

Event Handling with SkyAlert

Roy Williams
California Institute of Technology

The Skyalert project collects and disseminates observations about time-critical astronomical transients, and adds annotations and intelligent machine learning to those observations. The information can be “pushed” to subscribers, who may be either humans or machines that control telescopes. Subscribers can prepare precise “triggers” to decide which events should reach them and their machines, that may be based on the generic event, or on the specific vocabulary of parameters that define a particular type of observation. The system will not be centralized, but rather a set of interoperating nodes with caching. The twin thrusts are automation of process, and discrimination of interesting events.

An important qualitative change is happening in astronomy. The ability to survey large areas of the sky deeply and repeatedly, and to process the data in real time, has opened a whole new field of research: time domain astronomy. The sky is now studied as an ever-changing collection of dynamical phenomena, with moving objects, objects that change in brightness, or those that appear in an explosive manner. Many of these phenomena can be understood only through time-domain studies, and often require a rapid follow-up using a variety of ground-based telescopes.

Events are Universal: The ideas here apply not just to astronomy, but in other areas of society where events must be followed by rapid, often expensive actions. In all cases, it is good to keep the data and those evaluations in a structured manner—what we call a *portfolio*. The components of Skyalert are:

- A web-based event broker, allowing subscription so that events are delivered in near-real time.
- A web-based publishing system, so that authenticated users can inject events that may be delivered to others.
- An event repository, storing all events that come through the broker, and allowing bulk queries and drill-down.
- An interoperable event broker, conforming to international (IVOA) standards for event services.
- Open-source software to allow local implementations as well as the web-based application.

Skyalert.org [1] is a prototype system that has handled many different event streams. It is designed to work well with amateur astronomers and students, as well as professional astronomers scheduling follow-up on the big NSF-funded telescopes, operated by NOAO and its partners, or on the emerging world-wide robotic telescope networks.

Skyalert Design Goals

Many new event streams are coming online or are planned for the future. Thus, it is time to deploy a system that can work with events in general, rather than a custom solution for each survey or spacecraft. The system should encourage adoption of a standard framework and interoperability, avoiding duplication of effort in implementing similar systems.

From the web interface, subscribers can arrange to get event packets immediately (“pushed”). Users can build “alerts” consisting of a trigger and an action. Whenever an event is injected, it is tested against the trigger, and if it passes, then the action occurs. Actions can be messaging (email, IM, robotic telescope), fetching from archives, or running a computation. One objective of Skyalert is to encourage robotic follow-up in near real time to catch the most exciting, largely unknown, physics of rapid transients. Once such observations are made, we will encourage submission of that data to Skyalert as events, which will

be added to the web page for the original event, and of course the follow-up event can be sent to subscribers and other event brokers.

Authors of events build semantic content in advance, through an “event stream”, which expresses the common metadata shared by all the events that belong to that stream[2]. The stream has a description of these events, and semantic definitions of the parameters that will be used to describe each event of the stream. There can also be a template-builder to define the “Overview” presentation of the event. Streams can be private or public, with access roles according to owner, group, and public. Authors can gain trust by injecting and removing “test” events in a private stream.

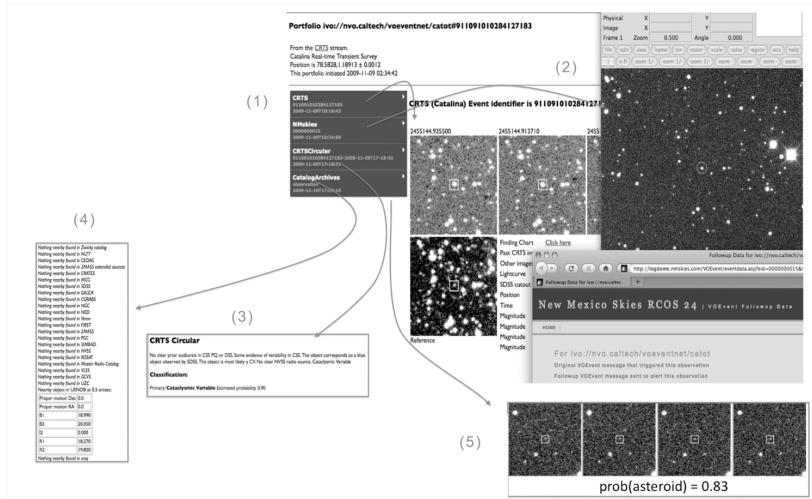


Figure 1: Here is the concept of Portfolio. This example portfolio from Nov 9, 2009, shows several events: (1) the original observations from the CRTS survey (top gray bar), (2) a robotic follow-up observation from the New Mexico Skies observatory 18 minutes later, (3) a human assessment of the meaning of the portfolio, and (4) a collection of archival results. A Portfolio can also contain events from data mining modules, a sort of a “computational follow-up”, where complex data is reduced to a simple quantity. Here (5) four images are analyzed for motion.

