# Building a Heterogeneous Sensor Infrastructure using a schema-less approach.

Neil Harris, Phil James

School of Civil Engineering and Geosciences, Newcastle University, NE1 7RU

Neil.Harris1@newcastle.ac.uk

Philip.James@newcastle.ac.uk

KEYWORDS: Heterogeneous Sensors, Open source, NOSQL

## Background and motivation

One of the aims of is project was to define a single flexible data structure to manage all the cross-sectorial heterogeneous data recorded. This is challenging as due to the diverse nature of the data defining one ubiquitous data schema to suit all scenarios without numerous redundancies is practically impossible.

Urban areas are complex systems, comprising many interacting infrastructure sectors. Understanding these inter-relationships is essential to sustainable urban and infrastructure development. Research focused on single sectors, or over limited timescales, will inevitably fail to capture these interdependencies and dynamics. Consequently this project is carrying out work to investigate these sectorial relationships. The initial facility monitors a range of infrastructures and sectors such as water, earthworks, transport, climate, waste. The concept of the system is displayed in figure 1.

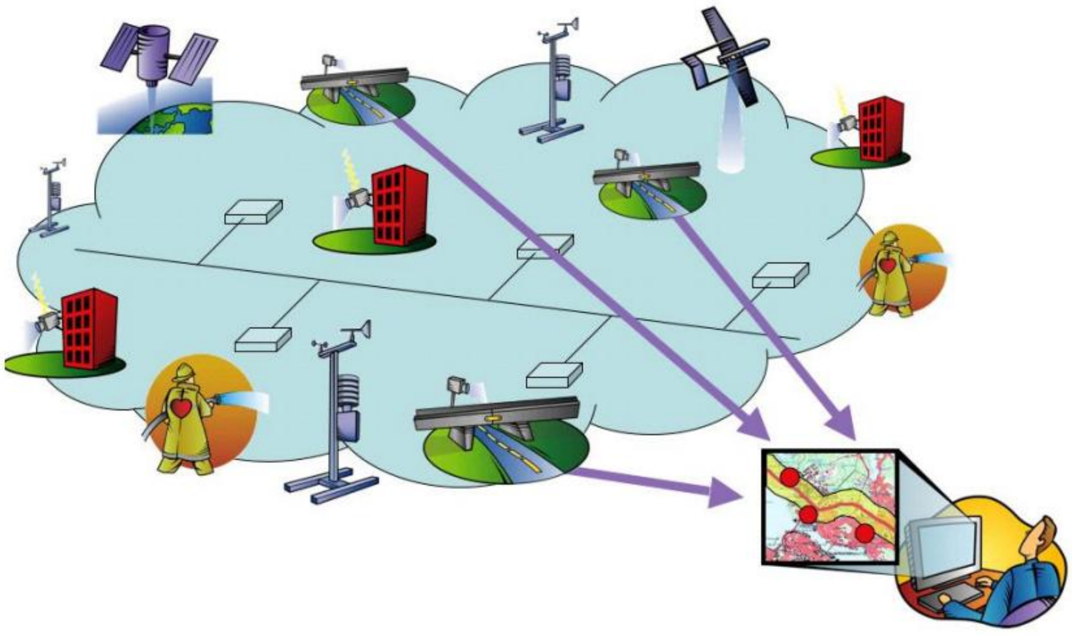


**Figure 1: Integrated spatial data system**
**(Source: http://portal.opengeospatial.org/files/?artifact_id=25562)**

In most modelling and monitoring systems where data is assembled of many heterogeneous cross sectorial data sources, the data is often provided by different through a mixed bag of real-time dynamic service systems from Geo-Web Services to aggregated files and simple data streams. This isolation of the data means any unified analysis on more than one data stream is impossible. However, making use of all available data sources is a prerequisite for holistic and successful monitoring system that allows broad decision support in an urban context. This applies to emergency situations as well as to the continuous monitoring of urban parameters. Fleischer *et al.* (2010) describes the need to integrate several different cross sectorial data sources in order to gain a further understanding for, in this case, prediction of a tsunami wave .As each data source is not only diverse in the recordings being made but also in the frequency and units in which it makes it reading, in order to integrate these in a cohesive whole within a flexible data structure then a flexible architecture is needed. As for example temperature maybe recording in Fahrenheit, Celsius and Kelvin but to reduce any delay in storing the record it is more efficient to do the conversion when the data is requested rather than when it is inserted. It also allows new data to be assimilated effortlessly into the architecture without any prior knowledge

