Towards a Wide Area Alerting and Notification System

by Graeme McFerren and Derick Swanepoel, CSIR Meraka Insitute

Abstract

This paper describes the concept of a Wide Area Alerting System (WAAS), a flexible ICT platform for building alerting and notification applications. It is tailored to the generation and dissemination of complex alerts or notifications derived from geospatial-temporal data originating from heterogeneous sensors, models and/or monitoring processes. A WAAS implementation explicitly recognises and harnesses spatial location and time and is interesting for its ability to receive and combine observed events of multiple phenomena, over wide geographic areas, into highly targeted, rich notifications/alerts, delivered to potentially large numbers of users over convenient communication channels. It can be deployed in a variety of scenarios that require functionality for notifying users about phenomena or events occurring at geographical scales. Furthermore, events that could trigger alerts or notifications may be fast-moving or slow to unfold – even over a period of hours or days. This paper describes the functional and non-functional design goals, system architecture and software infrastructure, usage scenarios and implementation experiences to date of the WAAS.

Keywords

geo-spatial alerts, geo-spatial message-oriented middle-ware

Introduction

In the Global Earth Observation System of Systems 10 Year Implementation Plan of the Group on Earth Observations [1], deep emphasis is placed on harnessing earth observation (EO) data for societal benefit. One key aspect described is monitoring the state of the earth to support informed decision making. This is pertinent for many scenarios, but especially so for disaster management where the GEOSS 10-year-plan identifies that systems can be built to monitor, predict, assess risk of, provide early warnings about and allow response to various natural and human induced hazards.

Yet as Fox and Hendler [2] and Hanson et. al. [3] point out, science data (including EO data) are varied, interdependent, often complex and generally large in size, generated from an array of instruments, fields, and sources. A challenge to scientists and scientific institutions is how to expose, harness, package and disseminate such data for societal benefit.

The use of EO data in monitoring and notification scenarios for societal benefit are numerous. This paper describes the concept of a Wide Area Alerting System (WAAS), a flexible ICT platform for building alerting and notification applications. In light of the characteristics of EO data, a WAAS is tailored to the generation and dissemination of complex alerts or notifications derived from geospatial-temporal data originating from heterogeneous sensors, models and/or monitoring processes. A WAAS implementation explicitly recognises and harnesses spatial location and time and is interesting for its ability to receive and combine observed events of multiple phenomena, over wide geographic areas, into highly targeted, rich notifications/ alerts, delivered to potentially large numbers of users over convenient communication channels. It can be deployed in a variety of scenarios that require functionality for notifying users about phenomena or events occurring at geographical scales. Furthermore, events that could trigger alerts or notifications may be fast-moving or slow to unfold – even over a period of hours or days.

This paper shares and describes the functional and non-functional design goals, system architecture and software infrastructure, usage scenarios and implementation experiences to date of a WAAS developed by CSIR Meraka Institute.

Usage scenarios

We describe here three scenarios where a WAAS system could be used to synthesise disparate EO data and disseminate meaningful notifications to users. Each scenario is characterised by different phenomena and scales and types of monitoring.

- Environmental control officers need to be notified via e-mail about above-threshold values for several water quality parameters measured by sensors in rivers. Monitoring happens through a sensor platform deployed to monitor at a point scale. Users are interested in this point (and perhaps a small buffer around it) as a feature-of-interest; they may have multiple features-of-interest, possibly widely spatially distributed, each with its own sensor. Data volume is low, but sampling tempo is high.
- Power utility line managers and national control managers need to be alerted via SMS, e-mail and possibly push notifications to their mobile smart phones to vegetation fire events near components of their company countrywide infrastructure. Monitoring capability is provided by polar-orbiting and geo-stationary satellites.

- Data volumes are large, but sampling frequency is low to moderate. Users are interested in long spatial corridors as features-of-interest. Such linear features-of-interest cross a large spatial area that must be monitored.

- National park fire protection officers and section rangers need to be notified via e-mail and SMS to active wildfires in different sections of the park but primarily when modeled fire danger index is high. This scenario relies on similar monitoring requirements to the above scenario, plus the addition of modeled data. Features-of-interest are typically large areas.

