Hindawi Publishing Corporation
International Journal of Distributed Sensor Networks
Volume 2015, Article ID 914612, 14 pages
http://dx.doi.org/10.1155/2015/914612

Research Article

A Data Stream-Based, Integrative Approach to Reliable and Easily Manageable Real Time Environmental Monitoring

Meilan Jiang,[1] Jonghyun Lee,[2] Karpjoo Jeong,[3,4] Zhenguo Cui,[5] Bomchul Kim,[6] Suntae Hwang,[7] and Young Jean Choi[8]

[1]*Department of Advanced Technology Fusion, Konkuk University, 120 Neungdong-ro, Gwangjin-gu, Seoul 05029, Republic of Korea*
[2]*Department of Computer Engineering, Konkuk University, 120 Neungdong-ro, Gwangjin-gu, Seoul 05029, Republic of Korea*
[3]*Department of Internet and Multimedia Engineering, Konkuk University, 120 Neungdong-ro, Gwangjin-gu, Seoul 05029, Republic of Korea*
[4]*Institute for Ubiquitous Information Technology and Applications, Konkuk University, 120 Neungdong-ro, Gwangjin-gu, Seoul 05029, Republic of Korea*
[5]*BaaS Division, APEX Platform, 253 Pangyo-ro, Bundang-gu, Seongnam-si, Gyeonggi-do 13486, Republic of Korea*
[6]*Department of Environmental Science, Kangwon National University, 1 Kangwonddaehak-gil, Chuncheon-si, Gangwon-do 24341, Republic of Korea*
[7]*School of Computer Science, Kookmin University, 77 Jeongneung-ro, Seongbuk-gu, Seoul 02707, Republic of Korea*
[8]*WISE Institute, Hankuk University of Foreign Studies, Global Campus, 81 Oedae-ro Mohyueon-myeon, Cheoin-gu, Yongin-si, Gyeonggi-do 17035, Republic of Korea*

Correspondence should be addressed to Karpjoo Jeong; jeongk@konkuk.ac.kr

Received 3 July 2015; Revised 6 October 2015; Accepted 18 October 2015

Academic Editor: Zhenhong Li

Real time environmental monitoring (hereafter, RTEM) is crucial for observing and studying dynamic, rare, or abrupt phenomena in the environment. Conventional RTEM systems are developed in a domain-specific way and not based on a well-defined distributed system model. Therefore, it is challenging to develop such RTEM systems, to extend them to support new requirements, and to integrate them with other application systems. In this paper, we raise challenging issues in RTEM, propose a generic distributed system model for RTEM to address those issues, and present the Galilee middleware system for RTEM based on the model. The distributed system model is based on the concept of data streams. It is intended to simplify and to conceptualize the design of RTEM systems. Therefore, the simple data stream-based distributed system model facilitates the development of general RTEM middleware system. The Galilee middleware system for RTEM is based on the distributed system model and designed to support various RTEM applications in a reliable and easily manageable way. In this paper, we also demonstrate and evaluate the current prototype implementation of the Galilee system with those datasets obtained from the previous sensor-based water quality monitoring project for Lake Soyang from 2011 to 2012.

1. Introduction

Environmental monitoring consists of the processes and the activities to observe the environment [1]. It is crucial for both environmental sciences and related sciences such as ecology. There are various environmental monitoring methods or technologies. A variety of human-involved sampling and lab-based sample analysis methods have been widely used but are only effective for *sporadic* or *periodic monitoring*.

As opposed to such sporadic or periodic monitoring, real time and continuous monitoring allow scientists to observe and study dynamic, rare, or abrupt phenomena in the environment. Such real time monitoring basically requires sensors for observation and measurement. Due to technical advances in sensors, wireless communication, and embedded computing, such sensor-based RTEM now becomes a viable approach to environmental research.

In fact, there have been lots of research efforts on sensor networks, sensor data repositories, and application systems for RTEM, respectively. However, there has been little research work on how to integrate them in a logical, extensible, and easily maintainable way. Therefore, conventional RTEM systems are developed in a domain-specific way and not based on a well-defined distributed system model. For environmental research, it is challenging even for IT professionals to develop such RTEM systems, to manage them, to extend them to support new requirements, and to integrate them with other application systems, effectively.

In this paper, we propose a generic distributed system model for RTEM that is based on the concept of data streams and intended to simplify and conceptualize the design of RTEM systems. In this model, sensor networks, sensor data repositories, sensor management, and applications are logically defined according to the concept of data streams. This simple and consistent system model facilitates the system development, the system management, the integration with other systems, and future system extensions for RTEM applications.

We also present the RTEM middleware system called Galilee whose design is based on the distributed system model and intended to facilitate the development of various RTEM applications. The current Galilee development is based on three middleware systems: the sensor network system called CSN (Conceptually Manageable Sensor Network), the sensor data repository called S4EM (Simple Sensor Data Stream Management System for Environmental Monitoring), and the complex event processing system called Esper [2–4]. The designs of three middleware systems are based on the concept of data streams. Their interactions and integration are also designed according to the concept of data streams.

In this paper, we also demonstrate and evaluate the current prototype implementation of the Galilee system with those datasets obtained from the previous sensor-based water quality monitoring project for Lake Soyang from 2011 to 2012. The water quality monitoring project was carried out by using a simple monitoring system.

