

Vincent J. Manzo
DAP 1
Final Report
December 3, 2005

Musical Protocols: Open Sound Control and MIDI

In all of the years before the 1980's, recording musicians who used synthesizers would have no way of manipulating digital variables of that data because, simply, it was an analog signal.

With the birth of the personal computer and a world-wide digital revolution, advanced computer integration techniques caused corporations to rely on their equipment and devices and, more importantly, the communication of those devices with other devices.

Since the 1980's, musicians have relied on MIDI (Musical Instrument Digital interface) as the language that musical instruments use to speak to each other. This type of computing language is known as a protocol. While this protocol is good at what it does, it is not without its limitations.

Such limitations are a product of the protocols design environment; its manufacturers' inability to foresee complications and expandability needs years down the line. While several advancements have been made, a new protocol with superior functionality has emerged.

This protocol is called Open Sound Control (OSC). Open Sound Control is optimized for superior functionality when compared to MIDI. Its strongest advantage is that it takes full grasp of thriving network protocol concepts and designs such as LAN and firewire.

Original MIDI designs were created to connect one keyboard to another (respectively) and transmit up to 16 channels of data through a single cable. Companies, in particular hardware developers, manufactured sound cards utilizing MIDI.

With the rapid growth of the internet, network card manufacturers and network analysts have improved upon early designs of networks and developed LAN's (local area networks). LAN's can be connected over wide distances to other LAN's. This is called a WAN (wide area network). Network cards have an external bus standard that transmits data at 400 megabits per second. This bus is called an IEEE-1394.

In addition to being very fast, IEEE-1394 transmits data at a synchronized rate and can guarantee that the data will get there when it gets there. This provides an enormous advantage over systems using the MIDI protocol which has been known to freeze up and have "sticky" notes which require pressing "panic" button to reset variables. MIDI cables also need repeaters over longer networks. The ability to transfer large data in real-time is vital and a single IEEE-1394 port can be used to connect up to 63 additional devices.

Local area networks can connect through a number of ways, one of them being wireless connectivity with the use of radio frequencies. By incorporating these networking concepts, Open Sound Control is able to broadcast with similar functionality of standard computer networks. This yields itself open to the ability of upgrading its functionality as network interface enhancements push back the limitations of modern network infrastructures.

Since Ethernet-based local area networks are already an established standard, Open Sound Control will be able to utilize these enhancements. MIDI was unable to utilize such enhancements since its nature was solely grounded in the realm of musical instrument protocol and had little to do with major network protocols such as the internet as it existed at that time.

MIDI History

The history of MIDI is a remarkable tale. As musical instruments began to go digital, the leading manufacturers, Roland, Oberheim, Fender Rhodes, and Sequential Circuits, developed interfaces that allowed their keyboard equipment to communicate with other units. The only glitch was that these companies chose to keep their protocol proprietary and therefore, the devices could not communicate with devices from a different brand.

Dave Smith and Chet wood devised a Universal Synthesizer Interface and proposed it at AES in 1981. This became the starting point for a standardization of MIDI: a musical instrument protocol.

From 1981 to 1983, the leading device manufacturers collaborated to establish this standard. By this time, some MIDI devices were already on the market, the first of which was Sequential Circuits' Prophet 600 which shipped in 1982. Regardless, these companies continued to hash out what was soon to be the MIDI protocol. They discussed the hardware specifications and the data specifications.

Advancements soon took shape. One such advancement was the implementation of Patch banks and General MIDI. Users would create a MIDI file which is really just a

string of numbers and messages that is transmitted in the protocol. That MIDI file would then play the same note data on any synthesizer since it is only data that is being routed not actual analog music.

However, what if you were using sound number 005 on your Oberheim and when you routed your MIDI playback through to your Roland you heard a totally different sound? This is because sounds are organized differently on different synthesizers. This problem was soon addressed and sounds were organized into banks. Subsequently, a concept known as General MIDI set standards for the first 128 instrument names and sounds.

Other advancements utilized MIDI's 16 channels of data that can be transmitted. A General MIDI requirement implemented multi-timbral playback meaning that MIDI data could be assigned to any MIDI channel (1-16) and could be played back simultaneously.

