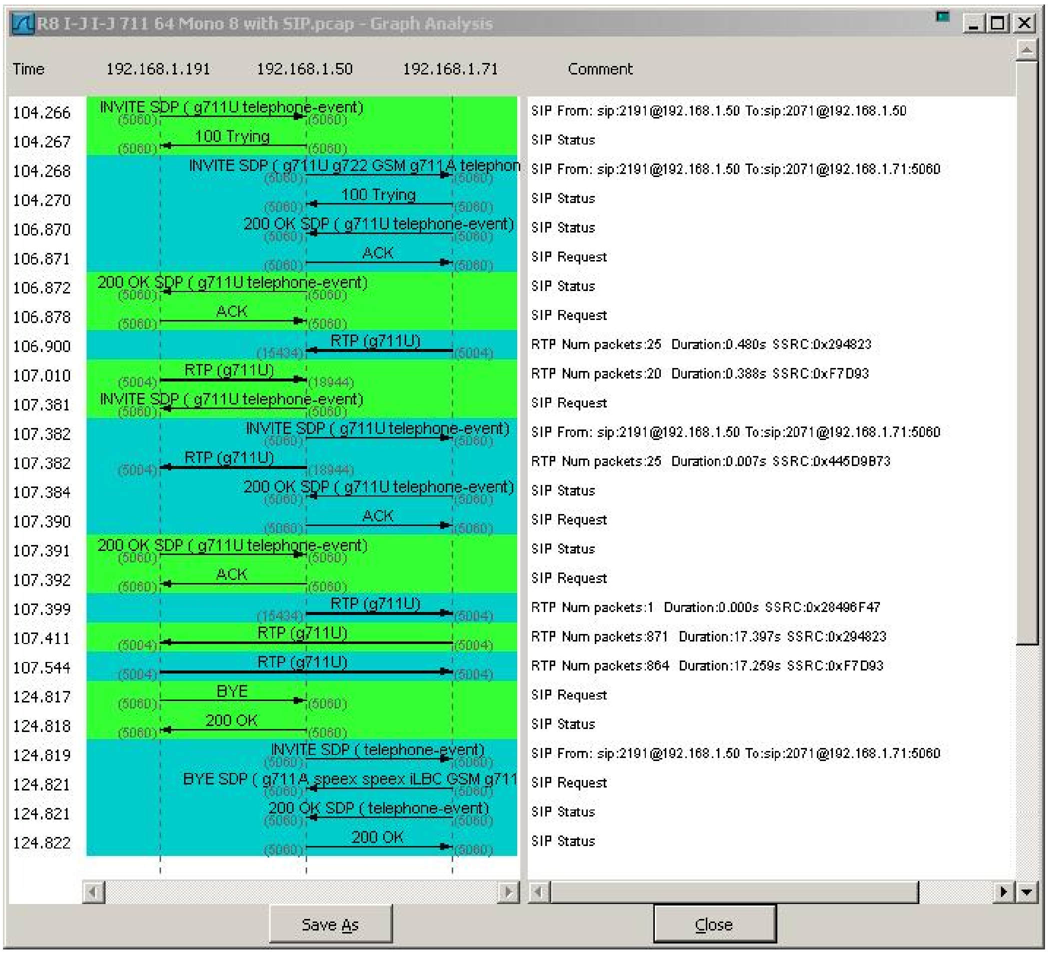

Figure 2 A typical complete SIP session

7 Verification of EBU TECH 3326

Initially this document existed only as a paper specification. The early involvement of manufacturers meant they were already starting to implement the various protocols and requirements to produce operational systems conforming to the specification. The EBU group N/ACIP decided that it would be prudent to build its own independent reference implementation of this document. This had a two fold purpose:

- to verify the document’s technical content and
- to independently be able to confirm manufacturers claims of interoperability

An important factor was to reassure manufacturers that the EBU (or those involved in the work of building the reference model) were not in the business of producing a commercial product in competition with them.

8 Success Triangle

The success triangle was the name given to the process to indicate how the reference would fit into the manufacturer interoperability testing.

