

Querying events with Simple Event Access Protocol

Matthew J. Graham
California Institute of Technology

Abstract The Simple Event Access Protocol (SEAP) provides a standard mechanism to query services containing VOEvent packets. Not only are positional searches supported but also queries against who created or published the event, how the event was observed, and what was observed, both in terms of measured quantities, e.g. flux, and the interpreted nature of the event, e.g. supernova. Queries against specific values as well as ranges of values are possible. Services offering this capability will be discoverable in the VO registry.

Introduction

From an event producer's perspective, VOEvent can be a fire-and-forget technology: once an event packet has been cast into the ether, it ceases to be of any further operational concern. For an astronomer, however, this ephemerality can be disconcerting – what if they missed the initial announcement or want to mine past events looking for correlations and causations, either between events or against known sources?

Within the VOEvent infrastructure, **repositories** provide a persistence layer to address these issues. Yet not all repositories are equal; there is scope for a spectrum of them, ranging from repositories that carry all events ever published by anyone to bespoke ones which only store events from a particular source, e.g. the Catalina Real-time Transient Survey, or of a particular type, e.g. more than 90% plausibility of being a supernova.

The Simple Event Access Protocol (SEAP, [1]) provides a standardized interface to repositories to facilitate interoperability. It is analogous to the other 'simple' VO data access protocols – Simple Cone Search (SCS, [2]) and Simple Image Access (SIA, [3]) – in providing a lightweight parameter-based mechanism for querying VOEvents that have been stored in some fashion.

Basics

A call to a SEAP service consists of a basic HTTP GET request including a list of ampersand-delimited keyword-value arguments:

```
http://some.seap.server?key1=value1&key2=value2...
```

The keywords identify which aspect of a VOEvent is being queried, e.g. spatial position or reported event type – asteroid, supernova, GRB, etc. -- and the values describe the specific value or range of values of interest, e.g. $\text{mag} < 15$ or within the last 24 hours. When multiple keywords are used, the results, if any, will satisfy all constraints: for example, a search employing both a spatial keyword and one relating to object type will only return events of that type within that region.

The default response from the service is a VOTable containing at least two pieces of information for each event that matches the query:

- the full IVOA identifier (IVORN) of the VOEvent
- an URL from which the full VOEvent packet may be retrieved

Other data might also be returned but that is very much a service-dependent issue. SEAP services might also offer the same response content in alternate formats, e.g. as a KML document or as a JSON structure, but that also will be service-dependent. Nevertheless such additional features will be described in the metadata record for a particular SEAP service in the VO registry.

A failed query should respond in a manageable fashion so that the query originator understands why their search did not work. This is both in case of error, e.g. a parameter value has been specified in an incorrect format, and where things are not necessarily erroneous, e.g. a spatial search exceeds the spatial coverage of the available data.

Keywords

The set of possible keywords that can be used in a SEAP call is summarized in Table 1. These essentially allow a query to be made against any of the major sections of a VOEvent packet, i.e. any of the Who, What, WhereWhen, How or Why elements. The data type of the corresponding value to a particular keyword must be in the specified format, e.g. the 'ivorn' value must be an IVOA IVORN-compliant string. The keywords can optionally be prefaced with a scheme of "seap", e.g. `seap:concept='GRB'`, to unambiguously identify them as belonging to the SEAP domain (namespace). This might be necessary for a service that augments its SEAP capability with extra parameters of similar meaning, e.g. size could also refer to a limit on the number of returned results.

Table 1. The possible keywords that can be used in a SEAP call

Keyword	Data Type	Description
response.content	string	The format of the response; possible values include: ivorn (default), full, custom, and kml
ivorn	IVORN	The identifier of the VOEvent
stream	IVORN	The identifier of the stream from which the packet originated
authorivorn	IVORN	The identifier of the author of the VOEvent
contactname	string	The contact name of the person who created the VOEvent
how	string	How the observation that resulted in the VOEvent was performed
pos	float/pos range	The position of a region of interest
size	float	The radius of a region of interest
time	ISO-8601	A time (interval) of interest
param	param range	A reported parameter of interest
concept	string	What the underlying physical event is thought to be
citation.cited	boolean	Which events have cited this one?
citation.role	string	The role of any citing event; possible values are: followup, supersedes, and retraction

Value rules

Multi-component values, e.g. a RA, Dec position, use a comma (",") as the separator between components. Embedded white space is not allowed and two successive commas indicate an empty item, as does a leading or trailing comma.

Most queries requesting matches to specific values will be against IVORNs or strings whilst those involving regions, times and parameters of interest will commonly be looking for a match against a numeric range of values. These use a forward slash (“/”) as the separator between range limits. The range is defined as inclusive, i.e. including the range limits, and if a third value is specified, it is a step size for traversing the indicated range. An open range may be specified by omitting either range limit: for example, `pos=120/240,-45` will search for events with $120 \leq RA \leq 240$ and $Dec \leq -45$.

Certain values permit a qualifier, e.g. time, and these use a semicolon “;” as the separator between the keyword values and a qualifier string.

The logical NOT of a value is specified by preceding the value with a “!”. For example, `pos=!120/240`, will search for events outside the region $120 \leq RA \leq 240$.

Finally, keywords are not case sensitive but string values are.

Positional range

The full syntax for values of the “pos” keyword is:

```
pos = minval1/maxval1,minval2/maxval2;coordsys
```

where `minval1/2` are (optional) lower limits and `maxval1/2` are (optional) upper limits for their respective coordinates and `coordsys` is an (optional) keyword indicating the coordinate system to use. The default coordinate system is ICRS – RA and Dec in decimal degrees – but alternate systems such as “GALACTIC” are allowed, although whether they are supported is a service-dependent issue.