Further developments included file format specifications known as SMF which described how MIDI data should be saved to a file, , MIDI time code, advanced polyphony (the Prophet 5 could only play 5 notes simultaneously) MIDI sample dump standard, numerous file extensions for companies like Korg and Roland, and eventually General MIDI 2.

As these companies soon realized, MIDI was a limited system. No major advances were made on the part of most of these manufactures because any attempt to devise a new protocol might not be accepted by the other leading manufactures. This would cause a drop in stock and possibly ruin the company. So as the major companies

continued playing a game of chicken, samplers were making the sounds of MIDI sounds seem archaic and funny.

Samplers play clips of prerecorded sounds from a buffered source called wavetable. Since the sounds are of genuine instruments, they can be very realistic depending upon the quality of the recording. These devices also implemented the MIDI protocol.

Chowning's FM synthesis and the Princeton/Yamaha device the DX7 was selling extremely well with its marketing slogan "any sound that you can imagine". This device used frequency modulation to create new sounds. The DX7 also implemented the MIDI protocol.

All of these remarkable devices were all still tied into a system whose protocol was limited by today's standards. Data transmission speeds were variables depending upon numerous factors including cable length and system latency. Connectivity was limited to 16 unique channels.

The overall design of MIDI is fashioned so that MIDI out devices cannot see what they are sending MIDI data to. This is not truly networking since there is only a single sided conversation between the instruments. Advanced functionality would mean that synthesizers can truly communicate with each other and do so reliably and quickly over large networks.

Sometime later, a new protocol was suggested by a group called the FUDI group. The idea was really networking protocol and was not solely intended for musical instruments. The protocol suggested, however, the implementation of numerous devices each with its own IP address.

Unfortunately, this system lacked the real follow-through wherewith to launch this great idea and it was not until the mid-nineties when Matthew Wright, Adrian Freed of Center for New Music and Audio Technologies, U.C. Berkeley suggested a new protocol utilizing the same idea as FUDI. This concept was first proposed in 1996 under the name “Synth Control” and after a year of development. This protocol is now known as Open Sound Control.

Open Sound Control

The intention of Open-Sound Control by its creators was to bring the music protocol up to date with surrounding technological advances. This meant incorporating concepts that were already in place such as networking technology and open ended URL-based concepts. It was not designed as an enhancement in MIDI technology, but a new protocol to replace MIDI altogether.

The creators believed that a better integration of computers, controllers and sound synthesizers will ultimately yield lower costs, provide increased reliability, create a more user-friendly feel and provide a more reactive control of the music.

The prevailing technology uses a computer (motherboard or PCI), some type of software synthesis (an operating system) or a LAN (USB, Firewire, Ethernet, etc). Open Sound Control was not designed for use only with LAN or Firewire. This is called a transport-independent protocol. Instead, it is able to be used on a wide range of connections, though Fast Ethernet and IEEE-1394 (Firewire) seem to be the most common.

The data packing is much more advanced than MIDI. Since there is more bandwidth to work with, there is not a concern for just transmitting numbers. On a standard LAN connection, Open Sound Control can transmit data at 10 megabits per second whereas MIDI could only transmit data at 31.25 kilobytes per second. Data is encoding in 32-bit or 64-bit quantities and is sent in packets. Packet-based delivery ensures synchronicity because it is time-tagged, a process which also removes jitter caused in data transfer.

Time-tags are represented in 64 bits. The first 32 bits represent the number of seconds while the latter 32 bits represent the fractional parts of a second precisely to about 200 picoseconds.

Like any network, a device is tagged with an address which allows other devices on the network to access it. Such is the case for Open Sound Control. Multiple devices can be attached to an open control network. The protocol is created to be able to send packets to and from all devices. A MIDI device cannot truly see where its data is being sent. Open Sound Control devices can send data back to the device that it is receiving data from.

With this, Open Sound Control maintains a client/server relationship in which nearly any device or program written properly can function as either the client or the server.

All Open Sound Control data is aligned on 4-byte boundaries. The basic unit of Open Sound Control data is called a message and it consists of a symbolic address and any amount of binary data up until the end of the message. Each packet contains

messages either singly or in a group known as a bundle. Bundles can contain other bundles.

