Features and Future of Open Sound Control version 1.1 for NIME

Adrian Freed, Andy Schmeder
Center for New Music and Audio Technologies
Department of Music
University of California, Berkeley
1750 Arch Street
Berkeley, CA 94720
{adrian,andy}@cnmat.berkeley.edu

Abstract

The history and future of Open Sound Control (OSC) is discussed and the next iteration of the OSC specification is introduced with discussion of new features to support NIME community activities. The roadmap to a major revision of OSC is developed.

Keywords: Open Sound Control, Time Tag, OSC, Reservation Protocols.

1. Introduction

After a brief survey of the history of Open Sound Control (OSC) we introduce the next iteration of the standard, OSC 1.1, describing new features of interest to the NIME community. We conclude by charting the immediate future of OSC and proposing a new roadmap for its evolution.

2. History and a basis for the Future

In 1997 Wright and Freed introduced Open Sound Control as:

> "a new protocol for communication among computers, sound synthesizers, and other multimedia devices that is optimized for modern networking technology. Entities within a system are addressed individually by an open-ended URL-style symbolic naming scheme that includes a powerful pattern matching language to specify multiple recipients of a single message. We provide high resolution time tags and a mechanism for specifying groups of messages whose effects are to occur simultaneously" [26]

Initially this protocol just represented recommended and actual practice at CNMAT but its use rapidly spread during the period of explosive growth of the internet as it was implemented in an increasing number of core music and media software development environments.

2.1 OSC 1.0 Specification

In 2002 the Open Sound Control 1.0 specification was published on CNMAT's website. This specification integrated important ideas proven in actual use by users with weaker original ideas expunged. The key addition of type tags enabled OSC messages to be completely self-describing. OSC addresses, type tags, and bundles together are powerful enough mechanisms for OSC to enable dynamic delegation-style object oriented programming [2, 14]

OSC is used extensively in the NIME community as a way to rapidly build ad-hoc encodings for new gestural controllers, for control structure programming [27], and also as a basis for new protocols such as TUIO [11] for multitouch surfaces and GDIF [10] for gestural data interchange.

Although never envisaged as a "standard" in the style of those created by committees of professional organizations such as the IEEE, AES or trade associations such as the MMA, many users refer to the OSC specification as a standard and the word snuck several times into our own OSC survey paper of 2003 [28].

2.2 OSC Conference 2004

In 2004 CNMAT hosted a conference on OSC bringing together many users and developers from around the world. Although no formal meetings were held to standardize OSC we polled the community for directions for the future of OSC and several presentations were directed specifically at exposing OSC weaknesses and exploring new mechanisms that are regularly needed in OSC applications, e.g. discovery, a query system [17], and a viable scheme for OSC time tags. [8].

2.3 Advances in practical use of time tags

A striking problem with OSC discussed at the 2004 OSC conference was that one of OSC's most important and interesting features had not been widely or correctly implemented: time tags. This was addressed in 2008 with the public release of a complete OSC implementation with time tag scheduling on two very different platforms -

within Cycling74 Max/MSP on Mac OS/X [16] and on a cheap (US$25) microcontroller board [17].

2.4 Looking forward: OSC 1.1

OSC's status as a standard can now be envisaged because complete implementations such as micro-OSC can be used to build interoperability and conformance testing tools [25]. Although neither the resources nor formal structure are currently in place to carry OSC forward as a standard, we have decided to update the OSC 1.0 specification to provide a stronger basis to build on– one that reflects 6 more years of experience from users, especially uses in the NIME community for representing and communicating performance gestures.

Most criticisms of OSC address errors of omission [6]: things they wish OSC would do that it doesn't. We have chosen not to address most of these in this effort and defer them to OSC 2.0, for which a roadmap is suggested in Section 7 of the paper.

3. What's New in OSC 1.1

The basic encoding format has not changed between OSC 1.0 and 1.1 so that well-formed OSC messages from 1.0 implementations will still work when processed by 1.1 implementations. The changes include clarifications regarding the role of OSC in application architectures, more specific recommendations for optional features, new data types and a small but import change to the pattern matching syntax.