The rest of this paper is organized as follows. In Section 2, we discuss challenging issues in RTEM. In Section 3, we explain our main design approach. Sections 4 and 5 describe the Galilee system in terms of the system design and implementation, respectively. In Section 6, we discuss related work. In Section 7, we finally talk about conclusions and future work.

2. Challenging Issues in RTEM

2.1. Sensor-Based Environmental Monitoring. RTEM requires sensors to monitor various phenomena about the environment continuously and autonomously. A single sensor can be used alone or a set of sensors (called a *sensor network*) can be used together, depending on both requirements for monitoring and the availability of sensors.

Sensor network middleware is a software system designed to manage sensors, communication networks, and data delivery. Such middleware allows sensors to send their data to a sensor data repository, applications to access sensor data in a real time manner, and administration staffs to manage sensors.

Sensor-based RTEM raises *serious management challenges* to environmental scientists [5–11]:

(i) Support for heterogeneous sensors: environmental research requires monitoring a variety of properties about the environment and therefore various sensors ranging from inexpensive small primitive ones to expensive advanced ones with data logging and communication supports. Sensor network middleware is required to integrate and manage these heterogeneous sensors easily and reliably.

(ii) Intelligent support for sensor maintenance: environmental sensors require periodic cleaning and calibration because sensors are exposed to physical, chemical, and biological materials such as water, air, chemical pollutants, and microbes. However, such sensor maintenance requires a great amount of time and effort because researchers or maintenance staffs must visit environmental monitoring sites. At the same time, if sensors are not effectively maintained, then data from those sensors may be inaccurate. Therefore, sensor maintenance must be intelligently supported.

(iii) Reliable management of communication networks: in environmental research sites that usually have no TCP/IP network connection, environmental scientists often need to build communication networks for their sites. In addition, they need to build base stations (a kind of computer system) in their sites that are connected to the TCP/IP network, collect data from sensors, and send data to a sensor data repository. Those sites are usually very distant from offices where environmental scientists and maintenance staffs work. Therefore, the reliable management of communication networks is crucial for environmental monitoring.

(iv) Support for large scale environmental monitoring: the scope of environmental research has been growing larger in time and space. Therefore, the area that environmental monitoring needs to cover has also been growing larger and larger. Sensor networks must be able to cover large geographical areas and this raises a serious challenge to traditional wireless sensor networks that are designed for small areas.

2.2. Sensor Data Stream Management. In sensor-based real time monitoring, a sensor generates a stream of monitoring data records continuously and there are many sensor data streams generated from sensor networks. A sensor repeats the same monitoring operation and generates a single value as a result of each operation. A sensor data stream is just an unbounded sequence of values of the same type.

Therefore, the management of sensor data streams alone is straightforward and simple to support.

However, there are three challenging issues in the management of sensor data [4, 12, 13]:

(i) Support for data integration and synthetic analysis: sensor data can be valuable only when they can be effectively analyzed for environmental research. The analysis of sensor data requires detailed metadata, but the value of sensor data itself does not contain metadata. Furthermore, sensor data needs to be used in other environmental research projects or to be integrated with other observation data for various analyses and syntheses. Environmental research requires monitoring various properties of the environment and needs to synthesize these various monitoring data. For example, when fish monitoring data is analyzed for a lake, water quality data from sensors in the lake is also needed to be considered. In order to address this data integration issue effectively, the management of sensor data must be based on a well-defined environmental data model.

(ii) Reliability and security of the sensor data repository: each record of sensing (i.e., sensor data) contains information about a particular environmental phenomenon that happens only at a certain time and geographical location. Since the same environmental phenomena cannot be recaptured, sensor data should not be lost. In addition, sensors continue to monitor the environment and send sensor data to a sensor data repository once they are installed and started. Therefore, valuable monitoring data can be lost while the sensor data repository fails and cannot receive sensor data. Such system reliability requirements are challenging even for IT professional staffs.

(iii) Scalability of the sensor data repository: environmental research continues monitoring and therefore, sensor data streams are basically unbounded and continue to grow in size. The sensor data repository must be easily scalable.

3. Design Approach

3.1. Data Stream-Based Distributed System Model for RTEM. In order to address those challenging issues discussed in Section 2, we propose a simple distributed system model for RTEM based on the concept of data streams. In this system model, *every system component or data object is designed according to the concept of a data stream* as follows:

(i) A *data stream* is a first class object in RTEM. Each data stream has a unique ID and can be globally referenced by its ID.

(ii) *Sensors* are modeled to generate data streams. In fact, a sensor is conceptually treated as a data stream. Therefore, sensor networks that logically consist of a collection of sensors are considered to generate multiple data streams.

(iii) The *data repository* is assumed to manage unbounded data streams. Therefore, the data repository is considered to be simply a collection of data streams.

(iv) The *communication between system components* is modeled as data streaming.

(v) The *administration system* is designed to manage system components in terms of data streams. For example, when a sensor is registered, the administration system assumes a new data stream to be created in both the sensor networks and the data repository, respectively. The administration system also creates a data streaming channel between them. In addition, the administration system detects faults or abnormal behaviors in sensors by examining data inside data streaming directly.

(vi) *Applications* are considered to access data streams when they access sensors in a real time manner or retrieve persistent data streams from the data repository.

