SOUND COMMUNICATION: A STANDARD SYNTAX FOR INTER-APPLICATION, INTER-DEVICE AND INTER-PLAYER COMMUNICATION OVER OSC

Scott Hewitt and Pierre Alexandre Tremblay

CeReNeM
University of Huddersfield
Huddersfield, UK
scott@ablelemon.co.uk, p.a.tremblay@hud.ac.uk

ABSTRACT

This paper offers a standard message format for easy intercommunication over a network, between laptop performers within a drop-in improvisation session. It is based on OSC-like reserved namespaces, namely /test, /setup, /chat, /app, /user, /time, /documentation and /hardware. We detail the syntax of use and provide worked examples. The proposed standard offers interoperability, extensibility and flexibility across network applications.

1. INTRODUCTION

This proposed standard has been developed to address difficulties often encountered when collaborating with other laptop musicians.

Network connectivity is a unique feature of the laptop instrument and one which offers exciting new performance opportunities such as telepresence. [1] When establishing network connectivity, technologies such as TCP/IP, DHCP, LAN networking and WIFI facilitate inter-device communication. Within a Max/MSP environment connectivity can be established using the udpsend and udpreceive objects. However, once the network is established, the musical inter-communication often relies upon an ad-hoc, instance specific protocol. This lack of a standard communication method distracts from rehearsal time and easy code/device substitution often causing a lower quality of performance, or at best a reduced utilisation of the unique set of features offered by the laptop instrument.

2. BASIC STANDARD FEATURES

The proposed standard makes no claim on the transit fabric or underlying protocol; rather, it seeks to offer a method for encapsulating data within the selected medium. The standard is a method of structuring dynamic, real-time data to provide easy management, utilisation and archival across a network environment.

The standard utilises a familiar, human readable data layout similar to OSC [2] namespaces and web URLs. Reserved namespaces are used to order and encapsulate data providing an immediate context and methodology for use. Namespace allocation is based on proposed data target rather than its source.

In use, this standard encourages a server/client topography, fosters the development of network agnostic applications, allows easy expansion and proposes a human readable data structure.

3. STANDARD IN DETAIL

3.1. Syntax

The syntax is based on reserved namespaces which are intended to be the sole occupants of the primary namespace. This is the fundamental element of this scheme and a requirement for compatibility.

The proposed First Layer Namespaces would be 1)`/test`, 2)`/setup`, 3)`/chat`, 4)`/app`, 5)`/user`, 6)`/time`, 7)`/documentation` and 8)`/hardware`. These namespaces are designed to separate data streams into logical, manageable locations that enable interconnection and flexibility, while providing an immediate context and expatiation as to style, nature and possible role.

3.2. /test Namespace

The `/test` namespace provides an immediate communication channel which is used to test application communication. The protocol and physical connection can be confirmed using other tools like ping.

This `/test` space should remain unencrypted and consequently should not be used as an extended communication channel. An auto test response should not be employed on this space: rather a request for auto response can be issued.

This request should take the form of

```
/test check + ip + time
```

and awaits the response of

```
/test auto + ip + time
```

Usage of the `/test` namespace offer a simple test structure within final connectivity layer and basic real-time network/application profiling.

3.3. /setup Namespace

This namespace would be reserved within all spaces and should be implemented throughout detection levels. The purpose of the space is to allow dynamic configuration and easy upgrading of detection code. The `/setup` space carries commands that configure which strings should be detected at each level and provides an easy way of mapping the current data layout. The `/setup` command should be the primary detection string within any space so to allow easy remapping and should act within the same namespace. The inclusion of the `/setup` space allows dynamic configuration and an auto-upgrade within predefined namespace structures.

The `set` message is reserved within `/setup` and is implemented like

`/namespace_path/setup set /new_namespace`.

eg. `/user/james/setup set /pan`

3.4. /chat Namespace

The `/chat` namespace is designed for instant messaging intercommunication between parties participating within a performance. Messages are delivered in the following format:

`/chat/<name> message content`

eg. `/chat/james hello`

Though it is intended for a chat channel to exist directly within this namespace, it is also permissible for more focused chats to occur within a sub-namespace. It is however desired that the primary chat space remains publicly accessible at all times in essence offering a foyer to the more general conversations.

`/chat` therefore offers an inbuilt communication channel, which is simple to implement and offers publicly accessible communication.

3.5. /app Namespace

Reserved for applications, this namespace is designed to offer a stable network-aware interface for common programs. Data targeted towards standard application interfaces and functions can be carried within this space allowing easy interaction between applications across a multi platform, constant method. Responsibility for specific applications would be managed by program authors who would look to maintain consistency through versions.

In application a program using `/app` should transmit data within an item-specific channel determined by the application name and then a unique identifier such as.

`/app/SC3/yuytr5654/`

This location will allow simple interface of major parameters within an application. Second layer names are intended to be application names with a formalised scheme and data layout. Locating application data streams within a dedicated space prevents cross contamination and simplifies variable addressing.

Programs supporting `/app` should look to publicly publish protocol details to a central repository to facilitate interoperability. `/app` is expected to hold established, unchanging interconnects rather than fluid structures.

Utilisation of `/app` offers stable, network-transparent communication, easy inter-application communication and an open communication standard.

