VOEvent Transport Protocol

Version 2.0

IVOA Recommendation 2017-03-20

Working group
   Data Access Layer / Time Domain Interest Group

This version
   http://www.ivoa.net/documents/REC-VTP-2.0-20170320/20170320

Latest version
   http://www.ivoa.net/documents/REC-VTP-2.0-20170320

Previous versions
   http://www.ivoa.net/documents/Notes/VOEventTransport/

Author(s)
   Alasdair Allan, Robert B. Denny, John D. Swinbank

Editor(s)
   John D. Swinbank

Abstract

The IVOA VOEvent Recommendation (?) defines a means of describing transient celestial events but, purposely, remains silent on the topic of how those descriptions should be transmitted. This document formalizes a TCP-based protocol for VOEvent transportation that has been in use by members of the VOEvent community for several years and discusses the topology of the event distribution network. It is intended to act as a reference for the production of compliant protocol implementations.

Status of This Document

This document has been reviewed by IVOA Members and other interested parties, and has been endorsed by the IVOA Executive Committee as an IVOA Recommendation. It is a stable document and may be used as reference material or cited as a normative reference from another document. IVOA’s role in making the Recommendation is to draw attention to the specification and to promote its widespread deployment. This enhances the functionality and interoperability inside the Astronomical Community.

A list of current IVOA Recommendations and other technical documents can be found at http://www.ivoa.net/Documents/.
Figure 1: VOEvent distribution system architecture showing the relationships between the various network roles.

Contents

Versioning

The first version of the VOEvent Transport Protocol submitted for IVOA approval is version 2.0. Earlier versions were described informally in an IVOA Note (?), which this document supersedes.

Acknowledgements

Thanks to Robert W. White (ex LANL, now at NREL), Phillip Warner (ex NOAO), Robert Seaman (NOAO). John D. Swinbank acknowledges support from the European Research Council via Advanced Investigator Grant 247295.

1 Introduction

The VOEvent standard (?) defines a means of representing transient celestial events with an implicit request for action on the part of the recipient. The VOEvent standard is transport neutral: it does not take a position on the mechanism by which the event should be transmitted from its author to interested recipients. However, it encourages the construction of “a robust general-purpose network of interoperating brokers” for event transmission.

To date, a number of different event distribution networks have been prototyped and met with varying degrees of technical success and community adoption. However, as the number of interested participants grows, and next-generation large-scale survey instru-
Figure 2: Diagram based on ? showing the VOEvent Transport Protocol (VTP) in the context of the wider IVOA architecture.

ments such as LSST\(^1\), LIGO\(^2\), LOFAR\(^3\) and SKA\(^4\), which promise event rates ranging up to the millions per day, are developed and begin to become available, it is clear that a standard, interoperable mechanism for event communication is required. It is such a mechanism that this document describes.

The purpose of the protocol described herein is to transport a VOEvent document from its sender to one or more interested recipients. To achieve this, we envision three distinct network roles: authors, which create events; brokers, which receive events from authors and distribute them, and subscribers, which receive and (if appropriate) act upon the events. Refer to Figure ?? for an illustration, while Figure ?? shows how this integrates with the wider IVOA architecture. Note that a single entity may perform more than one role within the network: for example, creating events and distributing its own creations (combining the author and broker roles) or receiving events from a broker and redistributing them to a list of subscribers (combining the subscriber and broker roles).

Building upon this architecture, a strongly-connected set of brokers which subscribe to each other’s event streams and redistribute to their subscribers (the “VOEventNet backbone”) provides a fault-tolerant system which is resilient against the failure of one or more network entities. Such a backbone system is already under construction by members

---

\(^1\)Large Synoptic Survey Telescope; [http://www.lsst.org/](http://www.lsst.org/)


\(^3\)Low Frequency Array; [http://www.lofar.org/](http://www.lofar.org/)

\(^4\)Square Kilometre Array; [http://www.ska-telescope.org/](http://www.ska-telescope.org/)
of the VOEvent community.

The protocol described herein is intentionally as simple as possible while still accomplishing the required task. More complex protocols will be required for addressing advanced use-cases, handling extremely large event or subscriber numbers\(^5\), or providing value added services\(^6\). These fall outside the scope of the current document.

