Mashups for Community Aware Sensor Processing with SCAMPI

Christian Kuka, Claas Busemann, Daniela Nicklas, Susanne Boll

OFFIS - Institute for Information Technology
Oldenburg, Germany
kuka|busemann@offis.de
http://www.offis.de

Carl von Ossietzky Universität
Oldenburg, Germany
daniela.nicklas|susanne.boll@informatik.uni-oldenburg.de

Abstract: Building up community web sites for sensor-based application like weather portals, traffic information, or stock data are problematic when the members of the community have little knowledge about data processing and visualization. Available middlewares and platforms do offer sufficient services to help users to fulfill important tasks on the way to their own community platform like filtering, aggregating, or visualizing. However, none of them fulfills all tasks or allows further usage of the data in other applications. In this paper, we show how the SCAMPI middleware allows the integration of sensor processing and visualization through mashups into existing web applications and the development of new web applications through a unified access to processed sensor data.

1 Introduction

More and more sensors are available in our world today. Several sensors can be found in Internet-capable devices like smart phones, weather stations, home appliances, etc.. However, even if a huge number of sensors are available, it is not easy to integrate live data into a web application. Let us assume a hobby meteorologists community wants to build up a small web application to present current weather forecast (Fig. 1). In a first step, the web application should fetch measurements from different sources like weather stations, weather services, and smart phones. In a next step, it should process the measurements based on individual interests of the members of the community. After processing the sensor data, the web application should visualize the continuous results of the processing, so that in order that other members or visitors of the site can see it. This results in the following requirements: Access to multiple heterogeneous sensor sources, process of different measurements based on individual needs and continuous visualization of the processing results. However, the members of the community may have little knowledge of implementation languages and communication protocols. This example illustrate the gap between

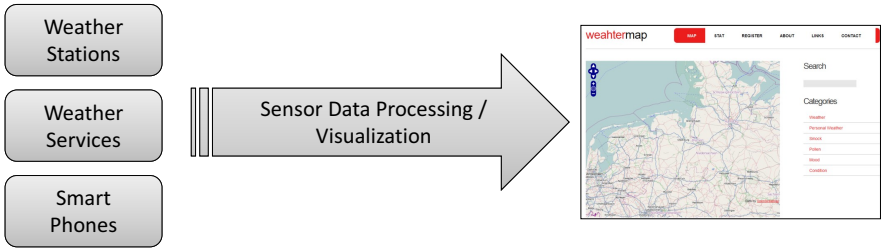

Figure 1: Weather Community Application

todays sensor systems and the communities who want to use these sensor systems in their applications. The gap originates in the missing opportunity to process, access and visualize continuously measured sensor data in the web.

In this paper, we present SCAMPI, a configuration and aggregation middleware for multi platform interchange. The SCAMPI middleware allows a unified access and management of sensors and the processing of sensor data. The internal processing unit allows members of a community to define individual processing of multiple physical and virtual sensor sources. Therefor it uses the Datastream Management Framework Odysseus [JG08]. Through a graphical web interface communities can manage different sensor sources and create their own virtual sensors to build up their community applications. Furthermore, SCAMPI provides multiple widgets for visualization of the sensor data to embed them into web applications and allows sharing of sensor data between communities and companies. The rest of the paper is organized as follows: Section 2 compares current available middlewares for sensor processing and usage. Section 3 gives an overview of the SCAMPI architecture and describes its layers. Section 4 describes the processing and further usage of sensor data in community applications. Finally, Section 5 summarizes this paper.

2 Related Work

Many middlewares are integrating sensor data processing and access to sensor data. Examples are the Sensor.Network [GPU10] application, the SenseWeb [SNL+06] platform, the GSN [AHS07] middleware, the SStreamWare [GRL+08] middleware, Oracle's SensorEdge [Ora] platform, the FireEagle [Yah07] platform, or middlewares implementing the SWE [Ope] standard. All of these middlewares share one or more common limitation. While SenseWeb only provides a predefined processing and only supports the visualization on a map, FireEagle, SensorEdge, Sensor.Network and SWE implementations only provide a web service to get access to the sensor data over the internet. However, none of these middlewares provide an individual processing except from filtering. The GSN platform and the SStreamWare middleware offer an individual processing but they do not allow members of communities to define the processing or visualize the process-

ing results depending on their preferences using web applications. None of the mentioned middlewares cover all requirements for our hobby meteorologists community to individual process, access and visualize sensor data on the web.

3 Architecture

In this section we present a closer look to the SCAMPI architecture (Fig. 2) and show the different responsibilities of each part. The SCAMPI architecture consists of the following five components:

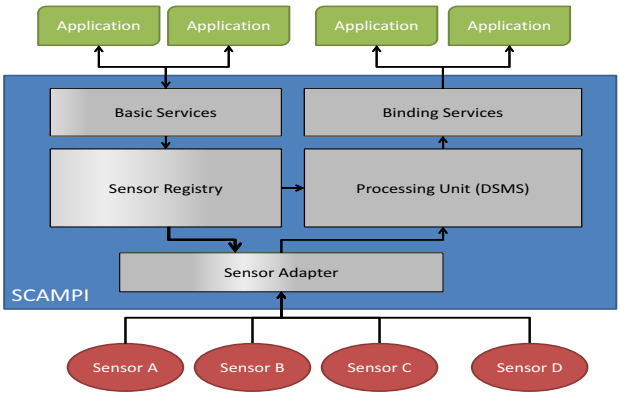

Figure 2: SCAMPI Architecture

The **Sensor Adapter Component** is responsible for the communication with the participating heterogeneous sensor sources. Thereby the component acts as a proxy between the sensor and the middleware allowing a bidirectional communication. The component receives sensor data from the sensor sources and pushes them to the processing component. The Sensor Adapter Component follows a modular design allowing to plug-in protocol wrappers for transferring of sensor data at runtime. Currently the Sensor Adapter Component supports the following protocols: NMEA for GPS positions, OpenGIS Observations and Measurements Encoding Standard (O&M) for SWE [Ope] supporting sensors, and the SCAI protocol developed within the SCAMPI project.