3.1 OSC is a content format

OSC is often referred to as a protocol, but it is only a protocol in the weakest sense in that it defines a message format—it does not define typical features of protocols such as processing semantics (e.g. command-response patterns), error handling or negotiation. It is more accurate to describe OSC as a content format. This means that OSC can be viewed and compared with other formats such as XML [21], WDDX [18] or JSON [4]. An application that uses OSC only guarantees compatibility with OSC parsers/formatters. Inter-application protocols are of course possible using OSC as the underlying format, but the syntax and semantics of those interfaces are beyond the scope of the OSC specification. Similar to the role of WSDL and XML Schema in web services, we expect systems for service enumeration, eventing, security and choreography to be defined as higher-level protocols that use OSC as a base format or another format such as XML.

3.2 Transport and delivery clarifications

In the 1.1 specification [7] we have factored out the parts of the 1.0 specification that refer to delivery mechanisms. These have been changed and expanded to reflect common practice and are explained in Section 4.

3.3 Stream meta-data

Once OSC messages are understood as content format encodings we can clarify OSC's role in service definition and discovery. This is elaborated in section 5.

3.4 A path-traversing wildcard

OSC 1.1 inherits the path multiple-level wildcard-matching operator '//' from XPath [23]. This overcomes the limitation of the OSC 1.0 '*' operator that only matches up to '/' boundaries. It also gives concrete semantics to the string '//' in an address—which was previously not explicitly forbidden but may have resulted in inconsistent address handling due to ambiguity (e.g., in UNIX style path operations '//' is a no-op equivalent to '/').

The '//' operator enables matching across disparate branches of the address tree and at any depth as illustrated in Figure 1.

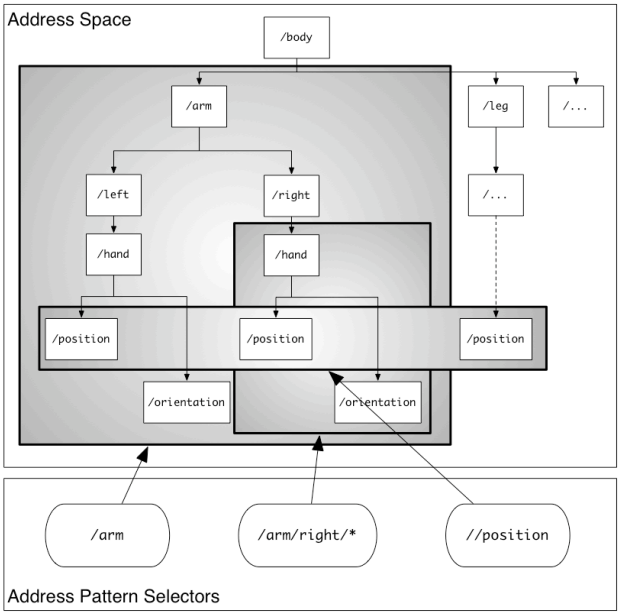

Figure 1: Address Pattern Hierarchy

This allows for some useful and interesting applications that transform OSC messages such as one that transforms OSC parameters between different units of measure, e.g.

/position/spherical (r theta phi)

is matched by the pattern //spherical to drive the transformation to:

/position/cartesian (x y z)

3.5 New types and recommended optional types

3.5.1 OSC 1.0 required types remain

The 1.0 specification describes four required standard types: integer, float, string and blob, identified by the type-tags 'i', 'f', 's' and 'b'. These are still required in OSC 1.1.