Each event will have a collection of data that we call a *portfolio* of data: a collection including spectra, cutouts, etc, from many different surveys and observations. A portfolio is defined through a citation mechanism inherent in the VOEvent packet [3], where one event can cite another. Thus an event with no citation becomes its own portfolio, but an event with a citation to another joins the portfolio that the other is in.

Skyalert integrates multi-sourced heterogeneous data, so long as the event author builds the metadata (event stream) through the web-based authoring interface. The evidence is thus collected, for past evidence of variability, for nearness to a galaxy, for excess infrared flux, and so on, that can help to classify that source. The data will be available not only to human clickers, but also through an Annotator API, which will be called in real time for data mining when the event is injected to Skyalert.

Skyalert allows events to become part of conversations in both traditional and new social media. One priority will be integration with the Central Bureau for Astronomical Telegrams [4], the Minor Planet Center [4], Astronomers Telegram [5], etc. Other media, such as Twitter, Facebook, Second Life, Google Wave, etc, should be used as places for humans and machines to see and comment on events. Skyalert also offers recent events in modern formats: Twitter, Facebook, Microsoft Worldwide Telescope, Google Sky, etc.

Skyalert allows software components called annotators, which can analyze the existing portfolio and add new data to it, for example fetching archival images, or comparing FITS images to detect motion of a source, or reduction of a spectrum to a single number of interest. These high-level criteria can then be used to trigger action, as with any other parameter in the portfolio.

Skyalert will be part of an emerging international infrastructure of astronomical transients using the VOEvent protocol [3]. There are now many streams and many follow-up facilities, and a general agreement to use the “VOEvent” protocol for representing and communicating astronomical transients. The system will use the international distributed Registry of the Virtual Observatory to enable publishing and discovery of event streams [2], so that all interested parties can subscribe to future events and browse past events from that stream. When a new event stream or service is created, its detailed metadata will be written and recorded by the author in a standard way.

Skyalert has a detailed access policy, so that people can confidently use Skyalert in the early stages of a project, when events are for a small group only. Skyalert is also available to download and install locally, and is built on Django; it requires Python, a relational database such as MySQL, with web service provided by Apache.

Real-time follow-up observations are crucial to a full understanding of the science of transients. These authors are already collaborating with event providers, building new collaborations with providers of events, with the amateur astronomy community, and with owners of robotic telescopes.



Figure 2: SkyAlert has several types of display and outreach. Above left is a display of recent events, with click-through to detailed portfolio. Below left is the full sky with recent events, shown with Microsoft Worldwide Telescope. WWT can show many image surveys, and access NED, Simbad, and other catalog resources. At right are the Twitter and Facebook streams, with detections automatically broadcast from the reduction pipeline, and also the human-built interpretation several hours later.

How to Use SkyAlert

Event and Stream Browsing

The front page of the SkyAlert web page shows graphically the most recent portfolios and which streams have been active (see Fig 2, top left) . Recent events can also be seen by selecting a specific stream (“browse event streams”), and selecting portfolios from there. Each portfolio is shown as a collection of events, and each event can be seen in different representations. Fig 3 shows a portfolio display: by hovering the mouse over each event, different representations can be chosen: Overview, or Params, or raw XML.



Portfolio [ivo://nvo.caltech/voeventnet/catot#911091010284127183](#)

[See context in WorldWideTelescope](#)

From the [CRTS](#) stream.
Catalina Real-time Transient Survey
Position is 78.5828, 1.18913 ± 0.0012
This portfolio initiated 2009-11-09 02:34:42



Move the mouse over the gray bar for other representations.

Figure 3: The portfolio display shows each event of the portfolio as a gray bar (four events here), and for each event there are three representations: Overview, Parameters, and Raw XML. The sky images here are part of the Overview of the CRTS event. The portfolio can also be viewed in Microsoft Worldwide Telescope (top right).

Available event streams (as of Jan 2010) are shown in the screen shot of Fig 4. For each stream, you can see the semantics and description, or go to the portfolios that are instances of that stream.