Although this document refers specifically to VOEvents, the protocol places only quite minimal requirements on the payload. We expect that a future evolution of this protocol would provide a convenient means of delivering diverse message types, perhaps including portfolios or containers of VOEvents, or even non-IVOA standard messages.

## 2 Terminology

Throughout this document, we adopt the terminology of RFC 2119 \(^?\). In particular:

- The word “must” indicates an absolute requirement of the specification;
- The word “should” indicates behaviour that is normally included in implementations of the specification, but there may exist valid reasons for excluding it in particular circumstances;
- The word “may” indicates purely optional behaviour which is permitted according to this specification.

## 3 Common characteristics

### 3.1 Design goals

The VOEvent Transport Protocol, hereafter VTP, provides a simple means of transporting VOEvent documents from authors through brokers to subscribers.

VTP transmits no more than one VOEvent in each transaction. If multiple documents are to be transmitted, multiple transactions must take place.

VTP delivers VOEvents to eligible subscribers which exist on the network at the time of transmission. It does not buffer events for later transmission. Subscribers who wish to retrieve historical events should consult an event repository.

VTP is non-transformational on VOEvents being transmitted: the document delivered to a subscriber should be bit-for-bit identical to that provided by an author. If an intermediary wishes to modify or annotate the VOEvent, they should not edit the document in transport, but rather generate a new document to supplement or replace it.

VTP does not provide a transport-level means of annotating or otherwise embellishing VOEvent documents, or of providing stream-level metadata.

VTP values simplicity of design and operation to lower the barrier to entry. It is not intended to meet every use case. As per Section ??, it is anticipated that some VOEvent-based services will require more complex protocols.

VTP is independent of implementation: conforming network entities should be able to interoperate seamlessly, even when derived from different codebases.

\(^5\)Refer to ? for a discussion of scalability

\(^6\)For example, Svom, the Space-based multi-band astronomical Variable Objects Monitor, [http://www.svom.fr/](http://www.svom.fr/), is developing an approach based on XMPP \(^?\).
3.2 Network layer

VTP operates over TCP (?) connections, and relies on TCP’s guaranteed error-free in-order delivery of data: no checksum or digest data is included. All documents are sent over the TCP connection preceded by a 4-byte network-ordered\(^7\) count, followed immediately by the payload data. The 4-byte count is interpreted as a 32-bit integer equal to the number of payload bytes following the count bytes. The payload is considered an opaque collection of bytes at this level\(^8\).

3.3 Message format

Throughout this document, the term “message” refers to a complete VTP message, including both the initial byte count and the message payload. The payload is an XML document. It must consist of an XML declaration followed by, in order, optional XML comments, a single `<VOEvent />` or `<Transport />`\(^9\) element, and more optional XML comments. It must validate against either the VOEvent XML schema\(^{10}\) or the Transport XML schema (Appendix ??). Messages may be conveniently referred to by their payload type (viz. “VOEvent message”, “Transport message”).

3.4 Broker behaviour

Although the simplest broker implementation may simply forward all unique events it receives, either directly from authors or from other brokers, to all of its subscribers, this behaviour is not required. Instead, the broker may provide “added-value” services which limit how messages are redistributed. For example, a broker may make arrangements with some or all of its subscribers to filter the events it receives, and forwarding only those events that fulfil some predefined criteria. Similarly, brokers may limit access to some clients based on various criteria (§??).

VTP does not provide in-band notification of these per-broker details. For example, the protocol does not make an author submitting to a filtering broker aware that their event might not be sent to all of the broker’s subscribers, and, similarly, it does not make a subscriber of a filtering broker aware that they might not receive a complete set of events. It is the responsibility of authors and subscribers to ensure that the brokers they use provide the services they require. Brokers should clearly advertise any added-value behaviour they provide, for example on a website or through the IVOA registry (??).

4 Network nodes

The VOEvent network consists of three types of nodes (refer to Fig. ??):

- Author
- Broker
- Subscriber

---

\(^7\) As defined by ?; also called “big-endian” ordering.

\(^8\) As a result, the format of the document being transmitted is opaque to the transport layer. Therefore both ASCII and UTF-8 are equally supported.

\(^9\) `<Transport />` elements are used by the VTP system itself and are invisible to end-users: see Section ?? for details.