More specific system design approach to the distributed system model is explained in the following subsections. Figure 1 illustrates the distributed system model. A sensor network has two sensors: S1 and S2. The sensors generate data streams D1 and D2, respectively. The data repository has the persistent copies of the entire data streams for S1 and S2. Sensors and the data repository communicate via data streaming. Applications can also access data from sensors in a real time manner via data streaming. The actual communication is implemented based on the Publish/Subscribe model.

The distributed system model intends RTEM systems to be as follows:

(i) *Open*. New sensors or environmental applications can be easily added to the RTEM infrastructure; at the same time, existing sensors or applications can also be deleted or replaced by new ones.

(ii) *Heterogeneous*. A variety of heterogeneous sensors can be supported in the same way. Every sensor is treated and managed according to the same logical model (each sensor just as a single data stream).

(iii) *Decoupling*. Applications and sensors are totally decoupled by data streams. Sensors simply generate data streams and applications read data streams. They interact with each other by data streaming.

3.2. Event Processing-Based Sensor Maintenance Support. In environmental monitoring, those sensors that monitor the environment must also be continuously monitored. In environmental fields, the accuracy of sensors is often compromised because of biological or chemical contamination, heavy rains, strong winds, or even wild animals. In addition, the shortage of battery power may also cause sensors to stop operating.

In those cases, taking prompt maintenance actions is very important; otherwise, sensors start to produce inaccurate data or stop operating. However, the detection of such problems requires either the permanent stay of human staffs

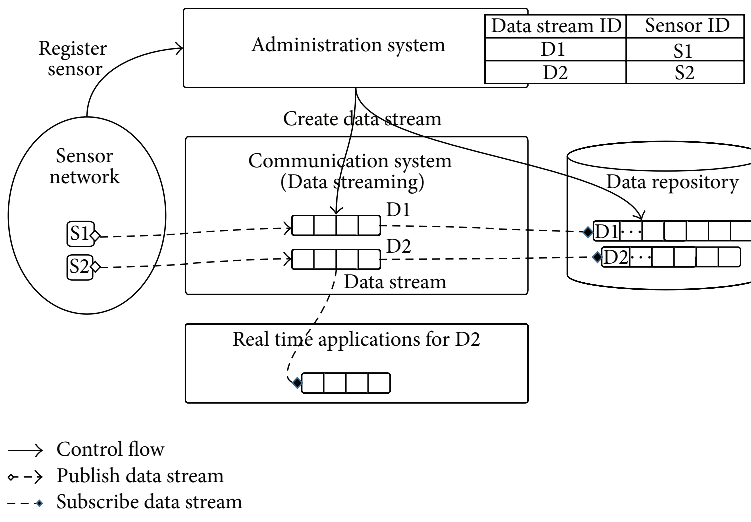


Figure 1: The distributed system model.

or some kind of automatic fault or error detection systems at monitoring sites.

Complex event processing is a promising approach to the automatic detection of problems with sensors. In this approach, data streams from those sensors are examined to recognize abnormal data patterns. There are complex event processing systems such as Esper, Streambase, and Drools Fusion [3, 14, 15]. These systems usually consist of an event processing language (e.g., EPL in Esper) and an event processing engine.

By using such event processing language, event types or event patterns are programmed as statements or rules and registered into event processing engines. Such event processing engines are configured to monitor data streams and to detect occurrences of registered event types or event patterns at runtime.

The Galilee system uses the event processing engine called Esper to detect problems with sensors. The Esper event processing engine is integrated with sensor network middleware and allowed to examine sensor data streams. EPL (the Event Processing Language) is used to specify a variety of abnormal behaviors of sensors easily. The event processing-based detection of sensor abnormal behaviors facilitates the prompt decision on sensor maintenance.

3.3. Support for Cellular Communication Networks. Wireless sensor networks are an effective approach to environmental monitoring [6, 8, 9, 16, 17]. A wireless sensor network (WSN) is a set of sensors which *wirelessly* communicate with gateways or base stations. In WSN, gateways or base stations usually interact with sensor data repositories via *wired* TCP/IP communication networks. In WSNs, sensors usually communicate by WPANs (Wireless Personal Area Networks) such as ZigBee [16]. Due to wireless communication, sensors are flexibly deployed for limited geographical areas. Therefore, WSNs facilitate the environmental monitoring of small areas.

However, it is *challenging to monitor a geographically large area* such as rivers by conventional WSNs. The construction and maintenance of WSNs are challenging for IT professionals because the monitoring of large geographical areas requires many WSNs.

Recent technology advances in Internet of Things and cellular communication networks (e.g., 3G or LTE) provide environmental scientists with opportunities for deploying sensors flexibly and conveniently for geographically large areas and reducing the work on the construction and maintenance of sensor networks significantly [7]. There are currently a variety of small-sized and easily attachable cellular communication modem devices that can make sensors wirelessly and directly connected to sensor network middleware or sensor data repositories without going through private base stations.

Such modems use public communication networks for cellular phones already existing and widely available [18, 19]. When they are integrated with programmable embedded systems, they can easily provide the TCP/IP connection for sensors. Since public cellular communication networks are always constructed and maintained by commercial companies (i.e., mobile network operators such as T-Mobile in USA and SKT in Korea), sensors with the cellular communication modem can immediately start to communicate with sensor network middleware or sensor data repositories wherever cellular communication networks are accessible. In addition, environmental scientists or maintenance staffs do not need to worry about the maintenance of those cellular communication networks once they are accessible. In Korea, those cellular communication networks are accessible in most areas of the country and their coverage continues to expand.

