Control Protocol Translation for Device Interoperability in the Context of Ethernet AVB

Osedum P. Igumbor, Richard Foss
Department of Computer Science, Rhodes University, Grahamstown 6140, South Africa
Tel: +27 46 603 8291, Fax: +27 46 636 1915
Email: g06i3351@campus.ru.ac.za; r.foss@ru.ac.za

***Abstract*—With the introduction of Ethernet Audio Video Bridging (AVB) technology, there is an opportunity for deterministic transmission of data over Ethernet. The guaranteed transmission of data using the AVB technology will enable streaming of media (audio/video) alongside non-media data on the same Ethernet network. Each pro-audio device within a digital audio network, be it Ethernet or some other networking technology, conforms to a single control protocol. Such devices can only communicate and interoperate with other devices that use the same protocol. This paper describes an application that will enable interoperability amongst networked audio devices with different control protocols, over an Ethernet AVB network. The design of the proxy application described here will reveal how interoperability can be achieved between AES-X170 and OSC devices on an Ethernet AVB network.** [1]

***Index Terms*—Ethernet, AVB, AES-X170, OSC, Proxy**

I. Introduction

ETHERNET is an IEEE 802.3 standard that provides for the transmission of data over a Local Area Network (LAN). A LAN is a packet-switched network that spans a relatively small geographic area, for example a single building, an academic institution, or a residential complex. Ethernet has been used in various contexts including VoIP and IP-TV. Within the context of digital audio networks, there is the need to transport audio and control instructions on the same network. However, existing Ethernet LAN technologies do not enable the following requirements that are needed for such real-time data transmission:

- device synchronization
- timely transmission
- deterministic transmission

As a result the Audio/Video Bridging (AVB) Task Group was tasked with addressing these issues [1].

A. Ethernet AVB - The promise

The Audio/Video Bridging Task Group has introduced some specifications and amendments to the IEEE 802.1 standards so as to ensure low-latency transfer of data (in particular audio/video) between time-synchronized devices on an Ethernet network. Issues such as network resource management and device synchronization are handled by the standards that make up the AVB suite. Worthy of mention amongst these are [1]:

- IEEE 802.1AS - ensures synchronization of devices on the network by defining an algorithm that is used to determine the clock master amongst networked devices.
- IEEE 802.1Qav - defines a mechanism necessary for the queuing and forwarding of audio/video across a network.
- IEEE 802.1Qat - defines the Stream Reservation Protocol (SRP) that enables an AVB *listener* to request for network resources such as bandwidth to be reserved.

B. Digital Audio Network Control Protocols

There exist a large number of audio control protocols, many of them being IP-based protocols. These include ACN [2], HiQnet [3], IEC 62379 [4], OSC [5] and AES-X170 [6]. Each protocol was designed to address specific user contexts and use-case scenarios.

Devices within a network can only communicate with other devices that implement the same protocol. This isolates devices based on the control protocol they implement.

C. Proxy for Device Interoperability

The use of a proxy application for interoperability among pro-audio devices within digital audio networks has been demonstrated with two audio control protocols, namely; Audio/Video Control (AV/C) and AES-X170 [7]. In that investigation, AES-X170 was used as the preferred protocol to implement a common controller for both AV/C and AES-X170 devices above firewire. The proxy fulfilled the role of a translator, which is capable of receiving AES-X170 messages on behalf of the AV/C devices, and in-turn sends the appropriate AV/C commands.

In this investigation, a similar proxy will be implemented for interoperability between disparate protocol devices on an AVB network. As a first step the proxy will allow for OSC and AES-X170 interoperability. A conceptual view is shown in Figure 1.

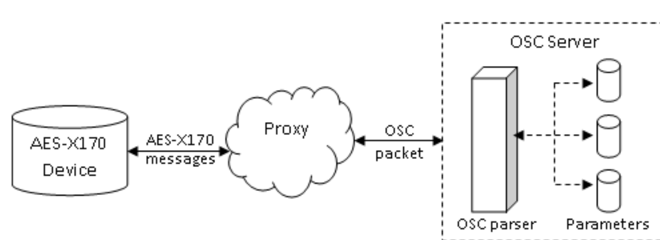

Figure 1. Proxy application translates OSC and AES-X170 instructions

[1]The work reported in this paper was undertaken in the Telkom Centre of Excellence in Distributed Multimedia at Rhodes University, with financial support from Telkom, Comverse, Tellabs, Stortech, Amatole Telecom Services, Bright Ideas 39 and THRIP. The financial assistance from the Rhodes University Prestigious Scholarship towards this research is hereby acknowledged.

The proxy should be able to:

- discover OSC devices on a network using zero-configuration networking mechanisms [8],
- model parameters within an *OSC server* according to the AES-X170's 7-level parameter hierarchy,
- translate AES-X170 messages to OSC messages, and vice-versa

II. Open Sound Control (OSC) Protocol

Open Sound Control (OSC) protocol is a digital media content format for streams of real-time audio control messages [9]. OSC messaging involves a device that requests for a service (*OSC client*), and a device that renders a service (*OSC server*). An OSC server can be conceptually viewed as a container that models parameters in a structured 'URL/directory-style' hierarchy of nodes. Typically, the root node in the hierarchy is an *OSC container* that contains other OSC containers and *OSC methods*. A leaf node will typically be an *OSC method* that is capable of performing a particular action.

