Short paper: Enabling Lightweight Semantic Sensor Networks on Android Devices*

Mathieu d'Aquin, Andriy Nikolov, Enrico Motta

Knowledge Media Institute, The Open University, Milton Keynes, UK
{m.daquin, a.nikolov, e.motta}@open.ac.uk

Abstract. In this short paper, we present an architecture to deploy lightweight Semantic Sensor Networks easily based on widely available Android Devices. This approach essentially relies on deploying a SPARQL endpoint on the device, and federating queries to multiple devices to build Semantic Sensor Network applications.

1 Introduction

Research into semantic sensor networks has been focusing on the treatment and processing of data aggregated from large networks of sensors, often based on specialised equipments geographically distributed in large areas. [1] discusses a number of challenges related to Semantic Sensor Networks in such scenarios. The challenge we are particularly interested in relates to the ability for "rapid development of applications" that make use of Semantic Sensor Network. We especially look at applications in scenarios where it is needed to set-up networks of simple sensors quickly and easily (e.g., school projects, small experiments).

In recent years, the Android platform[1] became a de-facto standard for different types of mobile devices from several manufacturers. These devices possess several types of embedded sensors such as a camera, an accelerometer, a GPS sensor and a microphone. On the other hand, as shown by our previous work [2], the computational power of these devices already allows efficient processing of small to medium volumes of semantic data.

In this short paper, we describe a lightweight architecture to create small scale sensor networks based on Android devices, and an application that makes use of such a network of Android devices/sensors. To realise this, we adapt a triple store to be deployed on an Android device, which provides a shared repository populated through sensor-aware applications on the device. The information gathered in this shared repository is exposed through an externally accessible SPARQL endpoint, making it possible to build applications exploiting data collected from a network of devices, through query federation.

2 Overview

The idea on which this paper is based is very simple: exposing the sensors attached to an Android Device through a SPARQL endpoint and using SPARQL query federation so that the information gathered through these sensors can be exploited as the product of a Semantic Sensor Network (see Figure 1).

* Part of this research has been funded under the EC 7th Framework Programme, in the context of the SmartProducts project (231204).

[1] http://www.android.com/

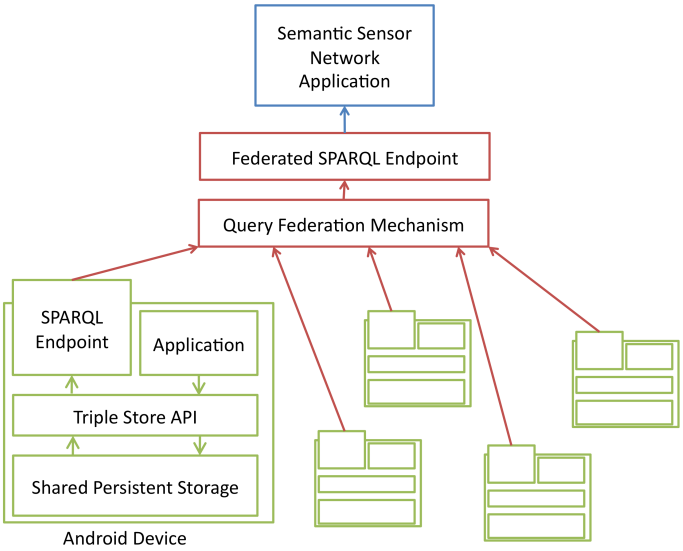

Fig. 1. Overview of the approach to create Semantic Sensor Networks out of Android devices.

In this approach, the use of the Semantic Sensor Network ontology is essential, as it allows to integrate the information coming from various devices and sensors. The idea here is that applications developed for the Android platform directly produce information using this ontology from sensors attached to the device. In this way, no post-processing is need for external applications to employ this information directly out of the SPARQL endpoint.

3 Implementation

At the core of our approach is the deployment of a triple store on the Android device, which is shared by the mobile applications populating it, using the Semantic Sensor Network ontology, and by the SPARQL endpoint deployed in the device. As discussed, in [2], Sesame[2] is, amongst the available options, the one that best fits an environment where only limited resources are available. We therefore adapted Sesame so that it can be deployable as an Android library. The Android environment is based on a specialised Java Virtual Machine, and Sesame being developed entirely in Java, most components of Sesame did not require any adaptation. Access to files is however different on the Android platform than it is on a usual computer. We therefore extended Sesame so that it provides a persistent RDF store using the shared, external storage available on most Android Devices (in SmartPhones, it corresponds to the SD card). In other terms, a shared persistent repository is installed on the Android device that is accessible, and can be populated, by applications accessing the device's sensors.

[2] http://www.openrdf.org/

The other element to be included on the Android device is a Web interface giving access, through the SPARQL protocol, to the content of the shared triple store. One of the difficulties here is to deploy a Web server on the such as device, being accessible externally. Luckily, the popular Web application server Jetty[3] has been ported to work on the Android Platform in iJetty[4], providing both a Web server and a servlet environment. We therefore implemented (a simplified version of) a SPARQL endpoint as a web application relying on our Android-adapted version of the Sesame API.