3.5.2 OSC 1.1 required types

The 1.1 specification moves some types previously specified as optional into the required list resulting in the following (Table 1):

i	Integer: two's complement int32
f	Float: IEEE float32
s	NULL-terminated ASCII string
b	Blob, (aka byte array) with size
T	True: No bytes are allocated in the argument data.
F	False: No bytes are allocated in the argument data.
N	Null: (aka nil, None, etc). No bytes are allocated in the argument data.
I	Impulse: (aka "bang"), used for event triggers. No bytes are allocated in the argument data. This type was named "Infinitum" in OSC 1.0 optional types.
t	Timetag: an OSC timetag in NTP format, encoded in the data section

Table 1

Support of these new types is the biggest burden the 1.1 imposes on OSC 1.0 implementers. Care was taken to select types that are broadly useful, easy to implement with all popular programming environments and that are actually tested extensively in real applications.

The True, False, Null and Impulse/bang types are useful for efficiently communicating bit strings, empty arguments and events, a very common scenario in robotic control, in gestural interfaces and bit twiddling for hardware development.

The OSC-timetag type is primarily used to build time synchronization protocols on top of OSC [15].

3.5.3 OSC 1.1 recommended and legacy optional types

We have added more optional reserved types including ways to describe complex numbers, double precision floating point, matrices, vectors and units qualification. These may be found in the specification and are omitted here in the interests of brevity.

3.6 Time Tag Semantics

When time tags were first introduced in OSC we expected they would be easy to implement. Unfortunately the availability to us of the required real-time features in Mac OS 9.1 and SGI IRIX turned out to be short-lived and no desktop operating system has been available since this time that can provide end-to-end communications latency guarantees and controlled, low jitter. We have confirmed the viability of the time tag semantics using a particular configuration of applications software, operating system and microcontroller. We also have discovered [8] that the OSC 1.0 semantics are not very useful for the common case of unidirectional OSC messaging. This is because the sender of OSC messages cannot know how far ahead in time to schedule OSC messages because it cannot learn of the network latency statistics seen by the receiver.

Three different and mutually incompatible uses of timetags have been employed over the years: the one outlined in the OSC 1.0 specification where the sender time tags messages to execute in the future by the receiver, one where the sender time tags messages to its own private clock to reflect in when events happened (e.g. when gesture data was acquired) and the third most common case where time tags are simply ignored and all message processing is "immediately on receipt".

Instead of outlawing these or other future scenarios we have decided to embrace all of them by simply not specifying time tag semantics at all in OSC 1.1. The specification simply provides a place in the stream for a time-tag, defines units for it and we still require that the least significant bit is reserved to mean "immediately".

4. OSC Delivery Specification 1.1

OSC messages may be sent directly without the need for framing in message oriented protocols such as UDP or MPI. Stream-oriented protocols such as TCP and serial byte streams need a framing mechanism to establish message boundaries. These streams are now required to employ SLIP (RFC1055) with a double END character encoding. This choice has been used extensively for years on the Make Controller board and in our micro-OSC work and we have established its efficiency and superiority over the OSC 1.0 size-count-preamble recommendation when recovering from damaged stream data.

This new specification enormously expands the range of protocols and hardware transports that can be used to communicate OSC encoded packets including Firewire, Ethernet, and USB (using TCP/IP); RS232 and RS422 and Serial USB and Serial Bluetooth and Serial Zigbee. USB support is especially important for the NIME community as USB provides power to remote devices and USB is currently the primary for connecting gesture controllers to computers. Serial USB is faster than the USB HID protocol typically used by such devices and with careful jitter management good enough for musical expressivity [15].

5. OSC stream meta-data: integration with discovery services and content handlers

It surprises new users of OSC to find out that no standard port for OSC is registered with the IANA and there is no OSC service registered in the usual places such as DNS-SD [3]. The reason why is easier to understand now that OSC is specified as a content format: OSC doesn't specify

service content or behavior, it is just a format for clients and servers to exchange data in to implement custom service behavior.