Each device in an Open Sound Control network has its own unique hierarchy. The Open Sound Control protocol displays this path information in a URL-based display style.

It is proposed that in the future, using this URL-based display will result in wireless studios, provided by wireless LAN connectivity, in which a user can utilize an array of devices including an entire room full of synthesizers saving time and studio space.

Information requests can also be sent as messages in Open Sound Control. The time tag provides the time that a device sends a request to another device and the time that the second device replies. Multiple requests and replies can be submitted simultaneously while retaining the URL-based identification concerning who sent and who replied.

Other information requests can include documentation such as manuals and help files. Information is also displayed as URL-based addressing in that typing a directory with a “\” will allow a user to see all directories in the subgroup without actually changing them.

Applications

Much of the strength of Open Sound Control lies in the protocol itself. However, its true strength and its future as an effective protocol will be determined by

the transport layer. That is, the way that the protocol is implemented into the existing market and its software and hardware uses.

Open Sound Control's code is hosted on their website:

<http://www.cnmat.berkeley.edu/OpenSoundControl/>

The code is free to download and use, and any major programmers and software/hardware developers have already started implementing Open Sound Control in their applications.

Open Sound Control is useful in with sensor based electronic musical instruments. This is one of the most popular uses to date. In this application, sensors are used as the clients to transmit OSC data to the server which, in most applications, is a network computer.

A user would thus be able to set musical parameters such as pitch, duration, and more inside of the right software and use sensors to trigger these events. Sensors can include nearly anything that will send OSC data including motion sensors, pressure sensors, cameras and more. The sensors will measure the information sent and map them appropriately to the software application.

This application is not solely limited to musical performances. In fact, more-so than MIDI, the strength of the software programmed using the Open Sound Control protocol will result in the outcome, not the user's ability to control, perform or compose music

In MIDI systems, a user would be able to notate a passage out and have a MIDI sequencer play it back at any tempo desired (since tempo like pitch and all other

variables in music are numerical parameters in a MIDI controlled music system). No real musical skill is required in order to generate sound.

In an Open Sound Control system, a program can be written that sends out musical data controlled by sensors. This musical data, can do everything that MIDI can do, but it can do it faster, more reliably and with improved functionality and adaptation by unconventional controllers that are fully programmable and customizable to the users needs.

With proper programming skill, sensor based data can control microtonality as well as all different types of synthesis including additive synthesis, subtractive synthesis, granular and more.

Cycling 74's Max/MSP was the first company to implement Open Sound Control by writing a group of proprietary "externals". By using the appropriate objects, Max/MSP programming can implement Open Sound Control data in limitless ways by using OSC sensors.

Other object-based software titles have implemented Open Sound Control including Supercollider and Native Instruments' Reaktor to name a few. It should be noted, however, that many popular software titles that market "Support for Open Sound Control" are really only converting Open Sound Control data to MIDI operations instead of mapping Open Sound Control data to Open Sound Control objects like Max/MSP has done.

Open Sound Control data can be mapped to more than pitches and musical parameters. It has been used to track motion in order to graphically notate

choreography for use in traditional dance. It can be used to move graphical objects in a given software program.

Since its connectivity is often over a local area network, Open Sound Control users can participate in performance events from across the globe in real-time. Some limitations are applicable, but as network technology advances, these limitations will dissipate more and more. This inability to upgrade was an enormous downfall for MIDI systems.

Support for wide area networks have also been implemented successfully. Quintet.net is a software title for such an application. The application, which is free and was written in Max/MSP, contains a server, client, conductor, listener and viewer component all with the intention of performing music via the internet in real-time applications.

Because Open Sound Control does not have the bandwidth limitations of a MIDI system, huge packets of data can be sent over a network at today's growing internet speeds.

Picker is a program designed to take RGB and X Y data from images, live video or another video source and map them with Open Sound Control objects to generate musical data. Picker is free and is downloadable as an object for object-based programs like Max/MSP and Supercollider. The program is fully customizable and can be used in a variety of applications.

Web-interfaced applications will benefit from Open Sound Control's synthesis objects. Macromedia has already implemented this technology into the program OSCar

in which a user can control sound synthesis through programmed media embedded into web pages.