System requirements

Functional

- The ability to monitor and notify users about phenomena/events at different scales (highly local, e.g. toxins at a gauge in a river, to widely regional e.g. fire in a national park).
- The ability to monitor and notify users about phenomena/events observed by different sensors.
- The ability to merge and filter event/phenomena data from different sources to create compound alerts.
- The ability to define and execute precise spatial, temporal and data attribute filters e.g. precise distance calculations, time-period calculations.
- The ability to "mute" alerts/notifications emanating from the same spatial location within a given time-period (useful to prevent notification repeats about the same event). This is important given the original assertion that sense needs to made of large amounts of data and information overload is to be avoided.
- The ability to disseminate notifications across multiple channels, e.g. SMS, e-mail, Instant Messaging, and the capability to include new channels (e.g. radio, push notifications to mobile devices)
- The ability to generate bulk alert/notification filters for an organisation.

Non-functional

- A WAAS should be easy to administer.
- Extensibility – a WAAS should be re-usable in different scenarios and it should be possible to add new functionality.
- Scalability – a WAAS should be able to handle large volumes of alert configurations. An organisation may, for example, need to monitor several phenomena in relation to thousands of assets or kilometres of infrastructure and deliver tailored alerts/notifications to hundreds of users.
- Standards-compliant – a WAAS should use inputs and generate outputs that are structured in internationally recognised data standards.

Implementation

There are, no doubt, several possible architectures that could be used to build a WAAS; likewise many technologies exist that could form the components of such a system. Here, we describe the architecture and implementation specifics of the CSIR Meraka WAAS capability as an exemplar of such a system. There exist three standards processes of the Open Geospatial Consortium Sensor Web Enablement Initiative that are concerned with alerting and notification, but we do not compare or discuss them within the scope of this paper. For completeness, these are:

- Sensor Alert Service [4]
- Web Notification Service [5]
- OGC Event Service [6]

The CSIR Meraka WAAS is a multi-tier system consisting of several loosely coupled subsystems, mostly written in the Python programming language:

Persistence subsystem

- *Description*

  Each WAAS instance defines, modifies and updates its configuration from a spatial database. Configurations consist of users, organisations, contact details, phenomena, features-of-interest, filters, time intervals and references, which, when placed together, form multiple "routes" which observations can travel through to reach users as actionable pieces of information. This persistence layer provides a canonical representation of the data model and instance data of the configured system. This allows for backup and recovery from failure, while providing the "description" of the system.

- *Implementation*

  The spatial database that provides the persistence layer is Postgresql [7] with the PostGIS [8] spatial extension.

Web frontend

- *Description*

  To meet with the non-functional requirement of ease of use, the Meraka WAAS provides a web-based interface to the configuration database. Users are guided in a "wizard-driven" procedure to add users, create alert conditions and bind them together.

- *Implementation*

  This is a Javascript/HTML-based client that communicates with the application middle-ware subsystem. Spatial components are provided by the OpenLayers [9] library.

Application middle-ware subsystem

- *Description*

  This is a middle-ware layer that is responsible for defining business logic, providing business logic components, managing the integrity of the system, providing access to the configuration data model and instance data and handling the HTTP request/response cycle necessary of a web-based system.

- *Implementation*

  The application middle-ware is primarily provided by the Django [10] web application framework, which includes geospatial functionality.

Notification and alerting engine.

- *Description*

  This is a message-oriented middle-ware subsystem. Observations are streamed in text format onto a message exchange. Observation messages are then consumed, filtered, processed and further exchanged through a number of component layers until alerts or notifications are distilled and submitted to a message gateway, such as an e-mail server or SMS gateway. Spatial, temporal and attribute filtering and processing of observations is undertaken by these components.

- *Implementation*

  The message-oriented middle-ware software is provided by RabbitMQ [11], an implementation of the Advanced Message Queue Protocol standard [12]. Spatial processing functionality is provided by components of Django, Rtree [13] and Pyproj [14].

Implementation conclusions

Most functional requirements are met through a combination of a) the flexibility of the the data model that allows complex and varied alert configurations to be constructed (encouraging reuse and re-purposing) and b) the atomic and extremely loosely coupled nature of the alerting engine, where no shared state is permitted. The latter factor provides for for scalability, since components conceptually can exist anywhere and communicate over networks via the simple mechanism of message passing. This allows the load of performing expensive spatio-temporal calculations to be distributed to take advantage of modern computer hardware, fast networks and access to data centres. This WAAS relies on information standards. Notably, it makes use of eXtensible Markup Language (XML) [15], JavaScript Object Notation (JSON) [16], the Advanced Message Queuing Protocol (AMQP) [12] and the OASIS Common Alert Protocol (CAP) [17].