The multiple data sources will also allow the creation of real-time data visualisations and well as feed directly into simulation models and qualitative interpretation to better understand cities, infrastructure systems and urban activity to aid policy making and well as potential disaster response.

In order to extract the appropriate data for these models and visualisations numerous queries will have to take place on the data. Effective query management in databases is important in order to enhance the speed of sensor interaction with networks that contain enormous sensor nodes; this can be achieved by designing and implementing a middleware layer that not only ensures effective queries but can also manages controls the data which is stored, i.e. flagging extreme values, eliminating ambiguities such as “temperature” and “Temperature”.

## Database Overview

Traditional database architectures create tables for each different “set” of data coming in (or create large data tables with lots of redundancy). The approach adopted here is to use a key-value pairs of semi-structured data. Key value stores allow the application developer to store schema-less data. Seeger (2009) outlines this key value approach to storing data, describing it as data usually consisting of a string which represents the key and the actual data which is considered to be the value in the "key - value" relationship. The data itself is usually some kind of primitive of the programming language (a string, an integer, an array) or an object that is being marshalled by the programming languages bindings to the key value store. This replaces the need for fixed data model and makes the requirement for properly formatted data less strict.

Nakamura *et al.* (2011) designed and implemented a new database approach for storage and delivery of sensor data called uTupleSpace. uTupleSpace is a schema-less style data store consisting of key/value pairs. It was found by using this schema-less approach they were able to meet increases in variety and quantity of sensor data.

As well as the ability to deal with variations in heterogeneous sensor data this schema-less approach is designed to allow for the fluid nature of a system designed to organically grow as more sensors come online. This is due to the fact that unlike in a strict schema approach there is no initial rigid encoding and therefore a priori knowledge of entity associations is not required. This advantage was noted in Pulsifer *et al.* (2010) whereby a database driven web application for sea ice monitoring was developed which was able to scale up dynamically as more data was integrated.

Openstreetmap also adopt a schema-less approach to storing their data, referred by them as tags (OSM, 2013b). In order to store this information they make use of the hstore (PostgreSQL, 2011) extension in postgres (OSM, 2013a). This module implements the hstore data type for storing sets of key value pairs within a single PostgreSQL column. This allows a No-SQL/schema-less style storage method to sit within PostgreSQL database. Consequently Openstreetmap has a flexible data store which allows attributes to be added on the fly without schema changes.

Further implementations of key-value pairs for sensor data include Brunette *et al.* (2012), Thantriwatte and Keppetiyagama (2011) and Amirian *et al.* (2013) who found that the less strict approach was able to deal with the high volume, high frequency of change (in both data content and data structure) and variety of structures, that conventional data storage systems cannot provide.

The goal of this system is to enable seamless access to sensor (in the widest sense) data through attribute, temporal and spatial queries to provide answers to questions such as what effect is the urban heat island having? At which locations is the air quality poorest? Whether there is a direct correlation between area of poor air quality and noise levels? Therefore the system needed to store information about where sensors are located (or the position of a reading in the case of a mobile sensor). In order to meet the dual requirements of a schema less approach to sensor data and the ability to analyse and query the data spatially a system was implemented in the PostgreSQL relational database using the PostGIS spatial extension using the HSTORE data type described above.

