The Challenges of Constructing a Sensor Web Data Stream Management System

Chih-Yuan Huang[1] and Steve Liang[2]
[1]Center for Space and Remote Sensing Research, National Central University
No. 300, Jhongda Rd., Jhongli City, Taoyuan County 32001, Taiwan
Email: cyhuang@csrsr.ncu.edu.tw
[2]Department of Geomatics Engineering, University of Calgary
2500 University Dr. NW, Calgary, AB T2N 1N4, Canada
Email: liangs@ucalgary.ca

KEY WORDS: World-Wide Sensor Web, Data Stream Management System, Geospatial Cyber-infrastructure

ABSTRACT: The world-wide sensor web provides a powerful monitoring capability for users to monitoring the physical world at spatial and temporal scales that were impossible in the past. With the advance of sensor technologies and the definition of interoperable open standards, the sensor web connects various types of sensors located worldwide and performs observations at high frequency. By linking the large amount of sensor data streams produced by the sensor web, scientists can observe phenomena that were previously unobservable. In order to harness the full potential of the continuous monitoring power of the sensor web, efficiently processing sensor web data streams and providing timely notifications for time-critical events (e.g., disasters or security breaches) are necessary. While there have been general-purpose systems proposed to handle continuous data streams (e.g., data stream management systems), we observed that there are unique challenges caused by the nature of geospatial sensor web data and sensor web data sources. Therefore, in this paper, we present our observations on the challenges of constructing a world-wide sensor web data stream management system.

1. INTRODUCTION

In recent years, the sensor arrays have been applied to monitor various kinds of physical phenomena in the world. While there are large scale sensor arrays such as the ARGOS buoys network and the weather networks of the World Meteorological Organization, small- to medium-scale sensor arrays operated by individual scientists become technologically and economically feasible with the advent of the low-cost sensor networks and data loggers. A large amount of effort, such as Global Earth Observation System of Systems (GEOSS) and National Oceanic and Atmospheric Administration (NOAA) Integrated Ocean Observing System (IOOS), has been put forth to web-enable these large-scale sensor networks so that the sensors and sensor data can be accessible through the World-Wide Web (WWW). With open standards defining data schemas and web interfaces, sensors and sensor data can be integrated in an interoperable manner, which is the main idea of the *World-Wide Sensor Web* (Liang et al. 2005; Botts et al. 2007; Bröring et al. 2011). Similar to the WWW, which acts essentially as a “World-Wide Computer”, the sensor web can be considered as a “World-Wide Sensor” or a “cyberinfrastructure for sensors”. This World-Wide Sensor is capable of monitoring the physical world at spatial and temporal scales that were previously impossible.

Similar to the WWW, the sensor web has three major layers, namely, the data layer, the web service layer, and the application layer. Our view of the sensor web layer stack is shown in the Figure 1 (Liang and Huang 2013). The data layer can be further divided into the physical layer and the sensor layer. While the data layer performs observations and transmits sensor data to the web service layer, the web service layer provides the application layer with access to the cached sensor data.

Since the architecture of a sensor web and WWW are similar, the current sensor web development is built on top of many existing WWW standards. However, as content on the sensor web is fundamentally different from that on the WWW, additional open standards should be defined. The Open Geospatial Consortium (OGC) Sensor Web Enablement (SWE) has defined a suite of open standards for the sensor web (Botts et al. 2007), including specifications for data models, data encodings, and web service interfaces. The development and adaption of sensor web open standards is one of the necessary steps to realize the sensor web vision.

With the development and deployment of sensor web and sensor network technologies, the sensor web generates tremendous volumes of streaming data that enable scientists to observe previously unobservable phenomena. These technologies are increasingly attracting the interest of researchers for a wide range of applications, including large-scale environmental monitoring (Hart and Martinez 2006; Stasch et al. 2012), civil structures (Xu 2002), roadways (Hsieh 2004), and animal habitats (Mainwaring et al. 2002).