Therefore, the support for cellular communication networks facilitates the development of large scale environmental monitoring infrastructures. Figure 2 shows how a large river can be easily monitored by a number of sensors with the cellular communication.

3.4. Cloud-Based Sensor Data Repository. In the Galilee sensor data repository called S4EM (Simple Sensor Data Stream Management System for Environmental Monitoring),

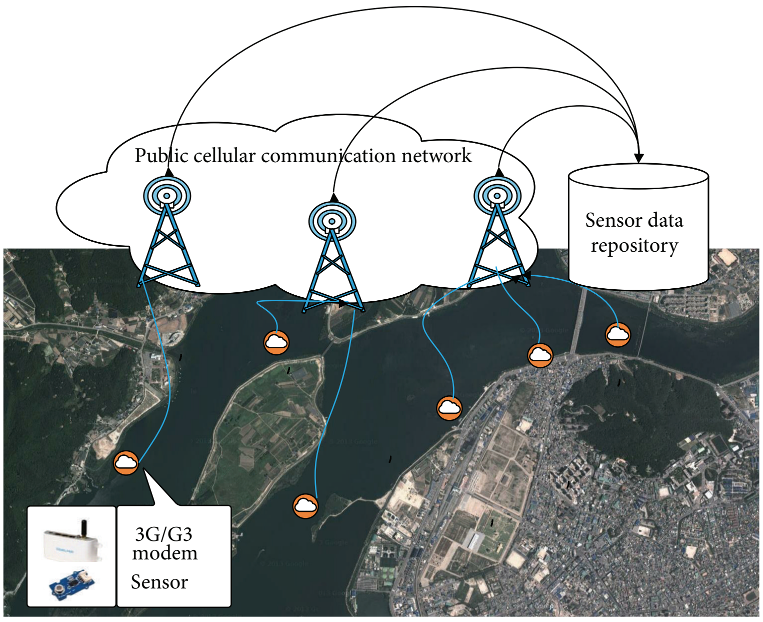


Figure 2: A hypothetical example of cellular communication-based large scale wireless sensor network for Han River which is one of the largest rivers in Korea and whose total length is 494 kilometers.

the data stream from a sensor is managed simply as a single data stream object. Therefore, the sensor data repository consists of a collection of data streams. However, sensor data can be only valuable when they can be effectively analyzed for environmental research [13]. The analysis of sensor data requires two types of data:

(i) Measured data from sensors: data from the sensor measurement is usually a single value. The format and structure of such data from various sensors are generally system-defined and usually similar for various applications.

(ii) Metadata about the observation and measurement: the analysis of the measured data requires information about the observation that is called metadata. Such metadata include information about the monitoring target, the monitored property, the related procedure, and the hardware sensor.

In order to support the later environmental analyses, the sensor data repository must be able to manage both measured data from sensors and metadata separately collected by scientists or staffs. There are data models about environmental observation and measurement including OGC O&M, Vega, and CUAHSI ODM [12, 20, 21]. The S4EM system is currently designed to support the Vega data model developed by the GLEON community [22] to manage data streams from sensors. However, we plan to redesign S4EM to be compatible with OGC O&M in the future.

The development of the S4EM system is based on the cloud computing technology. The management of sensor data is one of the most crucial tasks in RTEM because sensor data is too valuable to lose. Sensor data is not updatable and traditional DBMSs are not required. The construction and effective maintenance of a sensor data repository cause a significant amount of burden on environmental scientists: system maintenance, security provisioning, database backup operations, and the checkup of HW servers, to name a few.

The cloud computing technology provides environmental scientists with opportunities for reducing the work on the construction and maintenance of a sensor data repository for environmental monitoring significantly. Cloud computing is a computing concept, model, or technology where software, platforms, and infrastructures are provided as services that the user can access and use like utilities, via Internet [23, 24]:

(i) SaaS (software as a service): software is available as a service. The user can use the software without worrying about OS or hardware. A web mail service such as Gmail and a storage service such as Dropbox are SaaS examples.

(ii) PaaS (platform as a service): software development environments or database systems are available as services. The user can use PaaS services to develop SaaS services. Google Datastore is a PaaS example.

(iii) IaaS (infrastructure as a service): infrastructures such as server systems are available as a service. The user can install his or her own application software on the IaaS service which can be considered as a virtual machine. But the user does not have to worry about the administration or maintenance of the virtual machine.

In the development of the Galilee system, we choose the Google App Engine (more specifically, the Datastore component) as our PaaS service and implement S4EM as a SaaS service by using the PaaS service. Since S4EM runs on the Google Cloud Infrastructure, environmental scientists do

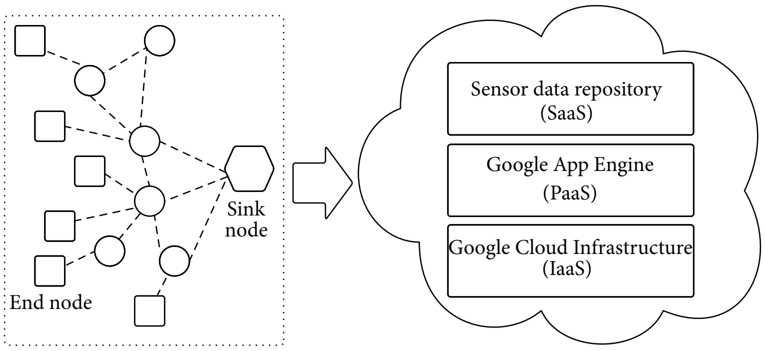

Sink node
End node
Sensor data repository (SaaS)
Google App Engine (PaaS)
Google Cloud Infrastructure (IaaS)

Figure 3: Cloud-based sensor data repository.

not have to worry about system maintenance such as regular hardware checkups, security provisioning, and data backup. Figure 3 shows the structure of the cloud-based sensor data repository.

