Addressing Classes by Differentiating Values and Properties in OSC

Timothy Place, Electrotap, tim@electrotap.com
Trond Lossius, BEK - Bergen Center for Electronic Arts, lossius@bek.no
Alexander R. Jensenius, U.o. Oslo & Norwegian A.o. Music, a.r.jensenius@imv.uio.no
Nils Peters, CIRMMT, McGill University, Montreal, nils.peters@mcgill.ca
Pascal Baltazar, GMEA, pb@gmea.net

ABSTRACT

An approach for creating structured Open Sound Control (OSC) messages by separating the addressing of node values and node properties is suggested. This includes a method for querying values and properties. As a result, it is possible to address complex nodes as classes inside of more complex tree structures using an OSC namespace. This is particularly useful for creating flexible communication in modular systems. A prototype implementation is presented and discussed.

PROBLEM

The original idea of OSC is that it is tree-structured into a hierarchy called the address space, where each of the nodes has a symbolic name and is a potential destination of OSC messages. In contrast to the static schema of MIDI, the open nature of OSC means that the address space is defined and created by the "implementor's idea of how these features should be organized". This open approach has made OSC useful in a broad range of applications, and adaptable to situations not foreseen by its developers. However, this lack of standardization in namespace schemas is also likely a ma jor reason that OSC has not gained more widespread use in commercial software applications.

STANDARDIZING MEMBERS

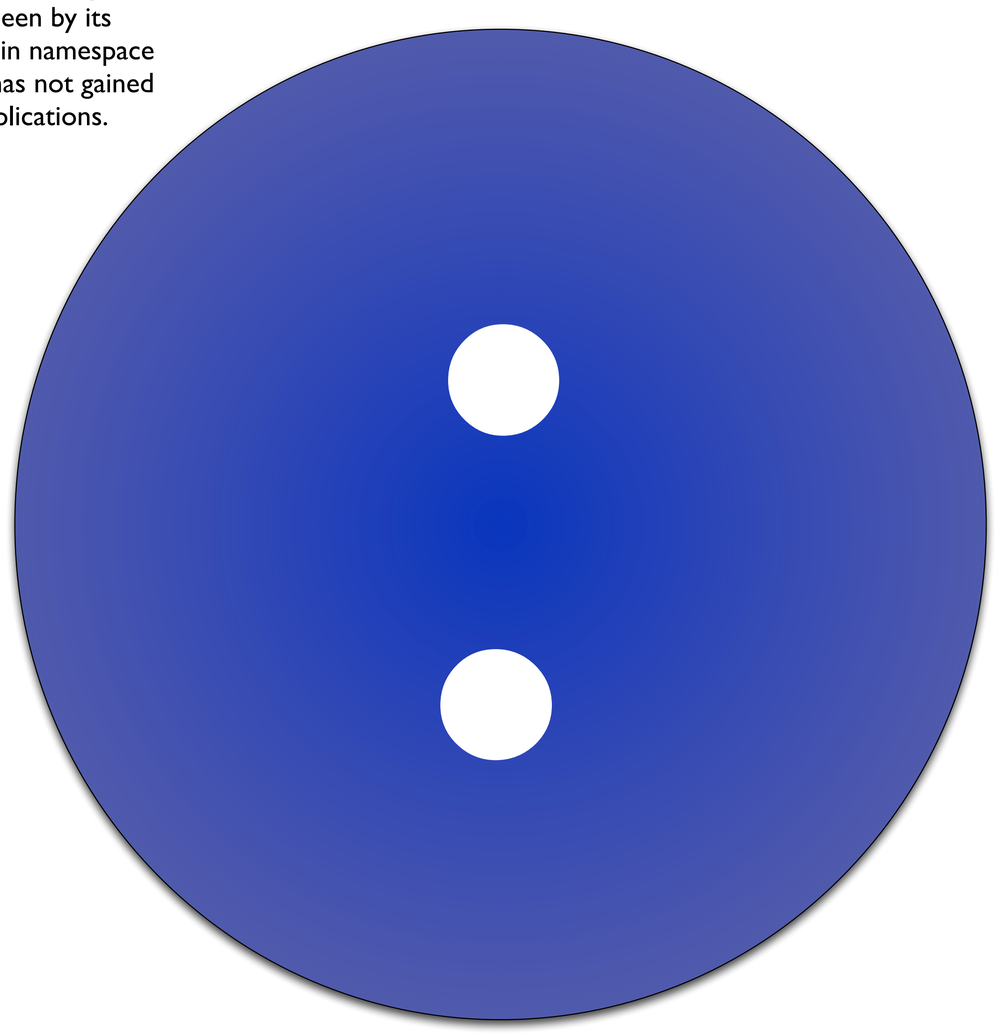
To make working with classes in OSC practical, it is important to have some standard members in place. At present we recommend standardizing the following member methods, and reserving their syntax:

`:/get`
returns the value of the node.

`:/dump`
returns the state of the node, which is to say the values of all of the properties including the value itself.

`:/namespace`
returns the namespace implemented at this node.

`:/catalog`
returns an enumeration of available options for a node, if relevant.



INTRODUCING THE COLON

In addition to the ASCII symbols already reserved for specific purposes within the OSC protocol [10], we introduce the colon ":" as a separator between the OSC address of a node and the namespace for accessing the members of the node:

```
<node address> <value>
<node address>:<member address> <value>
```

The former message sets the value of the node just as it would using the existing OSC conventions. This is because a property named value is considered to be implicitly addressed if there is no specific member address given. The latter form calls or sets a member of the node. The member itself is addressed using a fully-qualified OSC namespace. Again using gain as an example, we can send two messages: one for setting the unit property and one for setting the value.

```
/module/audio/gain:/unit midi
/module/audio/gain 120
```