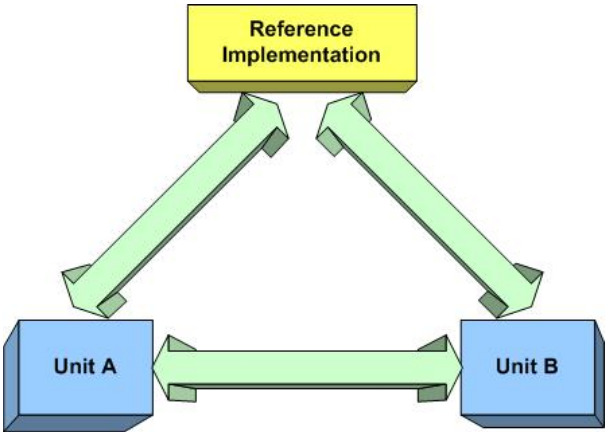

Figure 3 The Success Triangle

If both unit A and unit B can communicate successfully with a reference implementation, they then should be able to communicate with each other. The reference was required to implement all the mandatory parts of the EBU TECH3326 document only.

9 Open Source implementation of SIP

PJSIP is an open source SIP project[3], basically giving the user a fully usable SIP software phone. The package works well together with other tested VoIP SIP phones. The project is well documented and a lively mailing list exists. The software is released under dual license. Free as in open source, but if you are a manufacturer selling a product based on the source code a license must be obtained. Out of the box, after compilation, the PJSIP package has a reference application, PJSUA, which communicates well with all software and hardware SIP phones so far tested.

PJSUA is controlled via a command line interface. Using this on a portable computer with a good audio interface constitutes a free AOIP device for which, at the time of writing, is capable of at least G.711, G.722, as well as Linear PCM audio. Compatibility with MP2 is planned in the roadmap for PJSIP.

10 The EBU Audio over IP Reference Implementation (‘*Reference*’)

At conception of the project, due to the uncertainty of the exact amount of work involved, it was decided that the development of a reference would be a joint undertaking between BBC and IRT, two members of the EBU N/ACIP group.

The work was split between the two organisations to cover these aspects of the work required:

- user interface
- coding algorithms
- network
- packetization
- logging
- audio interfacing

Using PJSIP and PJSUA as a ready-made base, the following had to be done in order to create the ***Reference***;

- Graphical User Interface (GUI)
- Help in usage
- Logging functions for further studies

As PJSUA is a command line tool complete with the source code, building the GUI could be done either using the source code that was relevant or building an application at a separate layer, which in the background, communicated with the hidden console application. The latter was chosen for two main reasons. Firstly this would ensure compatibility with PJSUA in the future; any new version with the standard command line commands was more likely to work without having to re-write the GUI. Secondly, any coding algorithms added to the PJSIP package would easily integrate to the GUI without recompilation.

For the task, Microsoft's Visual C++ 2005 Express was used. Later in the project, compiling with Visual C++ 2008 Express was tested without any problems. The source code for this project is available at Sourceforge[4].

A screen shot of the resultant User Interface is shown below. Note that this may differ from the version that is downloaded from Sourceforge.