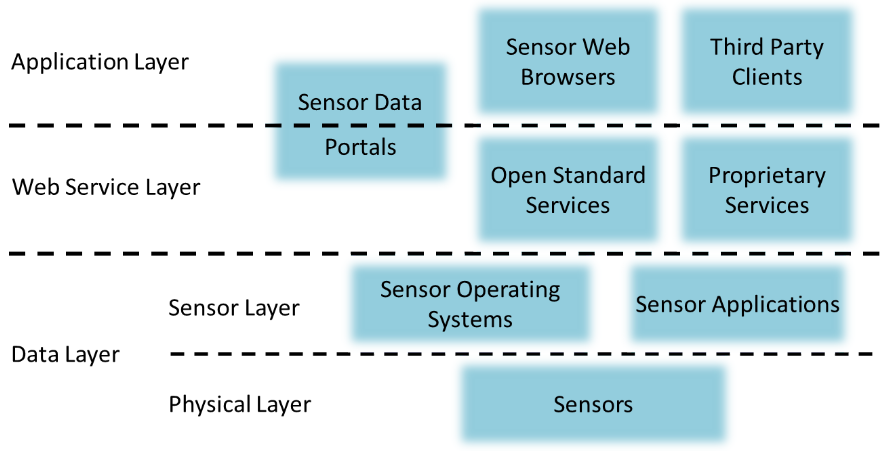

Figure 1 The sensor web layer stack.

Among the applications applying sensor web technology, some of them are time-critical and require efficient data processing for timely decision-making and notification, such as emergency response systems (Kassab et al. 2010). These time-critical applications may be used to support decision making when handling time-critical events such as natural disasters. As the world-wide sensor web vision is to connect various types of sensors that are located worldwide and perform observations at high-frequency, the sensor web may have the ability to capture these time-critical events and provide up-to-date information to support decision making. We believe that by efficiently converting sensor web data into information and providing timely notifications, we can lower down or even prevent the damage from time-critical events.

However, the traditional *request/response communication model* and database management system (DBMS) solutions are not suitable for efficiently processing continuous data streams as they are (1) based on point-to-point *pulling* interaction between users and data providers and (2) not designed for rapid and continuous data streams (Babcock at al. 2002). For instance, in a system following the request/response model (*i.e.*, *pull-based* system), each communication between a user and the system is a complete transaction (*i.e.*, one request and one response), in which the response is evaluated with a snapshot of the system. However, consider a use case requiring timely notifications, no user can predict when an event will happen (e.g., a start of a fire, a collapse of a bridge, or simply an observation). A request cannot be scheduled ahead of time. By the time a user sends requests and receives responses, events may be already outdated.

There is an alternative communication model known as the *continuous processing model*, which allows users to register queries in a system and the system executes the queries whenever it receives new data. In this case, the continuous processing model can provide notifications more promptly than the request/response model.

The fundamental distinction between the request/response and continuous processing model is the *one-time ad-hoc* queries in the request/response model and the *continuous predefined* queries in the continuous processing model (Babcock et al. 2002). One-time queries are only evaluated once with a snapshot of dataset, and then answers are returned to users at the moment queries are evaluated. Continuous queries, however, are stored in the system and evaluated whenever new data arrive. The answers of continuous queries are produced over time and sent to users as notifications or streams when new data meet users' query criteria. Hence, systems follow the continuous processing model are also called as *push-based systems*.

There have been different types of systems applying the continuous processing model to manage and process continuous data streams, such as the publish/subscribe system (Eugster et al. 2003), the simple event processing system (Michelson 2006), the data stream management system (DSMS) (Babcock 2002; Golab and Ozsu 2003; Cugola and Margara 2010), and the complex event processing system (CEP) (Luckham 2002). Although the original designs of these systems are different in terms of the targeted data type and query complexity, some of their functionality starts to overlap as they have evolved.

However, we have observed that in addition to the challenges that general-purpose push-based systems are addressing, the construction of a sensor web DSMS faces unique challenges from the geospatial nature of sensor web data and the web protocols of standard-based data services. Therefore, we need to identify these challenges before start designing and implementing the system. Hence, the major objective of this paper is to identify the challenges for constructing a sensor web DSMS.