The database structure is straightforward; sensors are stored in one table, and sensor data in another. These both consisted of simply 2 columns, a row id and a hstore row, Examples of both are shown in tables 1 and 2. As hstore only stores strings the POSTGIS geometry is created on insert, via the middleware library, and then stored as a string as part of the hstore. This allows POSTGIS functions to be utilised on the sensors if necessary.

These tables show the key values pairs of both the sensors and the sensor data tables. The sensor_id value is used as the foreign key to link the two tables. The flexible approach of the schema also allows unlimited tags to be added to the readings, for example you can see that row 595770 has been flagged as suspect, this is due to the negative CO reading.

**Table 1: Sensors table**

| Sensors | |
|---|---|
| 259 | "geom"=>"*0101…A4B40*", "sensor_id"=>"ITYNEAND10", "type"=>"Weather", "active"=>"True", "source"=>"Wunderground", "auth_needed"=>"True" |
| 258 | "geom"=>"*0101…D4B40*", "sensor_id"=>"123", "type"=>"Air Quality", "active"=>"True", "source"=>"emotes", "auth_needed"=>"True" |

**Table 2: Sensor_data table**

| Sensor_data | |
|---|---|
| 595770 | "flag"=>"suspect", "theme"=>"Air Quality", "units"=>"ppm", "value"=>"-2.02284329081", "reading"=>"CO", "sensor_id"=>"181", "timestamp"=>"2014-01-06 14:50:00" |
| 595769 | "theme"=>"Environment", "units"=>"db", "value"=>"74.9973862442", "reading"=>"Sound", "sensor_id"=>"181", "timestamp"=>"2014-01-06 14:50:00" |

This approach also allows simple integration of metadata with sensors being linked to relevant documents stored in metadata tables describing relevant second order information about the sensor such as their specifications. Data entries can also been themed dynamically, for example sensor 181 is producing both Environmental and Air Quality theme data entries. Theme keywords are stored in a separate data table.

The challenge with this limitless tag system is to ensure that different sensor networks recording the same variable contain the same tag to allow integration i.e. 2 sensors may record noise level but one may refer to it as sound but the other as noise. This is achieved by using a simple metadata table, also using hstore, to store the tags and the agreed upon possible values.

## Conceptual Overview

The work carried out in this project (*Establishing a long term Urban Research Facility*, 2012) makes use of long-term datasets generated by using multiple methodologies that observes phenomena at the individual, building, campus and through to city-wide and regional scales. This involved integrating a number of new and existing sensors into one data system. As well as this existing third party data sources were utilized such as Split Cycle Offset Optimisation Technique (SCOOT) which monitor traffic flow at junctions and are used to control traffic light systems. In addition the system integrates social media “sensors” into the same framework (currently only Twitter) utilising geotags in messages and the ability to store relevant data by searching on hashtags.

The Open Geospatial Consortium (OGC) has carried out work to create a fully integrated sensing system by developing a set of standard called Sensor Web Enablement (SWE) standards (Botts *et al.*, 2008). These enable developers to make all types of sensors, transducers and sensor data repositories discoverable, accessible and useable via the Web. However as Funk *et al.* (2011) points on due to the large communication overhead that comes along with the XML descriptions of the SWE protocols this was not deemed a sensible approach. Instead a new middleware layer has been developed. SWE extensions to the middleware layer could be developed at a future date.

A middleware layer has been developed to handle input and output to the database stores, adding additional metadata and descriptive information automatically. Further work is to be carried out on this middleware layer to create functionality to perform tasks such as generalising the data either temporally or spatially or conversion between measurement units. This middleware layer means that a user is able to extract all temperature values without needing to know that these values could potentially be from a variety of different sensor platforms. It also allow the system to grow organically as new sensor technology comes online or is made available

### *Middleware and Outputs*

To facilitate data I/O and to provide an extensible API for interaction with the data streams for visualisation e.g. on the web and analysis in third party software a python based middleware layer has been developed. The concept of this middleware layer is depicted in figure 2