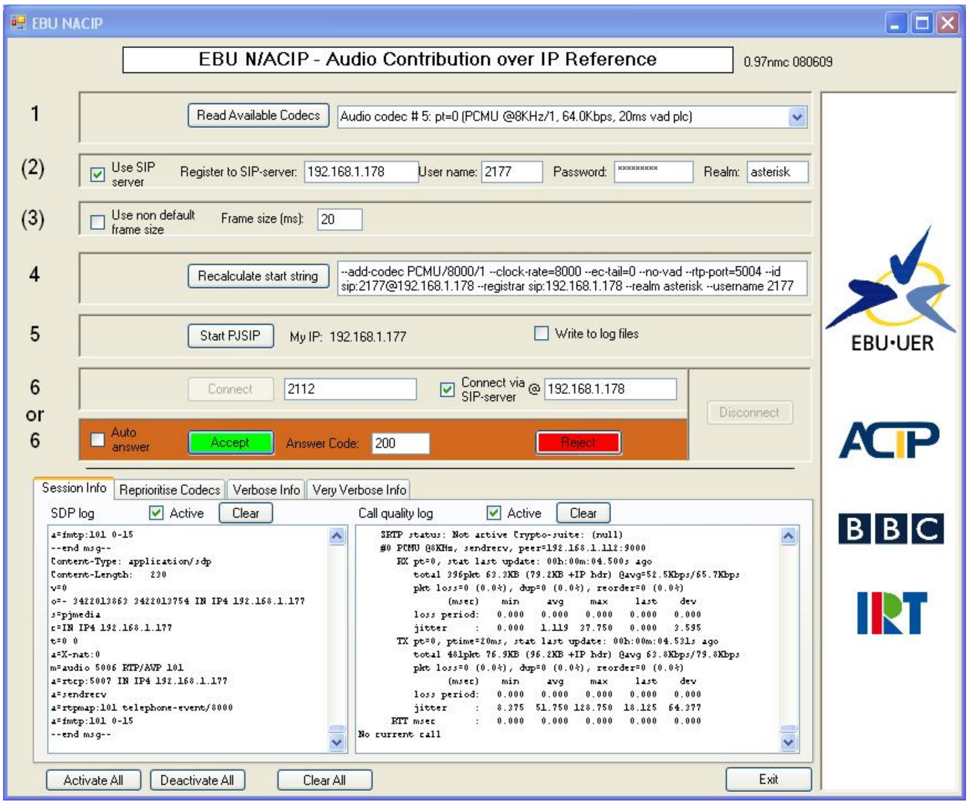

Figure 4 Screen shot of Reference Implementation

At runtime the ***Reference*** performs the following steps;

1. Checks which coding algorithms are available in the available (precompiled) PJSUA application.
2. The start-up parameters for PJSUA are selected using the GUI. The suggested start-up parameters are available in an edit box for further enhancements by the user. Among the start-up parameters, the registration with a SIP server is available.
3. Start-up of PJSUA.

4. Decide if logging is required. This option outputs logs into four files; SDP output, live connection quality feedback, verbose output and very verbose output.
5. A contribution link can now be started using one of the following procedures;
   - entering the other parties IP address (e.g. 123.123.123.1)
   - entering the other parties “phone number” or name, as previously set-up in ***your*** organisations SIP server (e.g.g matti@internalsip.bbc.co.uk)
   - calling, using the ***receivers*** SIP server (folkradionP3@externalsip.sr.se)
   - answering or auto-answering an incoming request. Answering can also be done sending various SIP response codes. The response code defaults to “200 OK”.
6. Once a successful link is set-up, audio, if present, is transmitted and received. The connection quality field in the ***Reference*** shows a continuous update of current connection information.
7. The disconnect button becomes active and the termination of a link can be performed.
8. At any time, any other PJSUA command may be issued.

The Appendices contains a note on Programming a GUI for a background console application.

11 Manufacturer Interoperability Plug Test

The Plug Test was an event held in Munich by the EBU and hosted at IRT. Attending this Plug Test were nine manufacturers with their equipment. The goal was to organise plug sessions where the manufacturers could test interoperability in a live environment. The ***Reference*** also took part in these tests. Additionally, a SIP server (based on Asterisk®[7])was available for testing.

The test rounds were organised into 5 groups (tables), as shown in Figure 5, each group containing two manufacturers and one EBU member, in the role of an independent adjudicator.