The remainder of this paper is organized as follows. Section 2 introduces the related general-purpose DSMSs and the challenges they are addressing. In Section 3, we present the identified challenges for constructing a sensor web DSMS. Finally, Section 4 contains the conclusions and future work.

2. RELATED WORK

In order to design a comprehensive architecture for a sensor web DSMS, we first review the challenges and existing solutions in general-purpose push-based systems. As mentioned earlier, these push-based systems follow the same concept of continuous query processing model, where users can register queries and receive notifications when new data match their queries. In order to provide an overview for general-purpose DSMSs, we summarize and categorize the challenges and existing solutions that are common in these systems.

There have been papers summarizing these general-purpose push-based systems such as Babcock et al. (2002), Eugster et al. (2003), Golab and Ozsu (2003), Cugola and Margara (2010). Readers interested in further details of this section are referred to these papers. In order to provide a high-level overview of challenges these systems, we observed that the challenges can be categorized into two major issues, namely *memory* and *query efficiency*.

Since push-based systems aim on handling continuous data streams, which are large in size and generated at a high rate, processing these continuous data streams could easily exhaust available memory (*i.e.*, memory issue) or result in poor query performance (*i.e.*, query efficiency issue). In order to address these two issues, previous works proposed several approaches. Here we categorize these approaches into three types, namely *approximation*, *query optimization*, and *adaptivity* approaches. We further explain these three categories as follows:

1. *Approximation approaches*: Due to the limited memory resource in a system, providing exact answers for data stream queries is not always possible. In this case, a high-quality approximated answer is preferable. Approaches belonging to this category are (1) sliding windows, (2) batch processing, (3) replacing blocking operators by non-blocking operators, (4) load shedding, (5) synopsis construction, and (6) the k-constraint in the STREAM system (Arasu et al. 2004). While the major objective of approximation approaches is to reduce the memory usage, some of these approaches indirectly speed up the query processing as they reduce the input data size.

2. *Query optimization approaches*: A typical approach to perform steam query processing is to first create a query execution plan (Figure 2), and then each new data will traverse through the operators and queues in the query plan. There have been approaches proposed to optimize queries by modifying the structure of query plans. For example, (1) minimizing the number of intermediate results (based on the *selectivity* of operators) before performing high-overhead operators (Arasu et al. 2004), (2) sharing synopses and operators across similar query plans (Arasu et al. 2004), (3) using an incremental evaluation approach to process only the new and expired data, and (4) distributing query plans to multiple processing nodes (Mokbel et al. 2005). These approaches are mainly designed for improving the query efficiency and would not degrade the quality of answers.

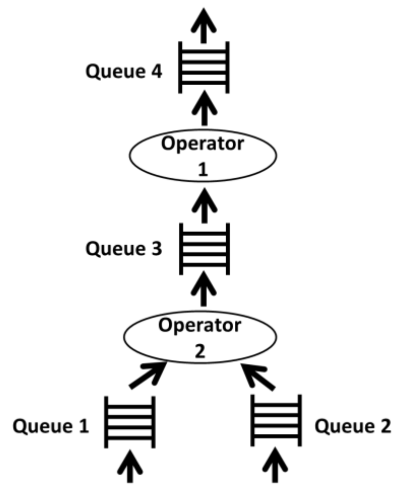

Figure 2 An example of a query execution plan.

3. *Adaptivity approaches*: The throughputs of query plans vary with the dynamic nature of data streams and queries. For example, the arrival rate of data streams, the number of intermediate results, and the available CPU and memory resources all change over time. In order to ensure the query plans are the most efficient for the current system condition, approaches are proposed to optimize the system performance on-the-fly. For example, the Eddies approach (Avnur and Hellerstein 2000) first routes data through query operators to discover fast and selective operators and then dynamically re-organize query plans. The StreaMon module in STREAM employs three components (*i.e.*, Profiler, Reoptimizer, and Executor) to adaptively adjust the *k* values of k-constraint and the structure of query plans (Arasu et al. 2004).