3.6. /user Namespace

The `/user` namespace offers almost complete flexibility. Except for a user_id, no other layout is stipulated. Suitable for ad-hoc setups and dynamic experimentation it is hoped that `/documentation` and `/setup` namespaces will exist to facilitate interaction with these dynamic structures.

It is also appropriate to embedded other reserved namespace with `/user` to signify individual ownership of resources or locality. Such as,

`/user/james/app/SC3/yuiysdg/`

A SuperCollider instance specific to user James

`/user/james/hardware/SY99/hjkuids/`

A hardware device, an SY99 in this case, specific to user James. This designation indicates that either the hardware is made available through user James, useful for fault testing, or else it is specific to user James.

3.7. /time Namespace

`/time` provides a referential time within the structure of the performance for simple synchronisation and queuing of events across the performance network. Its inclusion facilitates both record and playback functions embedded within the data structure.

The `/time` namespace contains `/ms`, `/sec`, `/mins`, `/hour` and `/day` each containing the current value relative to the start of the performance with a level of resolution as indicated by the namespace. Not all levels may be run or required throughout a performance and consequently may be disconnected at any point.

The purpose of the higher unit namespace such as `/hour` is to allow an opportunity to reduce bandwidth requirements when either a lower level of synchronisation is not required or available capacity or latency of transit makes such lower time periods impractical.

3.8. /documentation Namespace

The `/documentation` namespace offers a method of directing participants to archival material, manuals and usage information within the protocol. Useful when an unknown client/application is in use. Licensing details can also be communicated through this medium.

Reserved message within `/documentation` are `all-messages`, `help`, `manual` and `url`. All these commands function within there embedded namespace and return information within there own additional namespace.

For instance the use of `all-messages` is,

```
/app/a1/gy/documentation all-messages
```

List all commands relevant to instant of a1 into namespace,

```
/app/a1/gyuhuh/documentation/all-messages
```

The usage of `help` within the `/document` namespace would be,

```
/app/app2/gdhjfgs/documentation help
```

This message request help information concerning app2 and returns into namespace,

```
/app/app2/gdhjfgs/documentation/help
```

Reserved message `manual` would be used as below,

```
/app/app3/gdhjfs/documentation manual
```

Delivers location of manual for retrieval outside of standard. Manual location returned into

```
/app/app3/gdhjfs/documentation/manual
```

To request the 'home' location of the application the `url` message could be used as indicated,

```
/app/app4/g/documentation url
```

Provides relevant online information location of app4 for further information beyond that required to perform. Url delivered into,

```
/app/app4/g/documentation/url
                        http://www.site.com
```

3.9. /hardware Namespace

The `/hardware` namespace allows the connection of hardware devices to the network structure offering the ability to connect a device locally to an individual performer or globally across the entire networked transit.

In application, a hardware device should transmit data within a item specific channel determined by the manufactures device name and then a unique identifier such as, `/hardware/SY99/id6653hgnk/`.

This allows multiple identical devices to be employed simultaneously within the same network.

4. EXAMPLES OF STANDARD IN USE

Below are different situations and examples of possible standard use.

4.1. Single laptop Performer

The standard is used to facilitate inter-application communication using the local host network. In this example a performer is using a Hardware keyboard connected to a Max/MSP GUI which is controlling a SuperCollider Synth.

```
/app/SC3/"User_ID"
```

is used to communicate between MAX GUI and Super Collider Synth, and

```
/hardware/SY99/"Unique_ID"
```

is used to connect custom built hardware to applications

4.2. Laptop Orchestra

A Laptop Orchestra performing a work using pre-coded applications requiring basic synchronisation, and inter-laptop communication at both the application and user levels.

```
/test
```

is used to establish state of network connectivity,

```
/time/sec
```

is used to maintain orchestra time within a second, and

```
/chat/"User_ID"
```

is the message used to send messages to either individual users.

For instance,

```
/chat/James Sam: hi
```

is used to send a message to the user Sam, or to send a message to all participants, we use

```
/chat Sam: hello all

/app/user/"Users_Unique_ID"
```

is used to allow cross-network communication between users' audio applications

4.3. Multiple Live Coding Improvisers

In this example multiple live-coding improvisers communicate within there own specific `/user` channels using there own ad-hoc commands and data.

The `/test` and `/chat` channel are used as indicated in the previous example (4.2).

```
/user/"Users_Unique_ID"
```

is used to dynamically created data streams such as,

```
/user/"Users_Unique_ID"/ap4/Volume up
```

and

```
/user/"Users_Unique_ID"/ap4/Pitch 440

/hardware/SY99/"Device_ID"
```

is used to transmit hardware gestures, and

```
/user/"Users_Unique_ID"/ap4/documentation
```

is used to distribute documentation about ap4 as requested.

5. CONCLUSION

The method described above has been tested thoroughly in a number of performance situations and has been received with enthusiasm by participants. It is hoped that this standard will be endorsed and implemented widely offering a truly interoperable control method for multi-application, computer music creation.

6. REFERENCES

[1] P. Rebelo, A. Renaud, B. Alain. The Frequencyliator - Distributing Structures for Networked Laptop Improvisation In *Proceedings of the 2006 Conference on New*

Interfaces for Musical Expression pages 53-56, Paris, France 2006

[2] M. Wright. Opensound control specification, 2002.Available: http://opensoundcontrol.org/spec-1_0