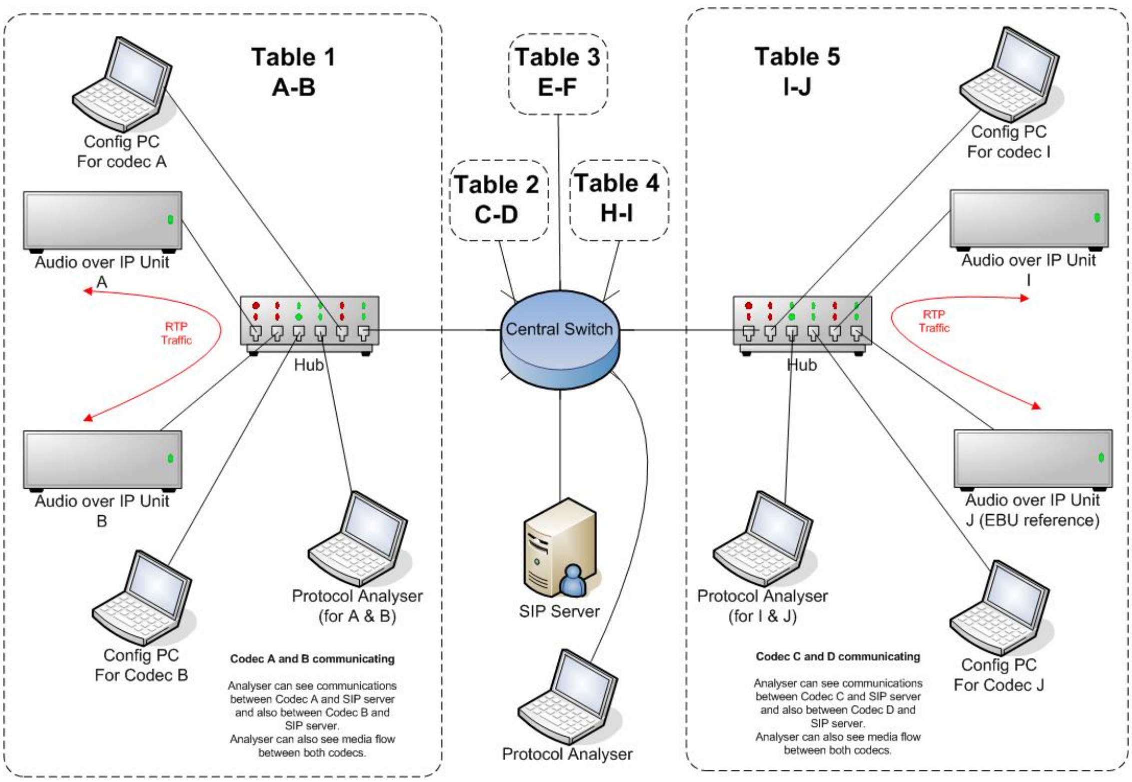

Figure 5 Analysis Tables and Network Structure

Each round consisted of tests to ensure;

- Coding interoperability. Coding algorithms tested were G.711 µ-law, G.722, MP2 and Linear PCM
- SIP connection set-up and destruction
- Call setup using a SIP server

All network communications were recorded and analysed using Wireshark©[5] running on portable computers through Ethernet HUB's. Wireshark© was used as an analysis tool due to being freely available and also having some very useful functions built in for visualising SIP connections. The use of network hubs ensured that the analysers (containing only single Network Interface Cards) on each table could see all the network traffic in both directions, including the communications to and from the SIP server. Each table was connected to a central switch. The SIP server was connected to this to act as a central service for all tables. The central analyser was able to monitor all flows from all tables to and from the SIP server.

Many of the manufacturer's equipment did interoperate with each other and the ***Reference***, using G.711, SIP and the SIP server, although not all could fully operate with the SIP server.[7]

12 Future ideas of usage

The main purpose of the EBU TECH 3326 standard is delivery of audio to replace contribution links over ISDN now in use. As stated previously, this replacement is forced by the closure of ISDN in some parts of the world. In addition, the cost of installing and using ISDN is high compared to using an already available Internet connection for contribution.

In choosing the SIP standard for the EBU TECH 3326, many ideas available of how to enhance communication in the world of VoIP are also immediately available for AoIP. For example, SecondLife communities are looking at how to utilise SIP stacks instead of proprietary voice communication[6]. Below are a few examples of how AoIP **might** be used in the future, if not already used in this way.[8]

- **Comms for TV (Both Outside Broadcast and between Gallery and studio)**

  Imagine a small AoIP unit that can be used for comms (4-wire) between gallery and all connected units available inside the BBC network. An easier setup might be to have a SIP server with conference call functionality, such that calls between different groups of users exist as virtual conference rooms within the SIP server. These pre-configured conference rooms are day-by-day assigned new uses. Conference room #11, for example, could be the point that the external OB camera operators connect to. If the gallery wants to contact the operator, the gallery contacts conference room #11. The CAR (Central Apparatus Room) can also connect to this conference call when needed.

  Broadcast, i.e. transmit only, network communication could be from another virtual conference room, such as network countdowns before regional opt-outs, for example.

- **Audio contribution in studios receiving multiple incoming connections**

  All incoming and outgoing signals in a studio could be done via software in a computer equipped with AES sound cards. The incoming and outgoing connections could be to AoIP interfaces in other studios using high quality codecs (Linear or MP2 384 kbit/s), or from listeners who are using a soft phone from the radio shows web page, using a better than

[7] Due to time limitations, only G.711 was formally tested using the SIP server. Tests using the other mandatory algorithms, G.722, MPEG LII and PCM linear were conducted as point-to-point connections, without the SIP server. At the time of the tests, the ***Reference*** did not support G.722 or MPEG LII.

[8] Note that all these are possible ideas only. They do not represent the BBC's confirmed current or future policy in this field.

telephony audio codec (G.722), or maybe an analogue phone-in using a gateway such as a SIP server. This computer would be a studios one-machine-for-all-inputs-and-outputs.