This review of the challenges and existing solutions in general-purpose push-based systems provides a foundation for designing a sensor web DSMS. While some of the aforementioned solutions could be directly applied to a sensor web DSMS, we found that the sensor web data streams and sensor web data sources cause some unique challenges. Therefore, before designing and implementing a sensor web DSMS, we need to identify the challenges to be addressed, which is the main objective of this paper.

3. IDENTIFIED CHALLENGES FOR CONSTRUCTING A SENSOR WEB DATA STREAM MANAGEMENT SYSTEM

In this section, we try to provide a comprehensive understanding about the challenges for constructing a geospatial DSMS in the sensor web context. While some of the challenges may be common with other general-purpose systems, others are unique because of the nature of geospatial sensor web data and sensor web data sources.

As we explained earlier about the sensor web layer stack (Figure 1), the data layer performs observations and transmits sensor data to the web service layer, and the web service layer provides the application layer with access to the cached sensor data. Since a sensor web DSMS is an application that processes sensor web data streams with predefined queries, a sensor web DSMS needs to consider the challenges of communicating with data sources in the web service layer. Moreover, as sensor web data is geospatial in nature, a sensor web DSMS would have unique challenges in handling geospatial queries or visualizing geospatial data.

We identify and present the major challenges for building a geospatial sensor web DSMS in the following list. In general, the first four challenges are related to the data sources, the fifth and sixth challenges are mainly about the query processing, and the seventh challenge locates in the client side. In addition, to provide a different view for the challenges, the first and fifth challenges arise from the nature of continuous data streams, which is similar to the challenges in general-purpose systems, while others are more related to the sensor web.

1. *Large amount of continuous data streams*: Although large amount of data streams allows users to observe events that are previously unobservable, traditional DBMS approaches are not designed for the rapid and continuous arrival characteristic of data streams (Babcock et al. 2002; Golab and Ozsu 2003). This challenge is also the main challenge that general-purpose push-based systems aim to address.

2. *Pull-based data sources*: We have observed that many of the sensor data sources only support a request/response communication model, e.g., OGC Sensor Observation Services (SOSs). Users need to proactively send data retrieval requests in the first place. As mentioned earlier, no user can know when new sensor observations will be available in data sources. In order to retrieve near-real-time data from pull-based data sources, a naïve approach is to send requests to data sources frequently. However, many of these requests would be unnecessary (*i.e.*, no new data returned) and cause extra burden on both clients and servers. Therefore, how to retrieve near-real-time sensor data from pull-based sensor web data sources with an acceptable overhead is one of the major challenges.

3. *Large number of data sources*: Follow on the previous challenge, if a data source only supports a pull-based communication model, at least one Internet connection is needed to frequently retrieve data from that data source. While the sensor web vision is to connect all the sensors in the world, a large number of sensor web data services are expected to host generated sensor data. Retrieving near-real-time sensor data streams from a large number of data sources requires a large number of Internet connections. However, the management of these connections becomes a critical challenge for a client that only has limited resource.

4. *Heterogeneous sensor web data*: We observed that the sensor web is highly heterogeneous in terms of communication protocols, data encodings, semantics, syntactic, etc. (Knoechel et al. 2011). Some of these heterogeneities can be addressed by applying open standard protocols, such as communication protocols and data encodings. However, even in the same open standard ecosystem (e.g., OGC SWE), there are still interoperability issues due to the lack of standardized naming, such as semantic heterogeneity and syntactic

heterogeneity. As an example for semantic heterogeneity, consider the two strings "precipitation" and "rainfall". Since rainfall is a type of precipitation, a user interested in precipitation data would likely be interested in rainfall. Although these concepts are intuitively related to human, to any computer these are simply different sequences of characters.