[View Streams](#) | [View All Events](#) | [View Alerts](#)

Logged in as: roy
(Roy Williams)
([logout](#))

Streams

Here are the streams known to SkyAlert. Click the Detail link to view or edit the stream. Some streams have first 'portfolio'. Click the All Events link to see all the events from the stream, and pointers to any portfolios of which

Stream Name	Streams	Portfolios	Description
AAVSO	(Stream)	(Portfolios)	AAVSO Alerts
ATA	(Stream)	(Portfolios)	The Allen Telescope Array (0.5 to 11 GHz)
CRTS	(Stream)	(Portfolios)	Catalina Real-time Transient Survey
CRTS2	(Stream)	(Portfolios)	CRTS 1.5m Transients
Fermi	(Stream)	(Portfolios)	Fermi events
GALEX	(Stream)	(Portfolios)	Galex TDS Alerts
LOFAR	(Stream)	(Portfolios)	LOFAR radio telescope array
MOA	(Stream)	(Portfolios)	MOA Microlensing Survey
OGLE	(Stream)	(Portfolios)	OGLE Microlensing Survey
PI_OF_SKY	(Stream)	(Portfolios)	Candidate Optical Transient
POSS	(Stream)	(Portfolios)	Possible Supernova from the Puckett Observatory Supernova Search
PTF	(Stream)	(Portfolios)	Palomar Transient Factory Event Stream
SWIFT	(Stream)	(Portfolios)	SWIFT GRB alerts
VERITAS	(Stream)	(Portfolios)	VERITAS VHE (E > 100 GeV) gamma-ray alerts

Figure 4: Some event streams whose metadata has been ingested to SkyAlert.

The semantics part of the stream describes the stream and its author, and has a list of parameters that describe an event of that stream. Parameters can be grouped, and each has a description, units, data type, and UCD (Unified Content Descriptor [6]). Parameter values that are URLs are identified with a UCD `meta.ref.url`, indicating that the data content is not in the value itself, but is a complex data object under the link.

Creating Alerts

An alert is a combination of a Boolean trigger expression and an action: the trigger decides if the action will happen. Each trigger runs on each portfolio as soon as it can, but not all triggers can run on all

portfolios; a trigger needs a specific set of streams before it can run. For example, an alert may need an event from stream *A* and one from stream *B*: the *A* stream may be the newly-discovered transient, and the *B* events come from an annotation. Other triggers may require only a single event from a single stream. In any case, whenever a new event is added to a portfolio, that portfolio is checked against all triggers to see which can run.

When creating a new alert, Skyalert needs to know which event streams are necessary to run it. If an alert needs multiple streams, multiple selections can be made in the usual OS/browser dependent ways. A trigger is a boolean expression written in Python, for example:

```
CRTS["First Detection params"]["magnitude"] < 17
and event["role"] != "test"
```

which is looking for bright transients detected by the CRTS stream (Catalina Real-Time Survey), which are not test events. In the forms for creating alerts, the parameter names are ‘live’, meaning that clicking on a parameter name inserts the relevant text in the trigger expression. When the expression is complete, click the Save button, which evaluates the expression and reports back any syntax errors.

Once the trigger expression is saved, another button (“See Events”) shows all the previous portfolios that satisfy it. At the top of the form is information about what action should take place. Currently the only action being publically offered is the email alert.

Authoring an Event Stream

Skyalert accepts reports of astronomical transients or information about them, using the VOEvent XML format. Injecting such data requires careful thought and some interaction with the Skyalert team. Before sending events, the meaning of the parameters should be defined. The authoring workflow is as follows:

1. Design the stream. Think carefully about what data you want to expose in the event. Some parameters are numbers and strings, and can be used to make decisions; some parameters are URLs that link to complex data objects. To help you design a stream:
 - Look at the existing event streams at skyalert.org (OGLE, Catalina, SWIFT, etc). At the bottom of each stream detail page is a sample event from that stream.
 - Look at the annotated sample VOEvent. All the gray text is boilerplate, the essentials are highlighted in yellow. Make sure you move the mouse over each highlight to see the explanation.
 - Work with the Skyalert Team (write to help@skyalert.org) to create a sample event that shows your parameters, each with description, unit, semantic descriptor (UCD), data type (int, float, string).
 - You can include complex data objects with a Param whose value is a URL link. In this case, make sure the UCD of that Param is `meta.code.url` so that Skyalert knows it is a URL.
2. Upload your stream. This will be done by Skyalert staff (help@skyalert.org), using the agreed sample event, which has all the metadata. You will be asked to check that your stream appears correctly in the web interface. If you are running your own Skyalert system, you can login as an account with staff status and do this yourself.
3. Test Events. Build code that can create events from your event stream. It is important that test events be labeled `role="test"` in the first element of the VOEvent, so they are not mixed with real astronomy.
4. Send events to Skyalert. You can send events, by arrangement, to a remote Skyalert server, including the one at skyalert.org. This is done through a popular Instant Messaging protocol (XMPP also known as Jabber [7]). There is software (Java or Perl) on the Skyalert site that will inject new events.