Finally, a mechanism is needed for the federation of SPARQL queries over multiple Android devices. In the cases where the data comes only from isolated and independent sensors, this federation mechanism can be very simple, as it only requires concatenating the results obtained from queries to different devices. In more complex scenarios where information from different sensors can be linked (such as discussed later), a more sophisticated mechanism is needed. We rely here on our own implementation of a distributed SPARQL query endpoint based on federating queries to multiple other SPARQL endpoints [3].

4 Example Application

To illustrate the benefits of the architecture we are proposing for lightweight Semantic Sensor Networks with Android, we developed a simple application used to collaboratively "map" a geographical area using pictures (for example, to give an idea of the views at certain points of a path, in an area that Google

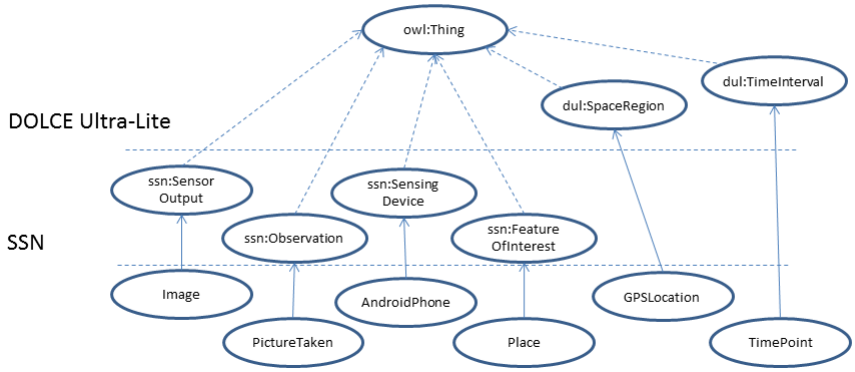

Fig. 2. Extension of the SSN ontology used in our example application.

We develop an application that can take pictures and represent the information about the picture and its location as an Observation using the extension of the Semantic Sensor Network ontology shown in Figure 2. This application records the path of the picture on the device, the location of the device at the time of taking it, as well as the time and the identifier of the device representing the sensor used to make the observation. We then used this application on several

[3] http://jetty.codehaus.org/jetty/
[4] http://code.google.com/p/i-jetty/

different SmartPhones. Using the simple SPARQL federation method described above, we implemented a Javascript application that displays the pictures taken from this network of phones on a map of the covered area (see Figure 3).

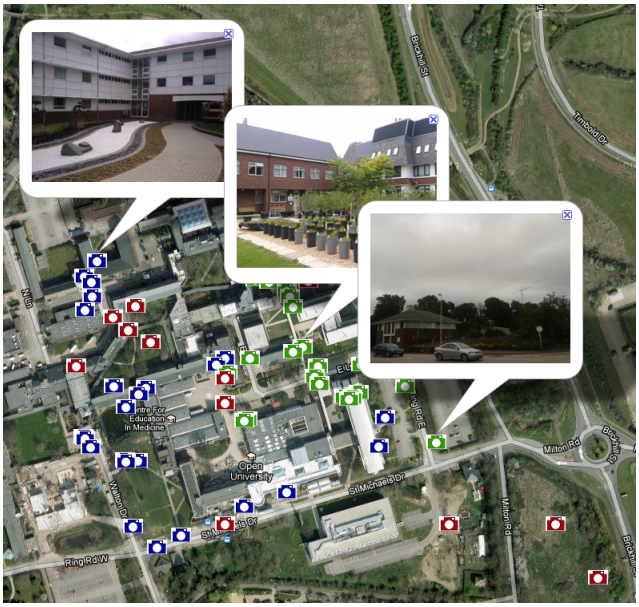

Fig. 3. Screenshot of the application mapping pictures taken from a network of Android phones using federated SPARQL queries.

5 Related Work

An architecture supporting a semantic sensor network commonly includes three layers: 1- Data layer: sensors producing observations; 2- Middleware layer: services responsible for sensor discovery and semantic data integration; 3- Service layer: services providing integrated data to the end-user applications. Implementing such an infrastructure for a specific use case requires dealing with five main challenges [1]:

- Choosing an appropriate *abstraction level* for sensor data representation.
- Adequate characterisation and management of the *quality of service*, including desired levels of data availability, completeness, and response time.
- Realizing *integration and fusion of data* from heterogeneous sensing devices.
- *Identification and location* of relevant sensor data sources.
- Supporting *rapid development of applications* handling integrated sensor data.

Depending on the requirements of the targeted use case scenarios, these issues can be addressed to a different extent, and a trade-off between them can be required. For example, ensuring high quality of service can make the architecture more complex and expensive and complicates deployment and application development.