\(^{10}\) [http://www.ivoa.net/xml/VOEvent/VOEvent-v2.0.xsd](http://www.ivoa.net/xml/VOEvent/VOEvent-v2.0.xsd)
As described in ?, it is expected that authors and brokers will be registered with the IVOA registry\textsuperscript{11}. It is not necessary for subscribers to register.

The flow of messages is over three types of connections:

- Author to Broker
- Broker to Subscriber
- Broker to Broker

Each type of connection is discussed qualitatively below.

4.1 **Author to Broker**

When an author wants to submit a VOEvent document to the network, it constructs a message encapsulating that document, opens a TCP connection to a broker, sends the message, waits for a response from the broker, and then closes the TCP connection. The response from the broker is a message containing a Transport document.

4.2 **Broker to Subscriber**

When a subscriber wants to receive VOEvent traffic, it opens a TCP connection to a broker. This connection is kept open continuously. When the broker receives a VOEvent message, it relays a copy of that message to each connected subscriber\textsuperscript{12}. Thus, a subscriber must continuously listen on the TCP connection and be prepared to receive new messages at any time, even when it is busy processing a previously received message. When a subscriber receives a VOEvent message from its broker, it must respond with an appropriate Transport message.

4.3 **Broker to Broker**

Traffic between brokers uses the preceding methods. Each broker takes the role subscriber as far as every other broker is concerned. A broker that wishes to receive a feed from another broker should connect to that broker’s subscriber port. No special protocol features are needed.

5 **Connection Maintenance**

All connections over which a broker sends VOEvent messages are kept open continuously. However, basic TCP does not provide any dead-peer indication\textsuperscript{13}. Further, network infrastructure devices might sever a TCP connection after some period of inactivity. This gives rise to the need for keep-alive messages. After no more than 90 seconds of inactivity on any given connection, the broker must send a Transport amalive message, to which the subscriber must reply with a copy of that message plus some optional identification information. The message format is described in Sections ?? and ??.

\textsuperscript{11}This is dependent on the VOEvent Registry Extensions (?), which are not fully standardized or widely deployed at time of writing. Unregistered services may therefore be deployed until such time as the relevant registry support becomes available.

\textsuperscript{12}If filtering as described in §?? is being carried out, the message may be sent only to a subset of the subscribers.

\textsuperscript{13}TCP does support a “keep-alive” service, but it is not universally available (?).
At both ends of the continuous connection, the node either expects to receive an `iamalive` message or expects to receive the response to its `iamalive` message. If not seen, the node should assume that the connection has been lost or the peer is dead. At this point, the node that was responsible for opening the connection may attempt to re-initiate it. The use of geometric back-off algorithm may help alleviate network load.

6 Transport messages

Transport messages are VTP messages §?? containing a `<Transport />` element. There are four classes of `<Transport />` element, distinguished by their `role` attribute:

- `iamalive` (Connection maintenance);
- `authenticate` (Authentication request/response);
- `ack` (VOEvent successful receipt acknowledgement);
- `nak` (VOEvent unsuccessful receipt acknowledgement).

All Transport messages have the same general syntax, and are defined by the Transport schema (Appendix ??). The connection maintenance and receipt acknowledgement message types are described in detail in this section; the authentication message type has a special role which is described in Section ??.

6.1 `iamalive` message

The `iamalive` message is indicated by a role equal to `iamalive`. The `<Origin />` element contains the IVOID of the broker which is managing the connection. The `<TimeStamp />` element contains the date and time at which the message was generated formatted as per §3.3.7 of ??. This time should be provided in UTC, and may include a “Z” timezone indicator.

Listing 1: Sample `iamalive` message.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<trn:Transport role="iamalive" version="1.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://ivoa.net/xml/Transport/v1.1
http://ivoa.net/xml/Transport-v1.1.xsd">
  <Origin>ivo://invalid.broker/example#</Origin>
  <TimeStamp>2001-01-01T00:00:00Z</TimeStamp>
</trn:Transport>
```

6.2 `iamalive` response

The `iamalive` response is an extension of the initial `iamalive` message. It also has a role of `iamalive`. The `<Origin />` element is preserved unchanged from the `iamalive` being responded to (that is, it contains the IVOID of the broker). It may include an additional `<Response />` element containing a URI identifying the subscriber15. It may also include

---

14International Virtual Observatory identifier; ?.
15If the subscriber is registered with the IVOA registry, this may be an IVOID, but registration is not required.
a <Meta /> element with <Param /> sub-elements which give additional information about
the subscriber or any other relevant information. <Param /> elements have no content and
must contain name and value attributes. The names and values may be any string. The
<TimeStamp /> element contains the date and time at which the response was generated,
formatted as per §3.3.7 of ?. This time should be provided in UTC, and may include a
“Z” timezone indicator.

Listing 2: Sample iamalive response.