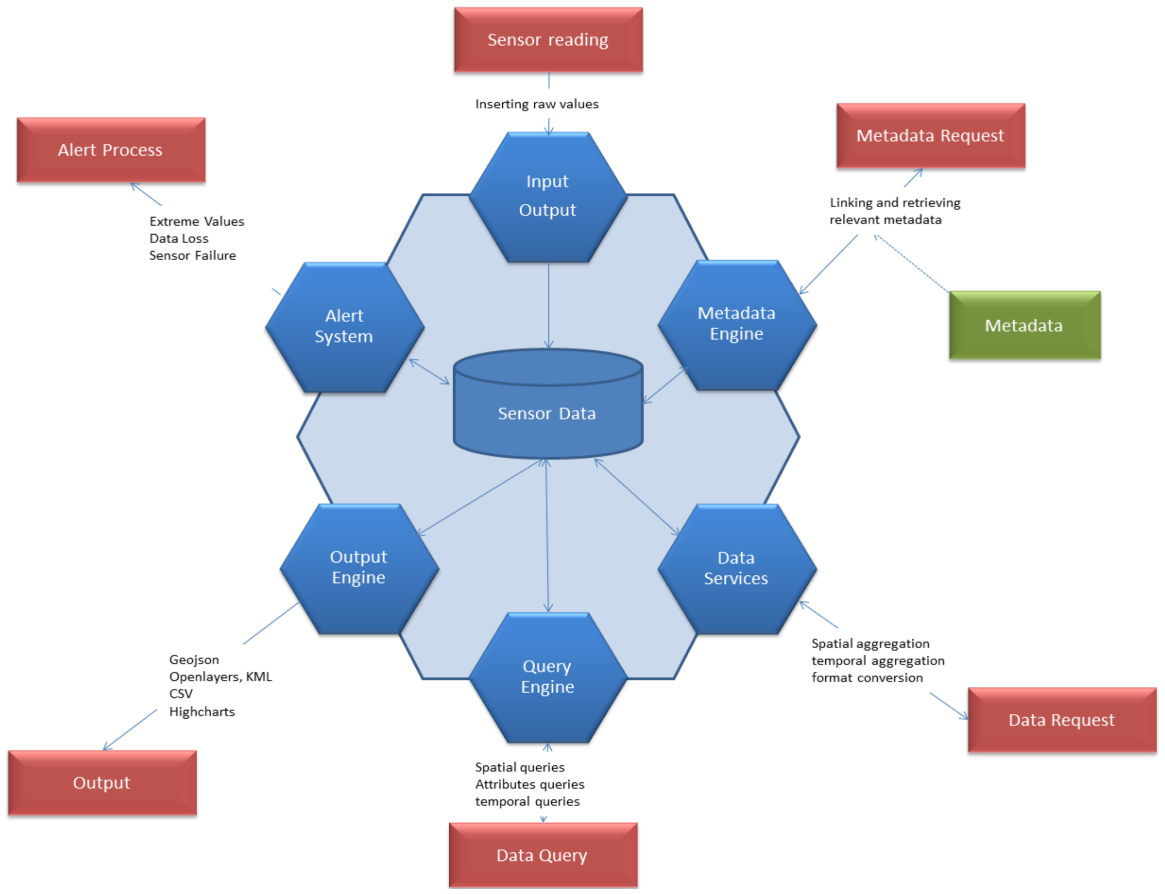


**Figure 2: Middleware Conceptual Model**

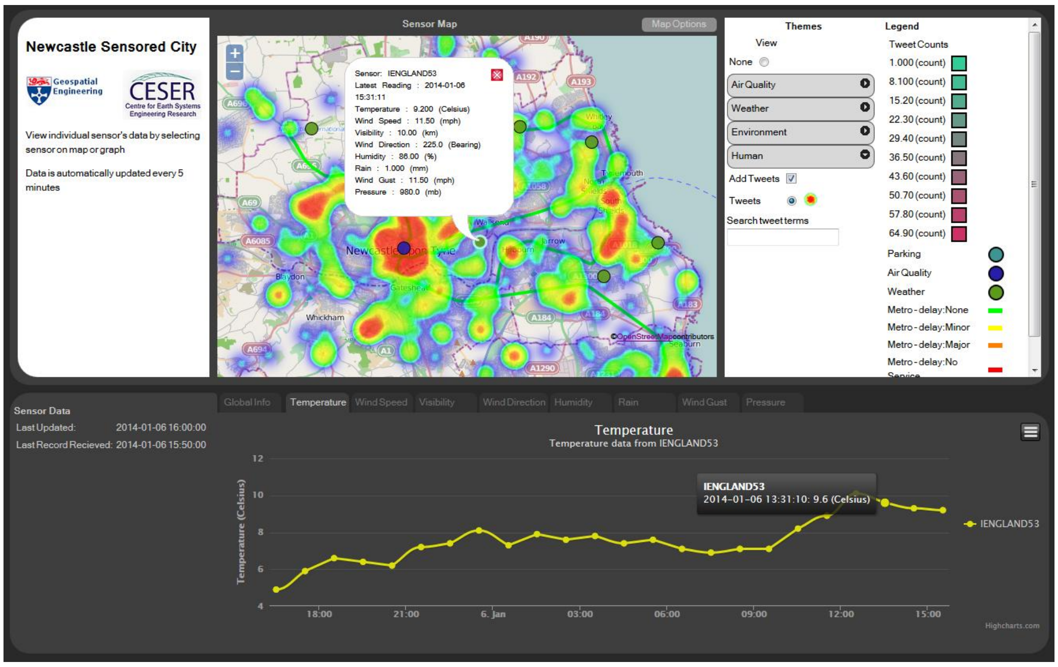
As well as assisting the data interaction current functionality also includes:

- Fetch sensor geometry in various formats such as geojson (example of this shown in Figure 3)
- Fetch sensors in bounding box
- Fetch the latest reading for an individual sensor
- Fetch data for sensor between 2 times
- Format sensor data to be compatible with various application, such as HighCharts (*Highcharts 3.0 released*, 2013), a JavaScript graphing library.

```
import nclsensorweb
import datetime
sensorweb = nclsensorweb.SensorWeb(database_host,database,user,passwd)
sensors =   sensorweb.Sensors.get_all()
end_time  = datetime.datetime.now()
start_time = end_time - datetime.timedelta(hours=6)
sensorjson =[ ]
for sensor in sensors:
        sensorjson.append(sensor.geojson())
```

**Figure 3: Example middleware use**

These functions are to aid the creation of visualisations and feeding the data into models in order to carry out further research. Using this python middleware layer in conjunction with Django (*The Web framework for perfectionists with deadlines | Django*, 2013), a live dashboard site was created display the data in real-time, figure 4. This was able to display the live data on an OpenLayers (*What is OpenLayers?*, 2008) map and make uses of Highcharts (*Highcharts 3.0 released*, 2013) to display the data in various charts.


Newcastle Sensored City

View individual sensor's data by selecting sensor on map or graph

Data is automatically updated every 5 minutes

Sensor: IENGLAND53
Latest Reading : 2014-01-06 15:31:11
Temperature : 9.200 (Celsius)
Wind Speed : 11.50 (mph)
Visibility : 10.00 (km)
Wind Direction : 225.0 (Bearing)
Humidity : 86.00 (%)
Rain : 1.000 (mm)
Wind Gust : 11.50 (mph)
Pressure : 980.0 (mb)

Sensor Data
Last Updated: 2014-01-06 16:00:00
Last Record Recieved: 2014-01-06 15:50:00

IENGLAND53
2014-01-06 13:31:10: 9.6 (Celsius)

**Figure 4: example web visualisation**

## Conclusion

The NOSQL schema-less approach for storing heterogeneous sensor networks has worked well allowing effective and efficient data management and exploration. The hstore module proved to work well as a key-value data type. It also maps neatly to a python array allowing the easy creation of the python based middleware layer. This has successfully been able to manage the data storage as well as feed data into a number of APIs needed for the real-time visualisation.

