Client Requirements for the 1394 Peer-to-peer Data Transport Stack

1394 Printer Working Group

July 4, 1999

Revision 1.2

1. Requirements placed on the transport stack by its clients

The following is a list of requirements the client places on the thick transport stack. The requirements are placed on everything from the API between the client and the transport stack to the API between the remote transport stack and it's client. The requirements are the sum-total of all requirements of all clients. A device that knows its clients do not require some things do not need to implement them. The requirements are split into two sections: musts and wants. They are intentionally brief, with definitions of terms following each requirement.

1.1 Musts

1.1.1 Support multiple, concurrent, independent, symmetrical connections

- **Multiple, concurrent:** allows for more than one connection at a time. The actual number of connections supported is implementation-specific.
- **Independent:** activity on one connection has no effect on other connections.
- **Symmetrical:** allows either endpoint to open and close a connection, and send data.
- **Connection:** a well-bounded communication path between two endpoints. The endpoints can be on the same device or on different devices.

1.1.2 Provide in-order, byte-stream and in-order, datagram services

- **In-order:** Data is delivered to the receiving endpoint in the same order as it was presented by the sending endpoint.
- **Byte-stream:** Data is delivered as a stream of bytes. The stream of bytes is not guaranteed to be delivered to the receiving endpoint with the same boundaries as it was presented by the sending endpoint. For example, a stream of 80 bytes of data may be presented as 4-20 byte buffers, but delivered as 2-40 byte buffers.
- **Message:** Data is guaranteed to be delivered to the receiving endpoint with the same boundaries as it was presented by the sending endpoint. For example, if data is presented by the sending endpoint in a buffer of 30 bytes, it must be delivered to the receiving endpoint in a buffer of 30 bytes. The transport stack may limit the size of datagrams. It does not have to support segmentation and reassembly of client buffers that do not fit in a single transport packet.

The selection of byte-stream vs. datagram service is made at connection time and is unchangeable during the duration of that connection.

1.1.3 Provide a service-name based connection service

Endpoints on a specific device may be referenced by their service name. This allows connections to be opened without any knowledge of the underlying layer's implementation of sockets, etc.

1.1.4 Provide a service directory mechanism

Devices shall provide a mechanism to return a list of the names of services provided by each node on the device. There shall be a separate list for each name. Not all services need be advertised in the list.

1.1.5 Transparently handle transient link interruptions

The transport stack shall handle transient link interruptions without affecting the endpoints. These link interruptions include: temporary cable disconnect, 1394 bus reset, etc. A “transient” link interruption is defined to be short and non-catastrophic with respect to the connection, the services provided, and human time. A temporary cable disconnect is explicitly defined to be longer than one second, since this is defined to be a human interaction.

1.2 Reliability

Clients expect and require a high-level of reliability when using the transport stack. The transport protocol may rely on lower layers in the transport stack to provide the high-level of reliability.

1.3 Wants

1.3.1 Connectionless service

A non-bounded communication path between two endpoints. Data may be sent without "opening" a connection.

1.3.2 Data tagging

Data can be tagged as "special data" by the sending endpoint. The transport will indicate to the receiving endpoint that the data is tagged. This is also known as out-of-band data. The special data is synchronous with the rest of the data.

1.3.3 Selectable quality of service

The ability to adjust various quality of service parameters, including:

- Isochronous delivery
- Priority
- Propagation Delay
- Rate of transfer (bandwidth)

1.3.4 Bridging

1394 busses can be connected together with bridges. We would like to be able to connect across bridges.

2. Internal Thick Transport Stack Requirements

The following are the requirements the transport stack places on itself.

2.1 Musts

2.1.1 Be application and O/S independent

The transport stack shall not put any requirements on the format of the application data, nor shall it interpret that data in any way. The transport stack shall work with any application that correctly uses the appropriate interfaces. The transport shall be implementable under any operating system.

2.1.2 Do not preclude concurrent operation of other protocol stacks

Devices may implement and use other protocol stacks concurrently with this transport stack.

2.1.3 Provide efficient data transmission

Prevent unnecessary bus traffic. Balance bus traffic with protocol overhead.

2.2 *Wants*

2.2.1 Reuse existing protocols

Save time by reusing existing protocols, rather than inventing new ones.

3. Glossary

- **Byte-stream service:** Data is delivered as a stream of bytes. The stream of bytes is not guaranteed to be delivered to the receiving endpoint in the same form as it was presented by the sending endpoint. For example, a stream of 80 bytes of data may be presented as 4-20 byte buffers, but delivered as 2-40 byte buffers.
- **Connection:** a well-bounded communication path between two endpoints. The endpoints can be on the same device or on different devices.
- **Datagram service:** Data is guaranteed to be delivered to the receiving endpoint in the same form as it was presented by the sending endpoint. For example, if data is presented by the sending endpoint in a buffer of 30 bytes, it must be delivered to the receiving endpoint in a buffer of 30 bytes. The transport stack may limit the size of datagrams. It does not have to support segmentation and reassembly of client buffers that do not fit in a single transport packet.
- **Independent:** activity on one connection has no effect on other connections.
- **In-order data:** Data is delivered to the receiving endpoint in the same order as it was presented by the sending endpoint.
- **Multiple, concurrent:** allows for more than one connection at a time.
- **Symmetrical:** either endpoint can open and close the connection, and send data.
- **Transient link interruption:** a short and non-catastrophic interruption with respect to the connection, the services provided, and human time.

4. Issues

- Should compliance require implementation of symmetrical connections even if the clients of a particular implementation don't require it? (e.g., a peripheral with nothing but servers may not need to provide its clients with an "Open Connection" service) – Closed – No
- How important is it that we match our protocol to limitations in current host operating systems? – Closed – It is important. We try to test our concepts with the O/S vendors.
- Can the transport really limit datagrams to a single transport packet, or do the clients require larger datagrams?
- Is connectionless service really a want, or should it be moved to a must?