An *OSC address space* forms a hierarchy from a root container to all of its sub-containers and methods. The 'URL-style' address (*OSC address*) is used to describe a specific OSC method. Any number of arguments can be passed to an OSC method.

An *OSC packet* is either an:

- *OSC message* - consists of <*OSC address pattern*>, <*OSC type tag string*> and <*OSC arguments*>, or
- *OSC bundle* - consists of <*"#bundle"*>, <*OSC time tag*>, and <*OSC bundle elements*>.

The terms used in the above description of an OSC message and OSC bundle are defined in the OSC specification [5].

III. AES-X170

AES-X170 is an IP-based peer-to-peer protocol that enables device monitoring, configuration and control over a network [6]. Each AES-X170 device models its parameters within an application *node*. A consistent (7-level) hierarchy is used to describe each parameter within a node. To address a parameter from a remote controller/connection manager, an AES-X170 message indicates the <*device/host IP address*>, <*application node ID*> and <*7-level parameter address*>.

AES-X170 enables *joining* of parameters into groups such that adjusting one parameter in the group will cause a corresponding modification of all other parameters in the *join-group*.

IV. OSC Proxy Design

This section highlights the various aspects that together will enable an application (OSC proxy) to discover OSC devices on a network, and translate AES-X170 messages to OSC messages.

The nature of the proxy application is shown in Figure 2.

Below is a summary of each of the library APIs used.

- *AES-X170* - an API that allows for software interaction with the AES-X170 protocol stack, which is necessary for AES-X170 messaging.
- *WOscLib* - a C++ API that implements OSC. It enables the creation of the various OSC containers and OSC methods, and enables OSC messaging.

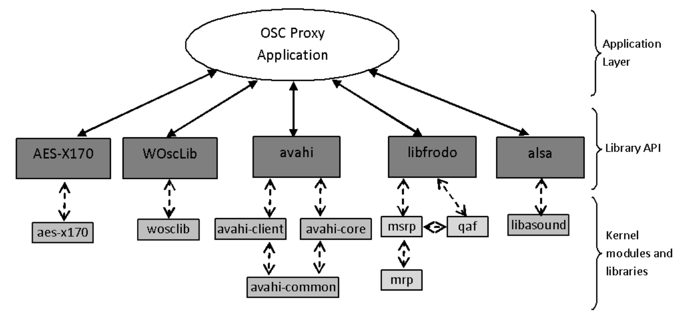

Figure 2. Proxy application interacts with various libraries

- *avahi* - an implementation of zero configuration networking licensed under LGPL.
- *libfrodo* - an API that implements *MRP*, *MSRP*, and queuing and forwarding in accordance with AVB.
- *alsa* - allows for audio and MIDI control on a Linux operating system.

The proxy application discovers OSC servers on the network using the *avahi* library. For each discovered OSC server the application models the device's parameters within an AES-X170 node using the *AES-X170* library. All OSC messaging between the proxy and an OSC server utilizes the *WOscLib*. The *alsa* API enables the application to capture and playback audio using its analog plugs. The application streams audio over an Ethernet AVB network using the *libfrodo* library.

V. Conclusion

This paper described the motivation for and creation of an application (proxy) that enables communication between devices on an Ethernet AVB network with different control protocols. The proxy is capable of enabling such communication between devices that implement different audio control protocols by translating instructions, thus enabling device interoperability and common control.

References

[1] Institute of Electrical and Electronics Engineers (IEEE). *Audio/Video Bridging Task Group*. IEEE, September 2008. http://www.ieee802.org/1/pages/avbridges.html [Accessed: 2010.06.01].

[2] ESTA. Entertainment technology - architecture for control networks. Entertainment Services and Technology Association - ESTA, 2005.

[3] Phillips J. Hiqnet overview. Sound Marketing Central, 2005.

[4] IEC. *IEC 62379-1: Common Control Interface - Part 1: General*. International Electrotechnical Commission - IEC, 1st edition, 2007.

[5] M Wright. *The Open Sound Control 1.0 Specification*, 1.0 edition, March 2002. http://opensoundcontrol.org/spec-1_0 [Accessed: 2010.06.01].

[6] Audio Engineering Society. *AES-X170: Integrated Control, Monitoring and Connection Management for Digital Audio Networks*, draft edition, 2009.

[7] O.P. Igumbor and R. Foss. *A Proxy Approach for Interoperability and Common Control of Networked Digital Audio Devices*. Presented at the 128th Audio Engineering Society Convention, London, 2010.

[8] S. Cheshire and D.H. Steinberg. *Zero Configuration Networkng - The Definitive Guide*. O'Rielly, 2005. ISBN: 0-596-10100-7.

[9] A. Schmeder, A. Freed, and D. Wessel. *Best Practices for Open Sound Control*, May 2010. http://opensoundcontrol.org/files/osc-best-practices-final.pdf [Accessed: 2010.06.01].

OSEDUM P. IGUMBOR is a PhD candidate and member of the Audio Research Group in the department of Computer Science at Rhodes University.