## Future Work

This work is ongoing with 300+ additional sensors being rolled out in the upcoming months. In addition the platform underpins the new £50 million Science Central development in Newcastle which will allow for a very dense network of internal and external environmental sensors and monitoring platforms linking data to policy makers and scientists through visualisations, a data API and the provision of a Decision Theatre.

## References

AMIRIAN, P., WINSTANLEY, A. AND BASIRI, A. (2013) 'NoSQL storage and management of geospatial data with emphasis on serving geospatial data using standard geospatial web services'.

BOTTS, M., PERCIVALL, G., REED, C. AND DAVIDSON, J. (2008) 'OGC® sensor web enablement: Overview and high level architecture', in *GeoSensor networks*. Springer, pp. 175-190.

BRUNETTE, W., SODT, R., CHAUDHRI, R., GOEL, M., FALCONE, M., VANORDEN, J. AND BORRIELLO, G. (2012) 'Open data kit sensors: a sensor integration framework for android at the application-level', *MobiSys*. ACM. Available at: http://dblp.uni-trier.de/db/conf/mobisys/mobisys2012.html#BrunetteSCGFVB12.

*Establishing a long term Urban Research Facility* (2012). Available at: http://www.ncl.ac.uk/ceg/about/news/item/establishing-a-long-term-urban-research-facility (Accessed: 08/01/2014).

FLEISCHER, J., HÄNER, R., HERRNKIND, S., KLOTH, A., KRIEGEL, U., SCHWARTING, H. AND WÄCHTER, J. (2010) 'An integration platform for heterogeneous sensor systems in GITEWS–Tsunami Service Bus', *Natural Hazards and Earth System Science*, 10(6), pp. 1239-1252.

FUNK, A., BUSEMANN, C., KUKA, C., BOLL, S. AND NICKLAS, D. (2011) 'Open Sensor Platforms: The Sensor Web Enablement Framework and Beyond', *MMS*. pp. 39-52.

*Highcharts 3.0 released* (2013). Available at: http://www.highcharts.com/component/content/article/2-news/54-highcharts-3-0-released (Accessed: 08/01/2014).

NAKAMURA, T., KASHIWAGI, K., ARAKAWA, Y. AND NAKAMURA, M. (2011) *Applications and the Internet (SAINT), 2011 IEEE/IPSJ 11th International Symposium on*. 18-21 July 2011.

OSM (2013a) *Databases and data access APIs*. Available at: http://wiki.openstreetmap.org/wiki/Databases_and_data_access_APIs (Accessed: 08/01/2014).

OSM (2013b) *Map Features*. Available at: http://wiki.openstreetmap.org/wiki/Map_Features (Accessed: 08/01/2014).

PostgreSQL (2011) *hstore*. Available at: http://www.postgresql.org/docs/9.1/static/hstore.html (Accessed: 08/01/2014).

PULSIFER, P.L., KAUFMAN, M., YOUNG, D., COLLINS, J.A., EICKEN, H. AND GEARHEARD, S. (2010) 'Using Schema-less Database Technology to Develop a Web Application for Sea Ice Monitoring', *AGU Fall Meeting Abstracts*, 31, p. 1292.

SEEGER, M. (2009) 'Key-Value stores:a practical overview'.

THANTRIWATTE, T.A.M.C. AND KEPPETIYAGAMA, C.I. (2011) *Advances in ICT for Emerging Regions (ICTer), 2011 International Conference on*. 1-2 Sept. 2011.

*The Web framework for perfectionists with deadlines | Django* (2013). Available at: https://www.djangoproject.com/.

*What is OpenLayers?* (2008). Available at: http://docs.openlayers.org/.