5.1 Specification of OSC stream meta-data

Content-format and protocol descriptions typically include a method for specification of optional parameters. We define a small number of meta-data keys appropriate to OSC streams, listed in Table 4:

version	OSC version support; 1.0 or 1.1
framing	Set to "slip" for serial transports, else omit
uri	Indicates a URI to identify the service running behind the endpoint or the source that generated the data stream
types	A string containing all the type-tag symbols supported by the endpoint/present in the stream

Table 4

5.2 Discovery with DNS-SD

Services available over IP networks can be located with DNS-SD (aka Zeroconf). These may be listed as protocol _osc._udp or _osc._tcp (with reservations, users may wish to register their own protocol that simply *uses* OSC). The attributes given in Table 4 are specified in the TXT field for DNS-SD, e.g. suppose we have a bidirectional endpoint over TCP at port 5000:

```
_osc._tcp.localhost:5000
        txtvers=1
        version=1.1
        framing=slip
        uri=http://myapp.com/
        types=ifsbhdu
```

5.3 IANA Mime Type Header for OSC data

As a practical means to store OSC data in files, we consider the file pointer to be a serial transport (thus, needing SLIP framing), and define the following IANA MIME content-type header:

```
MIME-Version: 1.0
Content-type: application/osc;
        framing=slip
        version=1.0 | 1.1
        uri=http://foobar
        types=ifsbhdu
```

5.4 Tunneling in USB Endpoints

For the transport of OSC streams through USB endpoints, we recommend putting the IANA MIME content type into the *iInterface* descriptor string for the interface containing the relevant endpoints as defined in chapter 9 of the USB 2.0 specification [19].

6. Documentation Requirement

The most unusual new requirement of OSC 1.1 is that applications and services that wish to embed the term OSC in their name or advertise Open Sound Control compatibility are free to do so providing they register their applications at http://opensoundcontrol.org. Registration takes a few minutes and benefits the OSC community immensely by avoiding duplication of effort and fostering further collaboration. This is a pragmatic compromise because there is no structure in place to formally qualify OSC implementations.

7. After OSC 1.1: a roadmap for OSC 2.0

In the next few years new, strong industry standards foundations will be built into the core infrastructure of operating systems, routers and processor and communications chips. We therefore envisage future work on OSC 2.0 should be a fresh start, a community collaboration and built wherever possible on existing standards.

7.1 Organizational Structure

OSC 1.1 was developed and asserted using the same structure that brought OSC originally to light: a benevolent dictatorship. OSC is now used so far beyond its narrow beginnings in sound synthesis control that this structure will soon have to be replaced by an organization able to integrate uses and ideas in robotics, web services, audio, music, and video and other control services.

7.2 A New Name?

The name and acronym should be changed to reflect its use in open systems control not just open sound control.

7.3 Building on Existing and Emerging Standards

In the short time OSC has existed we have experienced a rapid shift from a period of scarcity of applicable ideas and standards to build on to an overwhelming abundance. In the spirit of promoting collective understand rather than guiding a particular choice we present the following list of OSC features – extant and desired with a few corresponding existing and emerging standards that may be drawn from.

Time tag encoding	TAI64N [1]
Transport QoS	Ethernet AVB [9]
Time Synchronization	IEEE1588-2008 [12]
Query System	WSDL, WS-*
Wireless efficiency	WBXML [22]
Address Patterns	XPath [23]
Dictionaries, Attributes and Type Specification	XML-Schema-Instance [24]
Measurement units	SensorML [13]
Message semantics	XML-RPC, SOAP, REST [5]
Lightweight Implementations	EXI [20]

8. Acknowledgments

Thanks to our sponsors and partners including Sennheiser, Starkey, Meyer Sound Labs, Making Things and to the ongoing enthusiasm and energy reflected in the OSC developer group and OSC users.

References

[1] BIPM International Atomic Time. 2009. http://www.bipm.org/en/scientific/tai/tai.html.

[2] Chaudhary, A., Freed, A. and Wright, M., An Open Architecture for Real-time Music Software. in *International Computer Music Conference*, (Berlin, Germany, 2000), International Computer Music Association, 492-495.

[3] Cheshire, S. DNS Service Discovery (DNS-SD). 2009. http://www.dns-sd.org/.