Temporal range

The full syntax for values of the “time” keyword is:

```
time = minval/maxval;type
```

where `minval` and `maxval` are (optional) lower and upper limits in ISO-8601 format and `type` specifies the Date/Time reference as detailed in the VOEvent standard [4]. Possible values for `type` are:

- 1 `isotime` – this denotes the time that the observation reported in the VOEvent packet was made
- 2 `pubtime` – this denotes the time that the VOEvent packet was created/published

The default value is `isotime` and this value is assumed if no qualifier is specified.

Parametric range

The full syntax for values of the “param” keyword is:

```
param = "name",minval/maxval;"unit"
```

where `name` is the (optional) parameter name, `minval` and `maxval` are the (optional) lower and upper limits to the parameter value, and `unit` is an (optional) qualifier specifying the parameter unit. For example, `param="mag1",18/20` will search for events with a parameter named “mag1” that has a value between 18 and 20. When the parameter value is a string then `minval` and `maxval` need to be identical and string equivalence will be sought for, e.g. `param="filter";"Gunn z"/"Gunn z"` will search for events with a parameter named “filter” and a value of “Gunn z”.

In VOEvent, Params are often arranged in (named) Groups. The name used in the param value can therefore be qualified by the Group name (as a prefix and using a period (“.”) as a separator between the Group name and the Param name) if the search is for a specific parameter within a specific Group. For example, `param="Ast1.mag1"` will look for a Param called “mag1” within a Group called “Ast1”. Matching just on the Group name can be achieved by omitting any Param name but keeping the period separator; for example, `param="Ast1."` will find all events with a Group called “Ast1”.

VOEvent 1.1 does not support the specification of data types on parameters. It is therefore left to the individual service to interpret parametric ranges; however, the general assumption should be that when numerical characters are used in a range expression, e.g. 18/20, then a numerical search range is intended

and the query should be parsed as such. For example, `param="mag1",19/` will search for events where the numerical value of the parameter named “mag1” is greater than 19 rather than those where the value of the parameter named “mag1” come after “19” in an alphanumeric ordering. Note, however, that in VOEvent 2.0 parameters can have a data type associated with them.

Data type information is available for parameters in the registry record for a particular VOEvent stream and so a SEAP service could always retrieve the relevant record and check whether a given query is syntactically correct, e.g. whether the specified parameter takes numerical values if a numerical range is present.

Examples

Fig. 1 shows the following positional constraints around a point with RA=120, Dec=15:

- `pos = 120,15`
- `pos = 120,15 & size = 0.5`
- `pos = 119.5/120.5,14.5/15.5`
- `pos = 120,14.5/15.5`
- `pos = 119.5/120.5,15`
- `pos = 120,15 & size = !0.5`
- `pos = !119.5/120.5,!14.5/15.5`

(b) illustrates how the “size” keyword is used to specify the radius of a region of interest centered on the point identified by the “pos” keyword. (f) shows how a negation appears on the “size” keyword when it is used – any negations on the “pos” keyword are invalid in this context.

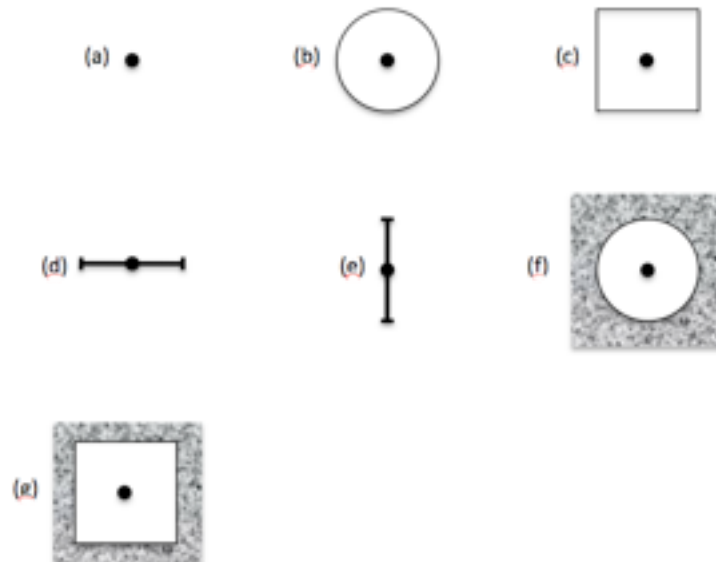


Figure 1. Different positional constraints. In (f) and (g), the shaded region is the area being defined, i.e. the circle and the square denote regions of exclusion.

Registering SEAP services

The VO registry allows users to discover astronomical resources – data collections and services – that match specified metadata requirements. A service supporting a SEAP interface should be registered as a Service or subresource (e.g. a VOEventServer) with a capability of type “voe:SimpleEventAccess”.

References

- [1] Graham, M.J., Allan, A., Seaman, R., Williams, R., Auden, E., Denny, R., Rots, A., Warner, P., *Simple Event Access Protocol (SEAP)*, v1.0, 2010
<http://www.ivoa.net/Documents/latest/SEAP.html>
- [2] Williams, R., Hanisch, R., Szalay, A., Plante, R., *Simple Cone Search*, v1.03, 2008
<http://www.ivoa.net/Documents/latest/ConeSearch.html>
- [3] Tody, D., Plante, R., *Simple Image Access Specification*, v1.0, 2009
<http://www.ivoa.net/Documents/SIA/20091116>
- [4] Seaman, R., Williams, R., Allan, A., Barthelmy, S., Bloom, J., Graham, M., Hessman, F., Marka, S., Rots, A., Stoughton, C., Vestrand, T., White, R., Wozniak, P., *Key Event Reporting Metadata (VOEvent)*, v1.11, 2006
<http://www.ivoa.net/Documents/latest/VOEvent.html>