4. System Design

4.1. System Architecture. Major system components of the Galilee system are as follows:

(i) CSN sensor network middleware: CSN is a general-purpose sensor network system to facilitate the conceptual management of sensor networks and the easy application development. In CSN, every sensor is managed as a data stream generator.

(ii) CSN sensor agent: CSN Sensor Agent is a simple system component to be deployed together with sensors and to manage interactions between the CSN Sensor Network Middleware and hardware sensors.

(iii) Event-driven sensor maintenance support system (ESMS): this ESMS system examines data streams from sensors and detects abnormal types of sensor data or patterns of sensor data streams. Currently, the Esper complex event processing engine is used as ESMS.

(iv) Cloud-based sensor data repository (S4EM): this S4EM system manages sensor data streams and is currently based on Google Datastore (Cloud NoSQL Database).

(v) Data portal: This system provides web-based data access services such as data search and download.

(vi) Real time access support for applications: this service provides application programs with the current information about live data streams and the library code to access those live data streams in a real time manner.

(vii) Potential applications: various environmental applications such as real time prediction, real time alerts, the simulation of prediction models, and the analyses of monitoring data can use real time sensor access services and data portal services in the Galilee system.

Figure 4 shows the major system components of the Galilee system and their layered architecture. These system components are explained in more detail from Section 4.2 through Section 4.6.

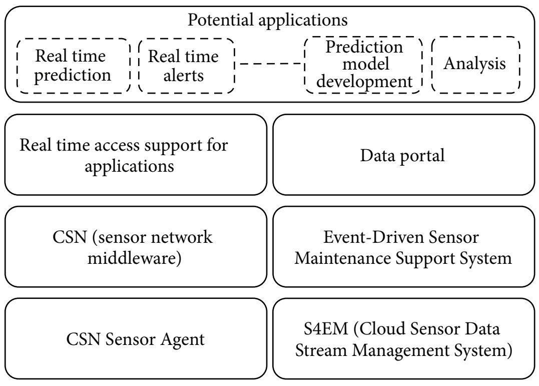

Potential applications
Real time prediction
Real time alerts
Prediction model development
Analysis
Real time access support for applications
Data portal
CSN (sensor network middleware)
Event-Driven Sensor Maintenance Support System
CSN Sensor Agent
S4EM (Cloud Sensor Data Stream Management System)

Figure 4: Galilee system architecture.

4.2. CSN Sensor Network Middleware. In the Galilee system, CSN is the main core system component. CSN is our separate project for the development of a general-purpose sensor network middleware to facilitate the conceptual management of sensor networks and the easy application development. CSN is now integrated into the Galilee system for the construction of sensor networks for environmental monitoring. In this section, we briefly introduce main features of the CSN system. Please refer to another paper on CSN for more details [2].

In CSN, two main system components are *sensor network middleware* and *Sensor Agent*. The CSN Sensor Network Middleware supports the management of sensor networks and treats heterogeneous sensors in a uniform way. In CSN, each sensor is modeled to be a logical data stream and, in fact, implemented as a message queue. The CSN Sensor Agent is designed to manage heterogeneous sensors to behave according to the logical data stream model. The CSN Sensor Agent encapsulates sensor-specific details inside the logical model.

Major system components of the CSN Sensor Network Middleware are as follows:

(i) Sensor network manager: the CSN Sensor Network Manager (CSNM) provides the logical creation and destruction of both sensors and sensor networks. On the creation of a sensor, CSNM receives only metadata (including sensor-specific data such as the product model number) and semantic tags about the sensor. Then, CSNM assigns a unique ID to the sensor and creates a dedicated message queue for it. The creation of a sensor network follows the same procedure, but in addition, CSNM receives IDs for sensors to join the sensor network. Metadata is a collection of key-value pairs where both keys and values are text strings. Semantic tags are a list of keywords.

(ii) Message queue manager: the CSN Message Queue Manager (CMQM) handles message queues for sensors or sensor networks. A sensor data stream is treated as a message queue.

(iii) Messaging system: the CSN Messaging System is in fact a general-purpose messaging middleware (currently Apache ActiveMQ).

(iv) Coordinator: the CSN Coordinator manages all the system components and allows the system staff to administrate the CSN runtime system.

(v) Programming APIs: the Publish/Subscribe model-based Message Queue APIs are provided for application development.

(vi) Galilee monitor dashboard: we customized the CSN dashboard to manage environmental sensor network and its data delivery. Galilee Monitor Dashboard provides web-based user interfaces for its administration tasks: booting/shutdown for the runtime system, the management of sensors and sensor networks, monitoring sensor data streams, and searching sensor networks by ID and semantic tags.

(vii) Sensor agent: the CSN Sensor Agent reads data from a sensor and sends it to applications or data management systems such as a sensor data repository. More detailed explanation about the CSN Sensor Agent is given in Section 4.3.