How Skyalert Works

Figure 5 shows the architecture of the system. Portfolios of VOEvents are stored in a database, with a web front end. The web application can also be used to set up streams for authoring, and to set up triggered subscriptions for real-time delivery. It is not only messaging that can happen as a result of a

trigger being satisfied, but any other kind of code; in fact there is an Annotator API which allows access to an existing portfolio, then the creation of a new event and injection into the system.

When an event arrives, either locally through the loader interface, or remotely through the XMPP interface, several things happen. The event is checked for correct XML syntax, and that it conforms to the VOEvent schema, then the parameters and other information is extracted and cached in a relational database. If the event has a citation to another event, and that other event is in the database, then the new event joins the portfolio of that other. Next, we take all the alerts that are in the system, and decide if the new event should cause any action.

Note that we have a *cyclic workflow* here: arrival of event A can cause an action to run which injects a new event B, and so there is a possibility of infinite runaway. In order to prevent this, event streams have an *ordinal* number (1, 2, 3, ...), and a rule requiring the triggering event (such as A) to have a lower ordinal than the consequent events (such as B). Thus we assign ordinal 1 to events that come *ab initio*, from the sky, and ordinal 2 to those that are created as a consequence of that, and so on, so no infinite cycles can occur.

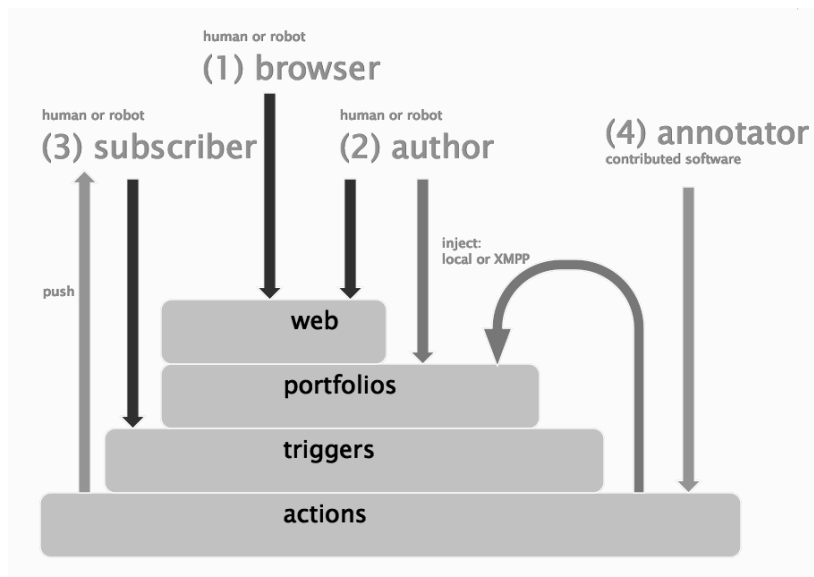


Figure 5: Interactions with Skyalert: (1) Browsing the events and streams, (2) Authors inject new events (3) Subscribers get immediate notification, (4) Annotation components can respond to new events with arbitrary actions, including injection of new events.

Conclusion

Skyalert is a web-based or downloadable application for handling events in general, rather than special code for each event. It can display the events in many ways, collect relevant data into portfolios, run machine learning code, send messages, and do all this in near real time.

References

- [1] Skyalert collects and distributes astronomical events in near-real time: <http://skyalert.org>
- [2] M. Graham and R. Williams, *Finding VOEvent Resources*, Section 4.2 in this book.
- [3] R. Seaman, R. Williams, A. Allan, et al, *Sky Event Reporting Metadata (VOEvent)*, IVOA Recommendation, <http://www.ivoa.net/Documents/latest/VOEvent.html>

- [4] Central Bureau for Astronomical Telegrams, <http://www.cfa.harvard.edu/iau/cbat.html>, and Minor Planet Center, <http://www.cfa.harvard.edu/iau/mpc.html>
- [5] Astronomer's Telegram, <http://astronomerstelegam.org>
- [6] S. Derriere et al, *An IVOA Standard for Unified Content Descriptors*, IVOA Recommendation, <http://www.ivoa.net/Documents/latest/UCD.html>
- [7] Extensible Messaging and Presence Protocol (XMPP), a set of open XML technologies for presence and real-time communication developed by the Jabber open-source community, <http://xmpp.org/>