[4] Crockford, D. JSON: Javascript Object Notation. 2009. http://www.json.org/.

[5] Fielding, R.T. Architectural styles and the design of network-based software architecturesThesis University of California, Irvine 2000.

[6] Fraietta, A. Open Sound Control: Constraints and Limitations *NIME*, nime.org, Genova, 2008.

[7] Freed, A. Open Sound Control 1.1 Specification. 2009. http://opensoundcontrol.org/spec-1_1.

[8] Freed, A. Towards a More Effective OSC Time Tag Scheme *Open Sound Control Conference*, CNMAT, Berkeley, CA, 2004.

[9] Garner, G.M., Feifei, F., den Hollander, K., Hongkyu, J., Byungsuk, K., Byoung-Joon, L., Tae-Chul, J. and Jinoo, J. IEEE 802.1 AVB and Its Application in Carrier-Grade Ethernet [Standards Topics]. *Communications Magazine, IEEE*, *45* (12). 126-134.

[10] Jensenius, A.R., Kvifte, T. and Godøy, R.I. Towards a gesture description interchange format *2006 conference on New interfaces for musical expression (NIME)*, IRCAM, Centre Pompidou, Paris, France, 2006.

[11] Kaltenbrunner, M., Bovermann, T., Bencina, R. and Costanza, E. TUIO: A protocol for table-top tangible user interfaces *6th International Workshop on Gesture in Human-Computer Interaction and Simulation*, 2005.

[12] NIST Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems

[13] . 2008. http://ieee1588.nist.gov/.

[14] OGC Sensor Model Language (SensorML). 2009. http://www.opengeospatial.org/standards/sensorml.

[15] Peters, N., Baltazar, P., Place, T., T, L. and Jensenius, A.R. Addressing Classes by Differentiating Values and Properties in OSC *International Conference on New Interfaces for Musical Expression (NIME)*, Genova, 2008

[16] Schmeder, A. and Freed, A. Implementation and Applications of Open Sound Control Timestamps *ICMC*, ICMA, Belfast, Ireland, 2008.

[17] Schmeder, A. and Freed, A. uOSC: The Open Sound Control Reference Platform for Embedded Devices *NIME*, Genova, Italy, 2008.

[18] Schmeder, A. and Wright, M. A Query System for Open Sound Control *OpenSoundControl Conference*, CNMAT, Berkeley, CA, 2004.

[19] Simeonov, S. WDDX (Web Distributed Data eXchange) 1998. http://www.openwddx.org/.

[20] usb.org USB 2.0 Specification.

[21] w3.org Efficient XML Interchange Working Group. 2008. http://www.w3.org/XML/EXI/.

[22] w3.org Extensible Markup Language (XML). 2008. http://www.w3.org/XML/.

[23] w3.org WAP Binary XML Content Format. 1999. http://www.w3.org/TR/wbxml/.

[24] w3.org XML Path Language (XPath). 1999. www.w3.org/TR/xpath.

[25] w3.org XML-Schema-Instance. 2004. http://www.w3.org/TR/xmlschema-1/.

[26] Wright, M., Dannenberg, R., Pope, S., Rodet, X., Serra, X. and Wessel, D. Panel: Standards from the Computer Music Community *International Computer Music Conference*, International Computer Music Association, Miami, FL, 2004.

[27] Wright, M. and Freed, A., Open Sound Control: A New Protocol for Communicating with Sound Synthesizers. in *International Computer Music Conference*, (Thessaloniki, Hellas, 1997), International Computer Music Association, 101-104.

[28] Wright, M., Freed, A., Lee, A., Madden, T. and Momeni, A., Managing Complexity with Explicit Mapping of Gestures to Sound Control with OSC. in *International Computer Music Conference*, (Habana, Cuba, 2001), International Computer Music Association, 314-317.

[29] Wright, M., Freed, A. and Momeni, A., Open Sound Control: State of the Art 2003. in *International Conference on New Interfaces for Musical Expression*, (Montreal, 2003), 153-159.