Figure 5 illustrates how a sensor is created and its data is delivered to an application in CSN. Major operations which are numbered in both the figure and the following descriptions for readability are as follows:

(1) Booting the CSN runtime system: the user (an environmental scientist or a system administration staff) starts the CSN Coordinator by the Galilee Monitor Dashboard. Then, the CSN Coordinator initializes the CSN Sensor Network Manager, the CSN Message Queue Manager, and the CSN Messaging System.

(2) Registering a new sensor: when the user registers a sensor by the Galilee Monitor Dashboard, the user provides the Galilee Monitor Dashboard with metadata and semantic tags. Then, the CSN Sensor Network Manager creates an entry for the new sensor in the system table so that the sensor can be searched with metadata and semantic tags later. In addition, the CSN Message Queue Manager also creates a message queue for the sensor in the CSN Messaging System.

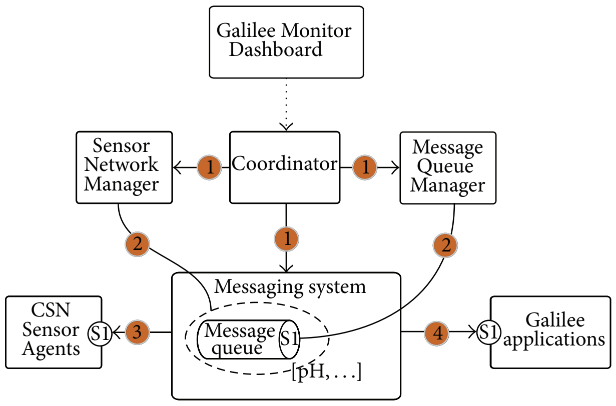


Figure 5: CSN runtime system.

Table 1: Data format for sensor data records.

Attributes	Data type	Contents
ID	Integer	Sensor network ID assigned by the CSN system
Timestamp	Datetime	Timestamp when the sensor data is created
Value	String	Measured value from sensor

(3) Starting a sensor: the CSN Sensor Agent is installed on the new sensor, configured to publish data to the message queue (in other words, topic), and started to run the sensor.

(4) Developing an application: an application is designed to subscribe the message queue. The format of sensor data is shown in Table 1.

4.3. CSN Sensor Agent with the Cellular Communication Support. In order for a sensor to be used as a sensor node of a sensor network, the hardware sensor and additional hardware components are required to be integrated into the sensor node. In sensor networks, a sensor node usually consists of four hardware components: sensing unit, computing unit, communication unit, and power unit. The sensing unit is a hardware sensor that measures a certain property about the environment such as temperature and humidity. In most cases, such sensors are usually passive devices and cannot process data or communicate with a sensor data repository. The computing unit is a kind of small-sized embedded computer to receive and process analog signals or digital data from sensors. The communication unit allows the computing unit to communicate and send sensor data to sensor data repositories by wired or wireless communication. The power unit is a battery or connected to the energy grid or solar panels and provides the other units with required energy. Figure 6 shows the structure of a sensor node.

The CSN Sensor Agent runs on the computing unit of a sensor node and provides a uniform interface for different types of sensors. The CSN Sensor Agent consists of two components: Device-Specific Sensor Library and Uniform Sensor Library. The Device-Specific Sensor Library manages

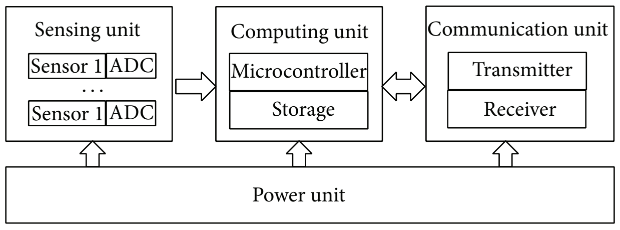


Figure 6: Sensor node architecture.

Table 2: Sensing range of sensors.

Sensor type	Range	Unit	Example
Water temperature	−5–50	°C	24.25
Dissolved oxygen	0–30	mg/L	10.9
Conductivity	0–100	mS/cm	50.29
pH	0–14	—	6.7
Chlorophyll-a	0–50	μg/L	2.98
Voltage	0–12	V	10.0

the sensing unit and encapsulates device-specific features. The Uniform Sensor Library provides the CSN-compatible interface to the CSN runtime system and also manages the communication unit.

In the Galilee system, the CSN Sensor Agent is extended to support cellular communication. For this extension, the sensor node requires the communication unit to communicate with cellular communication networks (i.e., base stations). We used two types of devices as the communication unit: a cellular communication modem and an embedded computer with a cellular communication modem. Specifically, two products are used as the communication unit: MR-2100s (from MARTNER, Inc.) and MPT-800 (MELPER). MR-2100s is just a communication modem, but the MPT-800 product contains both a cellular communication modem and a built-in embedded computer with Linux.

In the case of MPT-800, we ported a simplified version of the CSN Sensor Agent to the built-in embedded computer. For MR-2100s, the computing unit is developed by porting the CSN Agent to the Raspberry Pi system with the Arduino hardware board. Figure 7 shows MPT-800 and MR-2100.

4.4. Event-Driven Sensor Maintenance Support System. In the current design, the Galilee system uses the *complex event processing engine* called Esper as the Event-Driven Sensor Maintenance Support System (ESMS). Event processing is used to facilitate the systematic detection of faults, corruption, or abnormal behaviors of sensors. Main features of the Esper system are as follows:

(i) Event stream management: the Esper system is designed to receive and examine event streams (data streams).

