VOEvent v2 (Seaman et al, 2011) is a well established standard for describing transient celestial events.

- Who, what, when, where, why, how, etc.

- It does not specify how to communicate that description to others.

  - Modulo a brief discussion of the names and roles of various entities who might interact with VOEvent packets.

VOEvent Transport Protocol aims to provide a minimal baseline for users to subscribe to streams of VOEvents.

- Not intended to be exclusive of other approaches: you can still use e-mail, SMS, XMPP, … as required.
ORIGINS & VERSIONING

- Original VTP described in an IVOA Note by Alasdair Allan & Bob Denny in 2009 (VTP 1.0 and 1.1):
  - http://www.ivoa.net/documents/Notes/VOEventTransport/
- The current effort is to standardize a protocol that is wire compatible with the Allan & Denny note.
  - Existing implementations continue without changes.
- The text has been substantially reworked for clarity and to avoid ambiguity.
  - Particularly regarding the semantics of some messages.
- This revised text is presented as VTP 2.0.
A simplified version of the roles in the VOEvent v2 standard: we do not distinguish “author” vs “publisher”.
PROTOCOL DESIGN

➤ Simple TCP based client-server architecture.
  ➤ Implement a network entity in a few lines of Python.
➤ Transmits a single VOEvent per transaction.
➤ Non-transformative on VOEvents being transmitted.
➤ Two classes of messages supported:
  ➤ VOEvent documents themselves;
  ➤ Transport messages.
    ➤ Also XML documents, schema included in standard.
    ➤ Four message types: ack, nak, iamalive, authenticate.
INTERACTION BETWEEN ENTITIES

**Author to Broker**
- Author initiates TCP connection
- If connection accepted, network sends receipt response
- If connection refused, close TCP connection
- Send VOEvent message
- Receive receipt response
- Close TCP connection

**Broker to Subscriber**
- Broker initiates TCP connection
- If connection accepted, network sends receipt response
- If connection refused, close TCP connection
- Send VOEvent message
- Receive receipt response
- Repeat for each message
- Close TCP connection

**Subscriber from Broker**
- Subscriber initiates TCP connection
- If connection accepted, network sends receipt response
- If connection refused, close TCP connection
- Send receipt response
- Receive VOEvent message
- Repeat for each message
- Close TCP connection

**Broker from Author**
- Broker initiates TCP connection
- If connection accepted, network sends receipt response
- If connection refused, close TCP connection
- Receive VOEvent message
- Send iamalive
- Receive iamalive response
- Repeat periodically
- Close TCP connection
IMPLEMENTATIONS

➤ Comet (Python 2.7 released, 3.4+ on master)
  ➤ Author + broker + subscriber
  ➤ http://comet.transientskp.org/

➤ Dakota VOEvent Tools (C#/.net)
  ➤ Author + broker + subscriber
  ➤ http://voevent.dc3.com/

➤ PyGCN (Python 2.6+)
  ➤ Subscriber only
  ➤ https://github.com/lpsinger/pygcn

➤ GCN client demo (C)
  ➤ Subscriber only
  ➤ http://gcn.gsfc.nasa.gov/voevent_client_demo.c
PR as of 2016-05-04; RFC period opening soon.

For now, comments & questions welcome either to me directly or to TDIG and/or DAL mailing lists.