The syntactic heterogeneity comes from the various labels used to represent the same concept, as different data providers may label their data differently. For example, Table 1 shows an example of the various URIs used in SOSs to represent the concept of *wind speed*. This syntactic heterogeneity causes difficulties for a system to integrate all sensor data about wind speed. As a result, how to integrate heterogeneous sensor web data and provide users with a coherent view of sensor web data is one of the unique challenges for a sensor web DSMS.

Table 1 Various URIs used in OGC SOSs to represent the concept of wind speed.

1	urn:x-ogc:def:property:OGC::WindSpeed
2	urn:ogc:def:property:universityofsaskatchewan:ip3:windspeed
3	urn:ogc:def:phenomenon:OGC:1.0.30:windspeed
4	urn:ogc:def:phenomenon:OGC:1.0.30:WindSpeeds
5	urn:ogc:def:phenomenon:OGC:windspeed
6	urn:ogc:def:property:geocens:geocensv01:windspeed
7	urn:ogc:def:property:noaa:ndbc:Wind Speed
8	urn:ogc:def:property:OGC::WindSpeed
9	urn:ogc:def:property:ucberkeley:odm:Wind Speed Avg MS
10	urn:ogc:def:property:ucberkeley:odm:Wind Speed Max MS
11	http://marinemetadata.org/cf#wind speed
12	http://mmisw.org/ont/cf/parameter/winds

5. *Large number of queries*: There have been a number of applications applying world-wide sensor web on large-scale monitoring (Xu, 2002). It is foreseeable that numerous domain specialists or the general public will generate various types of queries to receive timely notifications from a sensor web DSMS. Therefore, how to maintain query efficiency while handling a large number of queries becomes one of the major challenges. While this challenge also happens in general-purpose push-based systems, the concepts of existing solutions would benefit the design of a sensor web DSMS.

6. *Geospatial data and queries*: One of the major differences between a sensor web DSMS and general-purpose push-based systems is the geospatial nature of sensor web data. While many general-purpose operators are simple and efficient enough to be applied directly in query execution plans, some geospatial operators are complex and time-consuming when processing a large number of geometries, such as topological operators (Clementini et al. 1994). While aiming on providing timely notifications, these geospatial operators would become performance bottlenecks. As a result, how to efficiently process geospatial operators in a publish/subscribe system becomes a critical research question for a sensor web DSMS.

7. *Sensor web data visualization*: Due to the geospatial nature of sensor web, the data streams, queries, and notifications are inherently geospatial. How to enable users to visualize sensor web data, create queries, and display notifications in a geospatial manner is another unique challenge for a sensor web DSMS.

4. CONCLUSTIONS AND FUTURE WORK

We have identified and presented the challenges in constructing a geospatial sensor web DSMS. While the vision of the world-wide sensor web is to connect various types of sensors in the world together, we believe that efficiently processing the continuous sensor data streams is necessary to provide timely notifications for decision making or early warning, which will consequently unleash the full potential of the sensor web. In this paper, we briefly discussed the differences between request/response communication model (*i.e.*, pull-based model) and continuous query processing model (*i.e.*, push-based model). We reviewed and summarized the main challenges that existing general-purpose push-based systems are addressing and their proposed solutions. Due to the nature of geospatial sensor web data streams and sensor web data sources, we observed that in additional to the existing challenges, there are unique challenges to be addressed in order to construct a geospatial sensor web DSMS. Hence, the major contribution of this paper is the identification of the challenges for constructing a sensor web DSMS.

In terms of the future work, we believe that each of the identified challenges is an interesting and important research question that is worth investigated. We are also aiming on designing and implementing a sensor web data stream management system, which includes proposing solutions to address the identified challenges. Our final goal is to construct a geospatial sensor web data stream management system that can accept users' geospatial queries, efficiently process those queries with newly received/retrieved sensor data streams, and finally notify users if the new data matches their query criteria.

REFERENCE

Arasu, A., Babcock, B., Babu, S., Cieslewicz, J., Datar, M., Ito, K.,Motwani, R., Srivastava, U., and Widom, J., 2004. STREAM: The Stanford Data Stream Management System.