```xml
<?xml version='1.0' encoding='UTF-8'?>
<trn:Transport role="iamalive" version="1.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://ivoa.net/xml/Transport/v1.1
http://ivoa.net/xml/Transport-v1.1.xsd">
  <Origin>ivo://invalid.broker/example#</Origin>
  <Response>ivo://invalid.subscriber/example#</Response>
  <TimeStamp>2001-01-01T00:00:00Z</TimeStamp>
  <Meta>
    <Param name="IPAddr" value="10.0.0.0" />
    <Param name="Contact" value="name@subscriber.invalid" />
  </Meta>
</trn:Transport>
```

6.3 VOEvent message receipt response

The VOEvent message receipt response is similar to the iamalive response except the
role is either ack or nak, the <Origin /> is the IVOID of the just-received VOEvent
message, and an optional <Result /> element may accompany the <Param /> elements.
<Result /> may contain any string; it is recommended that it contain a human-readable
error message if role is nak. The <TimeStamp /> element contains the date and time at
which the response was generated, formatted as per §3.3.7 of ?. This time should be
provided in UTC, and may include a “Z” timezone indicator.

The nak response indicates that the recipient is unable or unwilling to take responsi-
bility for this message. This may be because, for example, the message fails to validate
as a valid VOEvent, or because it was received from an unauthorized client (§??). A nak
response is not appropriate if the sender is able to accept the message but then decides
not to redistribute it (for example, if it is a duplicate of an event which has already been
distributed: Section ??).

A nak response should be regarded as a permanent failure state: delivery of the VO-
Event message which was met with the nak to the recipient which replied with the nak
should be aborted. If the VOEvent message was being transmitted over an author-to-
broker connection, the author may identify the cause of the failure (for example by making
use of the information in the <Meta /> element of the nak), construct a corrected VOEvent
message and attempt a repeat submission. If the VOEvent message was being transmitted
over a broker-to-subscriber connection, the broker should abandon the attempt to deliver
this message.

Listing 3: Sample VOEvent message receipt response indicating successful transmission
(ack).