The Open Sound Control protocol has been proposed as the language for many future Virtual Reality applications. With its ability to map out just about any data with sensors and its optimized nature for web-integrated abilities, many universities are using the OSC protocol in this fashion.

Open Sound Control can also wrap other protocols inside of its data packets for improved functionality of MIDI data optimized at Open Sound Controls fast speeds. Using MIDI data with the Open Sound Control will improve the overall speed of MIDI by nearly 300 times.

Soon, other forms of data could be wrapped inside of OSC because of its ease of network transport.

The Future of Musical Protocols

While manufacturers continue to hold on to MIDI concepts, the advancement of Open Sound Control will most likely supercede MIDI because of its advanced functionality, infinite expandability and superior speed.

Superior network identification and infrastructure makes Open Sound Control much more user friendly than MIDI. The use of symbols instead of complicated strings of numbers and channels will prove to be one of the defining forces in implementing Open Sound Control.

The open source nature of Open Sound Control provides software and hardware developers with a near limitless language to program in. A language that is fully upgradeable and will not become easily obsolete as MIDI has become.

With new developments coming everyday, it is projected that Open Sound Control will soon be the new standard musical instrument protocols over the existing MIDI protocol.

Open Sound Control Website Resource and Application List

Bidule <http://plogue.com/bidule>
CPS <http://www.bonneville.nl/cps>
Csound <http://www.csounds.com/>
The EyesWeb project <http://www.eyesweb.org>
Grainwave <http://www.7cities.net/users/mikeb/GRAINW.HTM>
EtherSense: <http://www.ircam.fr/equipes/temps-reel/movement/hardware/index.htm>
Flash www.macromedia.com
Gluion <http://www.glui.de/prod/gluion.html>
HTM <http://cnmat.CNMAT.Berkeley.EDU/CAST/Server/htm.1.html>
Intakt (Native Instruments) http://www.nativeinstruments.de/index.php?intakt_us
Java <http://www.mat.ucsb.edu/~c.ramakr/illposed/javaosc.html>
Lemur <http://www.jazzmutant.com>
Max/MSP <http://www.cnmat.berkeley.edu/OpenSoundControl/Max>
Objective C <http://www.mat.ucsb.edu/~c.ramakr/illposed/objcsc.html>
Open Sound World <http://www.cnmat.berkeley.edu/OSW>
Pd <http://barely.a.live.fm/pd/OSC>
Perl <http://barely.a.live.fm/pd/OSC/perl>
PHP <http://www.a2hd.com/software>
Picker http://www.ixi-software.net/content/body_software_picker.html
Python
The Slidepipe <http://www.argobot.com/projects/slidepipe>
The SmartController
http://www.users.bigpond.com/angelo_f/smartcontroller/SmartController.html
SuperCollider <http://www.cnmat.berkeley.edu/OpenSoundControl/SuperCollider>
Reaktor http://www.nativeinstruments.de/index.php?reaktor_us
RTMix <http://meowing.ccm.uc.edu/~ico/RTMix-doc/index.html>
Ruby http://www.dadaserver.com/blendobox/ruby/osc_service.rb
Sodaconstructor <http://www.soda.co.uk/explore/osc.htm>
SpinOSC http://www.ixi-software.net/content/body_software_spinosc.html
Squeak
VisualWorks Smalltalk
The Toaster <http://www.la-kitchen.fr/kitchen.lab/hardware/toaster-en.html>
Traktor (Native Instruments) http://www.nativeinstruments.de/index.php?traktor2_us

Bibliography:

Matthew Wright, Adrian Freed

Center for New Music and Audio Technologies, U.C. Berkeley

<http://www.cnmat.berkeley.edu/ICMC97/papers-html/OpenSoundControl.html>

Copyright 1997-2005 The Center For New Music and Audio Technology (CNMAT),
UC Berkeley

<http://www.opensoundcontrol.org/>

Shannon Simpson, © 2005

<http://www.ixi-software.net/content/inf/osc.html>

Open sound World, © 2003

<http://osw.sourceforge.net/oswfaq.php>

Music and Audio Computing / © 2004 McGill University. All Rights Reserved

<http://www.music.mcgill.ca/~gary/306/week9/osc.html>