Among existing solutions, the SemsorGrid4Env project[5] focuses on building large-scale semantic sensor networks for environmental management, in particular, for such critical scenarios as fire prevention and flood control. The proposed solution involves a multi-tier service-oriented architecture [4], which utilises a range of web services to integrate streaming data coming from sensor networks with data stored in static repositories. The nature of the scenarios requires the architecture to pay particular attention to such problems as quality of service, information latency, and security. The Sense2Web approach [5] uses the linked data principles to make sensor data publicly available via the Web. It allows the user to publish semantic descriptions of sensors and link them to other linked data resources (such as location URIs from DBPedia). The LinkedSensorData repository [6] implements a wrapper over meteorological data provided by sensors in the O&M XML format, combines the data in a single repository and makes it accessible with SPARQL queries. Similarly, the SensorMashup platform [7] assumes the existence of sensors producing streaming data and provides a semantic infrastructure for composing mashups over these data.

These architectures primarily concentrate on large-scale geographically distributed networks for industrial scenarios. We, on the other hand, look at more ad-hoc scenarios in which the deployment of a dedicated sensor network can be too complex and expensive. Other systems have been developed that provide RDF-based storing and querying of information on Android devices [8, 9]. They focus on the storage, querying and manipulation of RDF on the device, while we focus on how exposing information collected on the device through SPARQL can enable lightweight networks of devices as sensors.

6 Discussion

The architecture presented here is simple and lightweight by nature. It has however a number of advantages that favour the rapid development of applications relying on Semantic Sensor Networks, where the sensors are provided by one of the most available and affordable platforms. We presented an application that shows how somebody can easily set up a set of devices that act as a sensor network, providing information through a SPARQL interface for the application to integrate and use.

We can imagine many possible extensions to this application and other applications where a similar set-up could be considered, focusing on different types of sensor. For example, we could use the same application in a school trip dedicated to bird-watching. Groups of pupils would be given an Android phone where they could record with a picture seeing a particular type of bird. In this case, we could extend the application to also use the microphone of the phone to record the sound of the birds. Thinking about other sensors that could be used on an Android device, we could set-up a network of devices at different fixed positions in a building to record the vibrations on the floor of the building during the day, together with the sound intensity (for example, to find out about the impact of some building work). We could use a more complete combination of

[5] http://www.semsorgrid4env.eu/

sensors (camera, accelerometer, microphone) to check how busy different areas of a museum are during an opening day, and derive from this information the flow of visitors going from exhibitions to exhibitions.

In all these examples, the common aspect is that the sensor infrastructure is simple and lightweight. There is a need for an architecture that can be easily set-up, is highly re-usable, and is affordable. Also, the need for easy ways to semantically integrate data is very clear in such scenarios. In our application, we could annotate the pictures with the buildings they represent, and similarly, with the species of birds considered in the school trip scenario. Both the vibration and museum scenarios should be connected to a semantic model of the buildings being considered and to events happening during the day.

The approach using query federation makes such integrations easier. Indeed, an annotation service could be set up that provide an interface for users to annotate the pictures taken from the different phones with buildings or bird species. This service would not need to aggregate all the data in one place, but could simply use query federation to access information about the pictures and expose the annotations through its own SPARQL endpoint, therefore enriching the network with more information. Similarly, a SPARQL endpoint can be set-up that delivers data regarding the buildings and events in which a network is set-up. An interesting element is that such a SPARQL federation approach makes it easier to realise hierarchical networks: networks made of sub-networks. The use of the Semantic Sensor Ontology and of a coherent URI scheme would allow in this case to put together sensor networks that are heterogenous in nature and infrastructure.

References

1. Corcho, O., Garcia-Castro, R.: Five challenges for the semantic sensor web. Semantic Web Journal **1**(1) (2010)
2. d'Aquin, M., Nikolov, A., Motta, E.: How much semantic data on small devices? In: EKAW. (2010)
3. Miche, M., Erlenbusch, V., Allocca, C., Nikolov, A., Mascolo, J.E., Golenzer, J.: Final concept for storing, distributing, and maintaining proactive knowledge securely. Technical Report D4.1.3, SmartProducts Consortium (2011)
4. Gray, A.J.G., Garca-Castro, R., Kyzirakos, K., Karpathiotakis, M., Calbimonte, J.P., Page, K., Sadler, J., Frazer, A., Galpin, I., Fernandes, A., Paton, N., Corcho, O., Koubarakis, M., De Roure, D., Martinez, K., Gmez-Prez, A.: A semantically enabled service architecture for mashups over streaming and stored data. In: ESWC. (2011)
5. Barnaghi, P., Presser, M., Moessner, K.: Publishing linked sensor data. In: Semantic Sensor Networks, SSN. (2010)
6. Patni, H., Henson, C., Sheth, A.: Linked sensor data. In: Collaborative Technologies and Systems, CTS. (2010)
7. Le-Phuoc, D., Hauswirth, M.: Linked open data in sensor data mashups. In: Semantic Sensor Networks, SSN. (2009)
8. David, J., Euzenat, J.: Linked data from your pocket: The Android RDFContentProvider. In: Demo, ISWC. (2010)
9. Le-Phuoc, D., Parreira, J.X., Reynolds, V., Hauswirth, M.: Rdf on the go: An rdf storage and query processor for mobile devices. In: Demo, ISWC. (2010)