(ii) Event processing language (EPL): algorithms to detect various problems with sensors are programmed in EPL. Such algorithms specify event types of interest or event sequence patterns in an event stream. Such EPL code is called a statement.

(iii) Event detection: once data streams and EPL statements are created, the Esper engine continues to examine data streams to detect occurrences of events or event sequences matching event types or event sequence patterns specified in EPL statements.

(iv) Event stream monitoring: the Esper administration tool displays current event streams in various charts.

(v) Event notification: when the Esper engine detects occurrences of events or event sequences matching event types or event sequence patterns specified in EPL statements, it can send notification messages to the prespecified list of people by email.

Currently, simple error detection techniques are programmed in EPL and tested by the Esper engine. For example, one of the simplest methods is to check if values from a sensor belong to its hardware value range. Table 2 shows the range of possible values from the water quality sensor (Hydrolab MS5). If the Esper finds that a value from the sensor is out of the ranges, then it can decide that the sensor must have some faults.

The Esper engine is integrated with the CSN Sensor Network Middleware and the S4EM Sensor Data Repository as follows:

(i) Data streams from sensors in the CSN system are connected to event streams in Esper.

(ii) Detected events from the Esper engine are stored as data streams from the special sensor (called Esper Sensor) in the S4EM Sensor Data Repository. That is, the Esper engine is treated as a special type of sensor in the Galilee system.

Figure 8 illustrates how errors in sensor can be detected by event processing. An EPL statement to specify a sensor error condition is associated with an event stream and the Esper engine continuously examines the event stream to detect the event types or sequence patterns matching the condition.

4.5. Cloud-Based Sensor Data Repository. For the management of environmental data from sensors, we developed the S4EM Sensor Data Repository System as a SaaS service on a cloud database PaaS service (Google Datastore). An earlier prototype version of the S4EM system was presented to a conference [4]. The S4EM system is designed as follows:

(i) The SaaS service supports an environmental observation data model called Vega that is a variant of the CUAHSI ODM [12].

(ii) The SaaS service is available as an online service. Scientists who have little knowledge or experiences about information technology can use it for their sensor data, only with some initial efforts for Sensor Agents installation and service configuration.

(iii) The SaaS service is guaranteed to be autonomously available with almost no downtime, to be scalable, and to be secured.

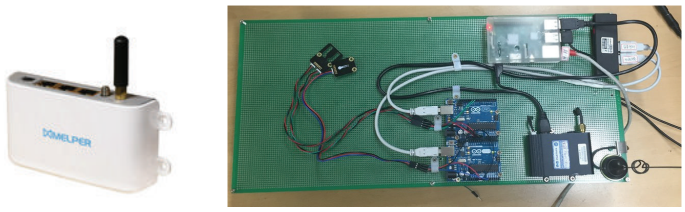


Figure 7: Cellular communication modems: MPT-800 and Raspberry Pi with MR-2100.

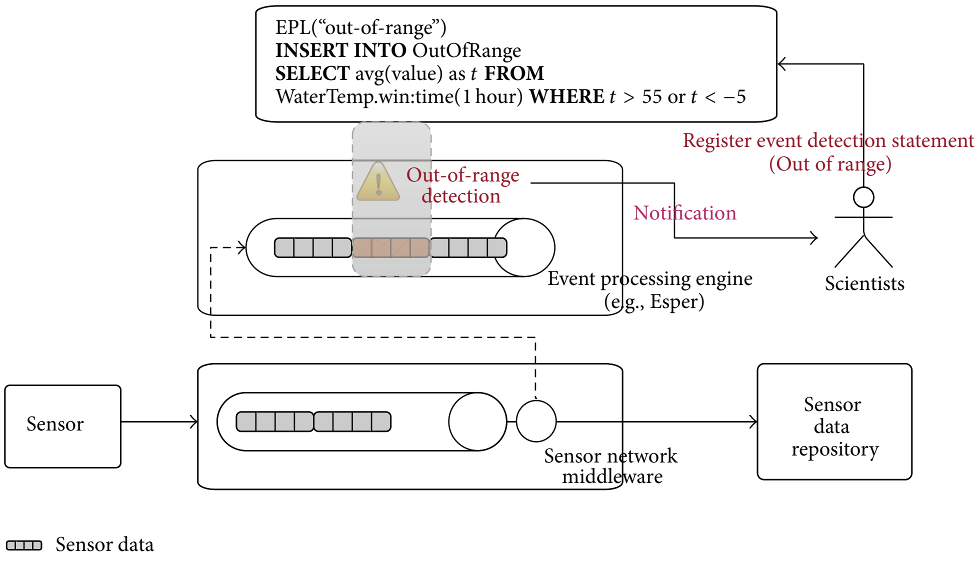

EPL("out-of-range")
INSERT INTO OutOfRange
SELECT avg(value) as t FROM
WaterTemp.win:time(1 hour) WHERE t > 55 or t < −5
Register event detection statement (Out of range)
Out-of-range detection
Notification
Event processing engine (e.g., Esper)
Scientists
Sensor
Sensor network middleware
Sensor data repository
Sensor data

Figure 8: Event-Driven Sensor Maintenance Support System.