```xml
<?xml version='1.0' encoding='UTF-8'?>
```
Listing 4: Sample VOEvent message receipt response indicating unsuccessful transmission (nak).

<?xml version='1.0' encoding='UTF-8'?>

7 Protocol operation

This section describes the operation and sequencing of VTP operation for each end of a connection between an author and a broker, as well as between a broker and a subscriber. See Section ?? above for a qualitative discussion of the protocol from the viewpoint of each entity.

7.1 Author sending to broker

The author initiates a TCP connection to the broker. The broker may choose to accept or reject that connection based, for example, on an access control whitelist (§??). If the author is rejected, the connection is terminated. If the connection is accepted, the author creates a VOEvent message by prepending a byte count to the VOEvent document (§??) and transmits it to the broker. The author should then wait for a VOEvent message receipt response (§??) from the broker. The author may use this to determine whether the message has been successfully delivered. The connection is then closed.
If a receipt response is not received, the author should assume that a temporary failure has prevented the broker from accepting the message for distribution. The author may close the connection and retry.

This transaction is illustrated in Figure ??.

7.2 Broker receiving from author

The broker awaits incoming TCP connections from authors. When a connection is received, the broker may choose to accept or reject the connection based, for example, on an access control whitelist (§??). If the author is rejected, the connection is terminated. Otherwise, the broker waits to receive a VOEvent message from the author. When the VOEvent is received, the broker should test the message for validity. The broker must return a VOEvent message receipt response (§??) to the author indicating that it has either accepted (ack) or refused (nak) the VOEvent message. The connection is then closed.

This transaction is illustrated in Figure ??.

7.3 Broker sending to subscriber

The broker awaits incoming TCP connections from subscribers. When a new connection is received, the broker may choose to accept or reject the connection based, for example, on an access control whitelist (§??). If the subscriber is rejected, the connection is terminated. Otherwise, the broker adds the subscriber to its distribution list.

Periodically, at intervals of no more than 90s (§??), the broker must send an `iamalive` message (§??) to the subscriber. The subscriber must reply with an `iamalive` response (§??). If the broker does not receive `iamalive` response messages from the subscriber in a timely fashion, it may assume that the subscriber is dead or gone and close TCP connection.
**Figure 4:** Transport protocol at broker receiving from author.

**Figure 5:** Transport protocol at broker sending to subscriber.
When the broker has a new VOEvent message ready for distribution, it is sent to the subscriber. The broker receives a VOEvent message receipt response (§??) in reply. The broker may use this to determine whether the VOEvent message has been accepted (ack) or refused (nak). The broker must not attempt to repeat delivery of the message if a nak is received.

If a receipt response is not received, the broker should assume that a temporary failure has prevented the subscriber from accepting the message. The broker may attempt redelivery. At the discretion of the broker, repeated failures to receive timely receipt responses may be grounds to terminate the connection.

The TCP connection remains open, and the iamalive exchange continues, until either the subscriber explicitly closes the connection or stops sending iamalive response messages, thereby implicitly indicating that the connection is closed.

These transactions are illustrated in Figure ??.

7.4 Subscriber receiving from broker

The initiates a TCP connection to the broker. The broker may choose to accept or reject that connection based, for example, on an access control whitelist (§??). If the subscriber is rejected, the connection is terminated. Otherwise, the connection remains open, and the subscriber begins receiving messages.
Periodically, at intervals of no more than 90s (§??), the subscriber should expect to receive an *iamalive* message (§??) from the broker. The subscriber must reply with an *iamalive* response (§??). If the subscriber does not receive *iamalive* messages from the broker in a timely fashion, it may assume that the broker is dead or gone and close TCP connection. The subscriber may attempt to re-connect to the broker. Re-connection attempts should use a geometric back-off algorithm.

When a VOEvent message is received, the subscriber may test it for validity. The subscriber must return a VOEvent message receipt response (§??) to the broker indicating that it has either accepted (*ack*) or refused (*nak*) the VOEvent message.

The TCP connection remains open, and the *iamalive* exchange continues, until either the subscriber explicitly closes the connection or stops receiving *iamalive* messages, which indicates that the connection has been terminated.

These transactions are illustrated in Figure ??.

### 8 De-duplication

In a network topology like that illustrated in Figure ??, multiple brokers service potentially overlapping sets of authors and subscribers. As per Sections ?? and ??, brokers will subscribe to each other’s event feeds to ensure that their subscribers have access to the full range of available events.

In this situation, there is a risk of event loops developing on the network: broker A receives an event from B and forwards it to its subscriber list, which includes A, which forwards it to its subscriber lists, which includes B, and so on. In order to prevent this, each broker must process each unique VOEvent message it receives a maximum of once.

Note that it is now established practice to distribute different descriptions (e.g. VOEvent 1.1 and 2.0) of the same celestial event with the same IVOID. Consequently, an IVOID is not a unique identifier of a particular VOEvent message, and is not, therefore, suitable for use in network de-duplication.

Instead, we regard two messages as being the same if the content between the opening `<` and the closing `>` of the `<VOEvent />` element is bit-for-bit identical, including all white space characters. The implementation of this check is left to the discretion of the broker.

In the event that some future revision of the VOEvent standard adopt an identifier which is unique to the message, rather than to the celestial event, it would be preferable to use that for de-duplication rather than calculating a hash over the event content.

### 9 Limiting access

For administrative or security reasons, broker administrators may wish to limit access to the services they provide to a restricted range of clients. These restrictions may be required on either or both of the connection types in VTP: author to broker (a limit on which authors can publish through a given broker) or broker to subscriber (a limit on which subscribers a broker is willing to provide with event streams).

---

16 Refer to [http://www.ivoa.net/pipermail/voevent/2012-March/002836.html](http://www.ivoa.net/pipermail/voevent/2012-March/002836.html) and subsequent discussion. Note that the VOEvent standard uses the now-deprecated term “IVORN” in place of IVOID.
17 Appropriate techniques may include directly comparing the bitstream (which would necessarily mean storing an archive of previously-processed events) or calculating a hash function such as SHA1 (?) over the event contents and storing the result.
Two mechanisms are may be applied within the VTP framework to address these requirements.

9.1 IP address whitelisting

If the broker knows a priori the IP addresses or ranges from which authorized authors or subscribers are permitted to connect (a “whitelist”), it may simply deny connections from addresses which fall outside that range. If the sets of authorized authors and subscribers are not the same, separate whitelists may be implemented.

This mechanism imposes significant administrative overhead on the broker owner if large and complex whitelists are required. Further, it is of limited applicability if the clients to be serviced are using dynamic IP addresses (that is, addresses which change periodically).

9.2 Cryptographic signatures

A digital signature scheme enables the recipient of a digital message to verify the identity of its author (?). By requiring authors and subscribers to apply appropriate signatures to VOEvent and Transport messages, it may be possible for a broker to verify their identity and restrict the services made available to them.

Various digital signature schemes which are appropriate for use with VOEvent and other XML documents have been suggested (?). At time of writing, none have seen significant adoption. Given that, this version of the VTP standard does not specify a particular approach, nor require that any form of cryptographic authentication be available. However, the <Transport /> schema provides an <authenticate> message type, shown in Listing ??, which may be used to implement either the ? or ? scheme, or as the basis for some other approach.

See also ?? for a description of the authentication schemes available for use across the Virtual Observatory.

Listing 5: Sample authenticate message.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<trn:Transport role="authenticate" version="1.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://ivoa.net/xml/Transport/v1.1 http://ivoa.net/xml/Transport-v1.1.xsd">
  <Origin>ivo://invalid.broker/example#</Origin>
  <TimeStamp>2001−01−01T00:00:00Z</TimeStamp>
</trn:Transport>
```
A Transport schema

```xml
<?xml version="1.0" encoding="utf-8" ?>
<xs:schema xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="Transport">
    <xs:complexType>
      <xs:sequence>
        <xs:element minOccurs="1" name="Origin" type="xs:anyURI" />
        <xs:element minOccurs="0" maxOccurs="1" name="Response" type="xs:anyURI" />
        <xs:element minOccurs="1" name="TimeStamp" type="xs:dateTime" />
        <xs:element minOccurs="0" maxOccurs="1" name="Meta">
          <xs:complexType>
            <xs:sequence minOccurs="1" maxOccurs="1">
              <xs:element minOccurs="0" maxOccurs="unbounded" name="Param">
                <xs:complexType>
                  <xs:attribute name="name" type="xs:string" use="required" />
                  <xs:attribute name="value" type="xs:string" use="required" />
                </xs:complexType>
              </xs:element>
              <xs:element minOccurs="0" maxOccurs="1" name="Result" type="xs:string" />
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
      <xs:attribute name="role" type="roleType" use="required" />
      <xs:attribute name="version" type="xs:string" use="required" />
    </xs:complexType>
  </xs:element>
</xs:schema>
```