Avnur, R. and Hellerstein, J., 2000. Eddies: Continuously Adaptive Query Processing. In: ACM Special Interest Group on Management of Data (SIGMOD), pp. 261-272.

Babcock, B., Babu, S., Datar, M., Motwani, R., and Widom, J., 2002. Models and Issues in Data Stream Systems. In: ACM Symposium on Principles of Database Systems, pp. 1-16.

Botts, M., Percivall, G., Reed, C., and Davidson, J., 2007. OGC® Sensor Web Enablement: Overview and High Level Architecture (OGC 07–165). Open Geospatial Consortium White Paper, 28 December 2007.

Bröring, A., Echterhoff, J., Jirka, S., Simonis, I., Everding, T., Stasch, C., Liang, S., Lemmens, R., 2011. New Generation Sensor Web Enablement. Sensors, 11, pp. 2652-2699.

Clementini, E., Sharma, J., and Egenhofer, M.J., 1994. Modeling Topological Spatial Relations: Strategies for Query Processing. Computers and Graphics. 18 (6), pp. 815-822.

Cugola, G. and Margara, A., 2010. Processing Flows of Information: From Data Stream to Complex Event Processing. Technical report. Politecnico fi Milano.

Eugster, P., Felber, P.A., Guerraoui, R., and Kermarrec, A.M., 2003. The Many Faces of Publish/Subscribe. ACM Computing Surveys, 35 (2), pp. 114-131.

Golab, L. and Ozsu M.T., 2003. Issues in Data Stream Management. In: The 2003 ACM Special Interest Group on Management of Data, 32 (2), pp. 5-14.

Hart, J.K., Martinez, K., 2006. Environmental Sensor Networks: A Revolution in the Earth System Science? Earth Science Review, 78, pp. 177–191.

Hsieh, T.T., 2004. Using Sensor Networks for Highway and Traffic Applications. IEEE Potentials, 23, pp. 13-16.

Kassab, A., Liang, S., and Gao, Y., 2010. Real-Time Notification and Improved Situational Awareness in Fire Emergencies using Geospatial-based Publish/Subscribe. International Journal of Applied Earth Observation and Geoinformation, 12 (6), pp. 431-438.

Knoechel, B., Huang, C.Y., Liang, S.H.L., 2011. Design and Implementation of a System for the Improved Searching and Accessing of Real-World SOS Services. In: Proceedings of 2011 International Workshop on Sensor Web Enablement, Banff, AB, Canada.

Liang, S.H.L., Croitoru, A., Tao, C.V., 2005. A Distributed Geospatial Infrastructure for Sensor Web. Computers & Geosciences, 31, pp. 221-231.

Liang, S.H.L. and Huang, C.Y., 2013. GeoCENS: A Geospatial Cyberinfrastructure for the World-Wide Sensor Web. Sensors, 13 (10), pp. 13402-13424.

Luckham, D., 2002. The Power of Events: An Introduction to Complex Event Processing in Distributed Enterprise Systems. Addison-Wesley.

Mainwaring, A., Polastre, J., Szewczyk, R., Culler, D., Anderson, J., 2002. Wireless Sensor Networks for Habitat Monitoring. In: The 2002 ACM International Workshop on Wireless Sensor Networks and Applications, Atlanta, GA, USA.

Michelson, B.M., 2006. Event-Driven Architecture Overview, Patricia Seybold Group.

Mokbel, M.F., Xiong, X., and Aref, W.G., 2005. Continuous Query Processing of Spatio-Temporal Data Streams in PLACE. GeoInformatica, 9 (4), pp. 343-365.

Stasch, C., Foerster, T., Autermann, C., Pebesma, E., 2012. Spatio-Temporal Aggregation of European Air Quality Observations in the Sensor Web. Towards A Geoprocessing Web, 47, pp. 111-118.

Xu, N., 2002. A Survey of Sensor Network Applications, IEEE Communications Magazine, 40, pp. 102-144.