The **Sensor Registry Component** is responsible for managing registered sensors and registered processing instructions. It provides an internal access point for sensor lookups and publishes this service to the Sensor Adapter, the Processing Unit and the Basic Services.

The **Processing Unit** performs the processing of incoming sensor data based on predefined processing instructions. To do so the component uses the Datastream Management System Odysseus [JG08]. Through the Sensor Registry a member of a community can configure

these operators and create application specific instances of them like specific time windows or geographic filter. Furthermore the developer or user can add user-defined aggregation operators and filter operators in different implementation languages. At the current state of development, the middleware supports operators implemented in Java, Python, and Ruby. After the processing of sensor data, the Processing Unit transfers the results to the Binding Services Component.

The **Binding Services Component** is responsible for transforming the processed sensor data into different protocols and transferring the results to other applications. At the current state of development, the Binding Service Component includes support for the HTTP and the upcoming WebSocket protocol, handler for the Java Messaging Service, and bindings for JDBC for persistent storage.

The **Basic Services Component** includes different web mashups to manage the middleware. These mashups include the Sensor Registry Editor, the Operator Editor and the Visualization Service. The Sensor Registry Editor provides a graphical interface for managing the registered sensors and defining the internal data types used for the sensor data. The Operator Editor helps communities to define and edit the processing of sensor data. The visualization service allows visualizing the processed sensor data through configurable charts.

4 Community Sensor Processing and Usage

In this section, we describe how communities can use SCAMPI to process and visualize the different sensor data to develop their desired web-application. To process sensor data, SCAMPI allows communities to define multiple virtual sensors. A virtual sensor is based on the processing of physical sensors and other virtual sensors. To define virtual sensors, members of a community define a processing by configuring available processing operators and linking them together with available sensor sources and one or more outputs. For this task, community members can use the Operator Editor mashup. Users can connect the sources, operators and outputs in this mashup in a box and arrow style. This mashup consists of a JavaScript library that is seamlessly integratable into any website.

Depending on the application the community can bind a processing output to one or more binding services for further usage. The binding service provides multiple visualizations – like line charts and pie charts – that are embeddable into a web application. The SCAMPI middleware transfers processed sensor data to the visualizations using the WebSocket protocol. This allows a continuous visualization of the data without polling from the client side.

Communities can also share their virtual sensors with other communities or companies. To do so, the middleware arranges sensors into domains and categories. A sensor is part of one or more domains that defines the access rights for the sensor. Categories are a hierarchical view on the registered sensors to allow a fast lookup of required sensors. This allows for example that a transport company can share their filtered position data for jam monitoring while still considering the privacy of the driver. In return, this company can use

the aggregated weather data published by a weather community to optimize their routing.

5 Conclusion

In this paper, we show how SCAMPI, a sensor configuration and aggregation middleware for heterogeneous sensors, can help communities to develop their individual sensor based web-application like the described weather application. Beyond that, other web applications have been realized in our research project like a web application to observe pets and visualize their position and relationship between each other on a map, or a mobility web application to find the nearest parking deck based on the measurements of the induction loops at the entrance of the parking decks. Future work includes configuring of participating sensor sources either through members of a community or through processed sensor data. Another topic of research is the user notification over multiple communication protocols like SMS and instant messenger based on the processing of sensor data.

References

[AHS07] Karl Aberer, Manfred Hauswirth, and Ali Salehi. Infrastructure for Data Processing in Large-Scale Interconnected Sensor Networks. In Christian Becker, Christian S. Jensen, Jianwen Su, and Daniela Nicklas, editors, *MDM*, pages 198–205. IEEE, 2007.

[GPU10] Vipul Gupta, Arshan Poursohi, and Poornaprajna Udupi. Sensor.Network: An open data exchange for the web of things. In *PerCom Workshops*, pages 753–755. IEEE, 2010.

[GRL+08] Levent Gurgen, Claudia Roncancio, Cyril Labbé, André Bottaro, and Vincent Olive. SStreaMWare: a service oriented middleware for heterogeneous sensor data management. In *ICPS '08: Proceedings of the 5th international conference on Pervasive services*, pages 121–130, New York, NY, USA, 2008. ACM.

[JG08] Jonas Jacobi and Marco Grawunder. ODYSSEUS: Ein flexibles Framework zum Erstellen anwendungsspezifischer Datenstrommanagementsysteme. (in German). In Hagen Höpfner and Friederike Klan, editors, *Grundlagen von Datenbanken*, volume 01/2008 of *Technical Report*, pages 86–90. School of Information Technology, International University in Germany, 2008.

[Ope] Open Geospatial Consortium (OGC). Sensor Web Enablement (SWE). Specification.

[Ora] Oracle Corporation. Oracle Sensor Edge Server. Website. Available online at `http://www.oracle.com/technology/products/sensor_edge_server/index.html`; visited on 2010-11-18.

[SNL+06] Andre Santanche, Suman Nath, Jie Liu, Bodhi Priyantha, and Feng Zhao. SenseWeb: Browsing the Physical World in Real Time. In *Demo Abstract, ACM/IEEE IPSN 2006, Nashville, TN*, April 2006.

[Yah07] Yahoo. FireEagle: Centralized management of user location. Technical report, Yahoo Research, 2007.