B Version history

B.1 Revised since v2.0-PR-20161230
- Split history by PR revision (Appendix ??).
- Note that different implementations of VTP should be interoperable (§??).

B.2 Revised since v2.0-PR-20160503
- Replaced all references to “IVORN” with “IVOID”.
- Make clear that authors and brokers should be registered with the IVOA registry (§??).
- Substantially trimmed the material on cryptographic signatures (§??). Made it clear that this version of VTP does not specify a particular approach.
• Describe appropriate actions when a receipt response message is not received (§§?? & ??).

• Indicate that timestamps should be in UTC (§§??, ??, ??).

B.3 Revised since IVOA Note v1.1

• Add Section ??, describing design goals of the protocol.

• Add Section ??, detailing requirements for message de-duplication to avoid network loops.

• Specify an explicit interval requirement to connection maintenance messages (§??).

• Clarify the semantics of nak Transport messages (§??).

• Make it explicit that brokers should not attempt to repeat delivery of messages which meet with a nak on the first attempt: VTP does not support the concept of a “temporary failure” (§§??, ??).

• Reword the descriptions of protocol operation so that they describe only the traffic exchanged over the network and not the implementation of the various entities (§??).

• Allow timezone specification in iamalive <TimeStamp /> elements (§§??, ??, ??).

• Remove identifying information from example XML documents (§§??, ??, ??, ??).

B.4 Revised since IVOA Note v1.0

• Add an optional <Result /> sub-element (containing text) within the optional <Meta /> element. This is intended to convey details on errors encountered if the Transport response is nak but may also be used for informational purposes in ack messages.