- **Multi channel audio contribution**

  Today there is no standardised way of contribution of multiple sample accurate synchronised audio streams. This could be used in surround sound, sports broadcasts with two commentators as well as the stadium audience in stereo, with balancing back at studio, or for multi-channel contribution, e.g. a few sub mixes from a live event for live on air balancing being done down the line at a contribution studio. The latter example could save some cost at live music events. The sound insulated OB van might not be needed for all of these jobs. This could also lead to better control of Sunday church events, which for some broadcasters, is done today without a sound isolated balancing room (OB van).

- **Journalist in field, one software fits all**

  A journalist in the field, far away from home could have one portable computer that could handle all needs in live broadcast, as long as some form of Internet is available. This all-in-one application would include a small browser window for quick surfing, a chat window with the producer back home while on air, sound levels for reporter microphones, return headphone mixer, and also some form of playback device for previously recorded clips. This software maybe also could include a simple sound editor.

- **Securing stream by sending same stream over multiple paths**

  A reporter with a mobile unit at a sport event could use multiple 3G data links back to studio. These links should be with separate operators. The link with safest signal at the time is chosen (diversity).

- **Securing stream by selecting bandwidth dependant on available bandwidth**

  The data rate of the connection could automatically be selected depending on the available bandwidth. If the packet loss is high, the system could automatically reconnect using lower bandwidth and when the loss drops the system could automatically increase the bandwidth. Much the same way as is done today in streaming media applications to end users.

- **Implementing AoIP devices in a central apparatus room**

  In a large broadcast environment, all the AoIP devices might be placed together in a central apparatus room. This would make it possible for backup devices on standby. A matrix could further route the audio output from these devices, controlled by the SIP INVITE command. This is how the Asterisk® server router for analogue telephony works. For example, +46705137596@sip.telephonecompany.ac.uk would connect to the SIP server at sip.telephonecompany.ac.uk and from there connect the call further onto an analogue phone circuit, dialling the phone number preceding the '@' symbol. For our use, the audio router would manage the connection to the correct studio console input, e.g. studio32mixin12@internal.bbc.co.uk. (Thanks to Fredrik Bergholtz at Swedish Radio for suggesting this.)

- **Using SIP servers to get through Firewalls at broadcasters**

  Opening many separate holes in the broadcasters firewall is usually not recommended, and sometimes the preconfigured routes through a firewall are not always applicable. A journalist out in the field may not know the IP address until close to transmission or if the journalist is behind a Network Address Translation (NAT) the IP of the Wide Area Network (WAN) might never be known. If there existed one SIP server outside the firewall filtering the calls to a SIP server inside the firewall only the connection between the SIP servers must be safeguarded. Note that these SIP servers would then not only set up the calls, but they would also transfer (but not transcode) the media itself.

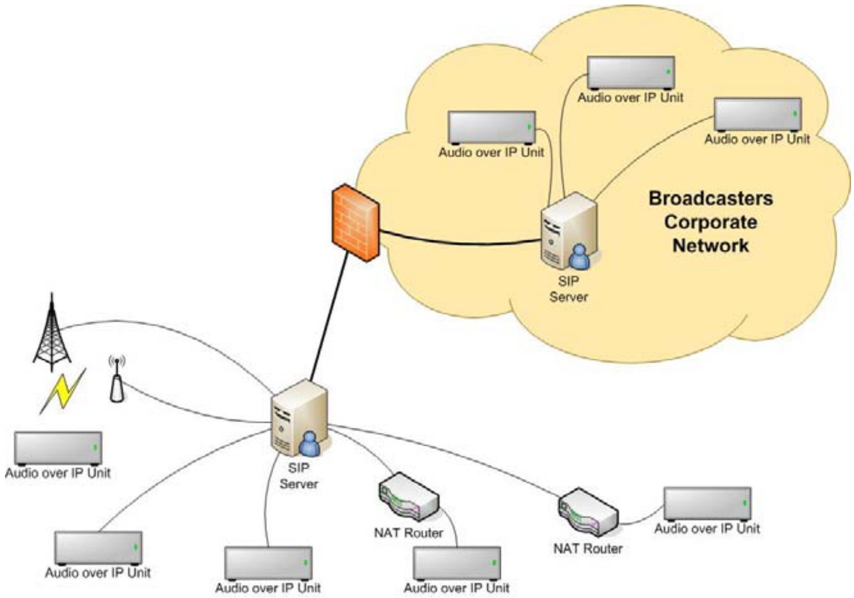

Figure 6 Possible SIP Server arrangement to traverse Firewalls

13 References

[1] N/ACIP-Reflector, October 2007. Report on 10th Meeting, Amsterdam September 2008. Tesch, P. 2007, Dissertation title "Switching to the packet - an approach for changing to audio transport over IP".