Some metrics of a single, currently operational WAAS are listed below:

- Monitoring data from four satellite sensors to provide eight alert trigger combinations.
- Monitoring large areas in three countries.
- There are ~17 500 defined features-of-interest being monitored.
- Alerts can be disseminated over five channels (SMS, e-mail, instant messaging [Jabber, Google Talk], Twitter and via iOS push notifications).
- There are ~200 individual users in the system, from nine organisations.
- Resulting in ~92 000 alert conditions that could be triggered.

Conclusions

The CSIR Meraka WAAS demonstrates that it is possible to construct EO systems that meet a complex set of requirements, but that can still process large volumes of heterogeneous data into meaningful information, delivered to users over appropriate channels. Such a situation is made possible, in this case at least, by utilising a loosely coupled system architecture. This architecture allows components to be distributed as necessary to take advantage of available compute resources in order to provide scalability, flexibility and performance.

References

[1] GEO Implementation Plan Task Team, *Global Earth Observation System of Systems GEOSS 10-Year Implementation Plan Reference Document*, ESA Publications Division, Noordwijk, The Netherlands, February 2005.

[2] P. Fox and J. Hendler: "Changing the Equation on Scientific Data Visualization", Science, Vol. 331 no. 6018 pp. 705-708, 11 February 2011. DOI: 10.1126/science.1197654.

[3] B. Hanson, A. Sugden and B. Alberts: "Making Data Maximally Available", Science, Vol. 331 no. 6018 p. 649, 11 February 2011. DOI: 10.1126/science.1203354.

[4] I. Simonis (ed.): *OGC Sensor Alert Service Candidate Implementation Specification*, Open Geospatial Consortium Inc, 13 May 2006, http://portal.opengeospatial.org/files/?artifact_id=15588.

[5] I. Simonis, J. Echterhoff (eds.): *Draft OpenGIS Web Notification Service Implementation Specification*, Open Geospatial Consortium Inc, 18 November 2006, http://portal.opengeospatial.org/files/?artifact_id=18776.

[6] J. Echterhoff, T. Everding (eds.): *OGC Event Service - Review and Current State Discussion Paper*, Open Geospatial Consortium Inc, 23 November 2011, https://portal.opengeospatial.org/files/?artifact_id=45850.

[7] PostgreSQL Global Development Group, *PostgreSQL*, www.postgresql.org, retrieved 30 August 2012.

[8] PostGIS Development Community, *PostGIS*, http://postgis.refractions.net/, retrieved 30 August 2012.

[9] OpenLayers Dev Team, *OpenLayers*, http://www.openlayers.org, retrieved 30 August 2012.

[10] Django core team, *Django: A Web framework for the Python programming language*. Django Software Foundation, Lawrence, Kansas, U.S.A., http://www.djangoproject.com, retrieved 30 August 2012.

[11] The RabbitMQ team, *RabbitMQ*, http://www.rabbitmq.com, retrieved 30 August 2012.

[12] Advanced Message Queuing Protocol (AMQP) [http://www.amqp.org/resources/specifications.

[13] RTree Community, *Rtree: Spatial indexing for Python*, http://toblerity.github.com/rtree/, retrieved 30 August 2012.

[14] Pyproj Developers, *Pyproj*, http://code.google.com/p/pyproj/, retrieved 30 August 2012.

[15] XML Core Working Group, *eXtensible Markup Language (XML)*, http://www.w3.org/XML/Core/, retrieved 30 August 2012.

[16] *JavaScript Object Notation (JSON)* http://www.json.org/, retrieved 30 August 2012.

[17] OASIS Emergency Management Technical Committee, OASIS Common Alert Protocol (CAP) http://docs.oasis-open.org/emergency/cap/v1.2/CAP-v1.2.html, retrieved 30 August 2012.

Contact Graeme McFerren, CSIR Meraka Institute, Tel 012 841-3405, gmcferren@csir.co.za