[2] www.ebu-acip.org

[3] www.pjsip.org

[4] https://sourceforge.net/projects/aoip

[5] http://www.wireshark.org

[6] http://sourceforge.net/projects/voipforvw/

[7] http://www.asterisk.org/

[8] http://www.freeswitch.org/

[9] http://www.iptel.org/ser/

[10] http://www.mysipswitch.com/

[11] http://www.sipfoundry.org/sipX/

[12] http://www.brekeke.com/

[13] http://openser.org/

14 Appendices

14.1 Appendix 1 - Note on SIP servers

The SIP server mainly used in this development project has been the Asterisk® server. Asterisk® is an open source SIP server that also could act as a SIP device, e.g. answer phone functions. Asterisk® is widely used in the SIP community as a gateway from analogue phone lines to SIP. Using Asterisk® as your switchboard a fully programmable call switcher becomes accessible. Asterisk® is also capable of transcoding calls. This is used when an incoming audio stream and outgoing audio stream use different coding algorithms. Asterisk® is out of the box capable of G.711 and G.722. As the software is open source, changes can be made to incorporate other codecs into the Asterisk® environment. Within the AoIP space the server should not perform any transcoding; this can be set in Asterisk®'s config file.

Different SIP servers act in slightly different ways. As an example, If Asterisk® (A) is to set up a call from device (B) to device (C) the first few milliseconds of the call go through Asterisk® (A). Other SIP servers don't exhibit this behaviour. After this short period, Asterisk® reroutes the call so that the packets of audio flow directly from device (B) to device (C). This is done by Asterisk® using the SIP command REINVITE. Not all manufacturers at the Munich Plug Test had implemented this command, as it not is specified in the EBU TECH 3326 document. As a side note, at least one of the manufacturers' devices could handle this without any problem if Asterisk® instead used the UPDATE command for this part of the call. This is also settable in the Asterisk® config file. Other SIP servers handle this type of call setup in a different way.

Many differing SIP servers exist such as these, some free;

- Asterisk®[7]
- Freeswitch[8]
- SIP Express Router (SER)[9]
- Mysipswitch[10]
- sipX[11]
- Brekeke[12]
- OpenSIPS (formally OpenSER)[13]

14.2 Appendix 2 - Programming a GUI for a background console application

The central part of the ***Reference*** is how to set up the control of the console application. This is done using the '`Process`' command in Microsoft Visual Studio. As output from the console is received, this new text is placed in one of two buffers.

```
private: System::Void StartPJSIPButton_Click(System::Object^  sender,
System::EventArgs^  e) {
       using namespace System;
       using namespace System::Diagnostics;
       using namespace System::ComponentModel;
       using namespace System::IO;
       using namespace System::Text;
       using namespace System::Threading;

       ProcessStartInfo^ myProcessStartInfo = gcnew
ProcessStartInfo("C:pjsua_vc6d.exe");
       myProcessStartInfo->UseShellExecute = false;
       myProcessStartInfo->RedirectStandardOutput = true;
```

```
        myProcessStartInfo->RedirectStandardInput = true;
        myProcessStartInfo->CreateNoWindow = true;
        String^ startupArguments = SipStartupStringTextBox->Text;
        myProcessStartInfo->Arguments = SipStartupStringTextBox->Text;
        myProcess->OutputDataReceived += gcnew DataReceivedEventHandler(
OutputHandler );
        myProcess->StartInfo = myProcessStartInfo;
        myProcess->Start();
        myProcess->BeginOutputReadLine();
        myProcessRunning = true;
        }

private: System::Void quitPjsipButton_Click(System::Object^  sender,
System::EventArgs^  e) {
        myProcess->StandardInput->WriteLine("q"); // Send “q” to Pjsua
        myProcess->CancelOutputRead();
        myProcess->Kill();
        myProcessRunning = false;
        }
```

Controlled with a timer, this buffer is parsed and processed. As one buffer is parsed, new output from PJSUA is added to the second buffer, which subsequently is parsed in the same way as the first buffer. In testing, the second buffer is never used, but this part has been left in the code for slow processors or other unknown processor interruptions. The parsed output is used both for various logs and for status control in the ***Reference***, such as incoming calls. In the future, the parser could easily also add support for Instant Messaging (IM) as this is defined within the SIP protocol.

Other parts of the ***Reference*** allow the possibility of preference changes, such as Auto Answer and setting frame sizes for the outgoing packets.