Vega is a data model developed by the GLEON (Global Lake Ecological Observatory Network) community [22] and is similar to the O&M and CUAHSI ODM standard. In Vega, the main modeling concept is *Stream* while it is *Observation* in O&M. *Stream* in Vega is intended for a sequence of observation instances, but *Observation* is for a single instance of observation. Figure 9 shows major data entities of the Vega model:

(i) Stream: the *Stream* entity represents an instance of data streams (i.e., a series of observation values) from a sensor. A sensor may generate multiple data streams (multiple instances of the Stream entity). A sensor may be considered to produce a new data stream when it is reconfigured or recalibrated.

(ii) Variable: a *Variable* entity contains metadata about a property for which environmental monitoring is carried out.

(iii) Value: a *Value* entity contains an actual measurement value from a sensor.

(iv) Site: a *Site* entity contains metadata about a site where monitoring is carried out.

(v) Source: a *Source* entity contains metadata about the institute or the researcher who carries out the monitoring.

4.6. Data Portal. In addition to the SaaS service, the Galilee system provides the Galilee Data Portal for accessing streams of sensor data on the cloud-based sensor data repository. The Galilee Data Portal provides the following features:

(i) Stream management: in our system, *Stream* is a major data entity which has a unique ID and is associated with most other data entities. It must be created before sensor data is inserted into the system. There are two ways to create *Stream*. First, the system administrator creates a *Stream* data object for a sensor before the sensor generates in a real time manner. Second, a scientist creates a *Stream* data object when he or she uploads a series of sensor data in a batch style.

(ii) Online sensor data upload: each sensor is associated with a *Stream* object. A sensor sends a series of sensor data records to the Sensor Data Repository, one record at a time. The Sensor Agent attaches the ID of its associated Stream instance to each record. When the server receives a sensor data record, it creates a new *Value* object with a *timestamp*, a measured value, and a *streamID* data from the record. The *Value* object is inserted into Google Datastore.

(iii) Data search: scientists can search sensor data by specifying matching conditions against *Stream*,

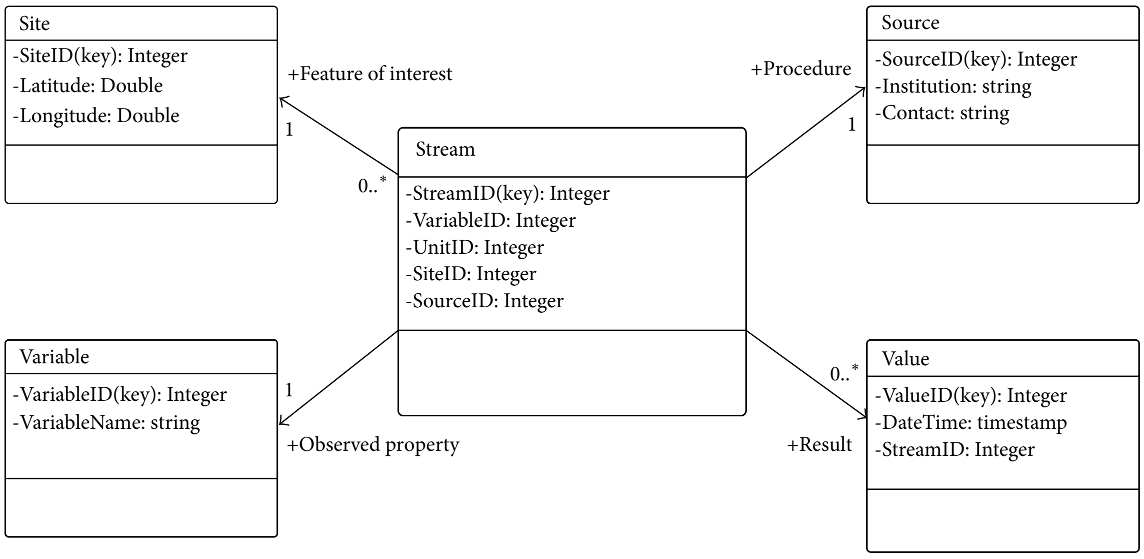


Figure 9: System data model diagram.

Variable, *Site*, and time intervals. Search results can be displayed as plots or tables. In addition, those results can be downloaded as CSV files that can be accepted by spreadsheet programs.

5. Implementation

5.1. Messaging System-Based System Integration. In the development of the Galilee system, we intended to avoid the implementation from scratch as much as possible and to focus on the seamless integration of system middleware: CSN, S4EM, and Esper. Due to the data stream-based system model, the integration is simple and straightforward as follows:

(i) Data streaming-based integration: all of these components are designed to manage data streams. These data streams are connected via streaming. For example, data records from a sensor in the CSN system are streamed into the corresponding data stream object in the S4EM system. In the same way, the CSN system is connected to the Esper system by data streaming.

(ii) Messaging system-based integration: data streaming is implemented by the messaging system called ActiveMQ [25]. The ActiveMQ messaging system provides message queue's or topics for data streaming and the Application Programming Interface (API) based on the Publish/Subscribe model.

Figure 10 illustrates the messaging system-based system integration. For example, when a sensor in CSN sends data to S4EM, a message queue's (or topic) is first created for data streaming in the ActiveMQ runtime system. Then, the sensor in CSN continues to publish its data stream to the message queue's and S4EM reads the data stream by subscribing the message queue's. During this interaction, both CSN and S4EM use the Publish/Subscribe model for communication.

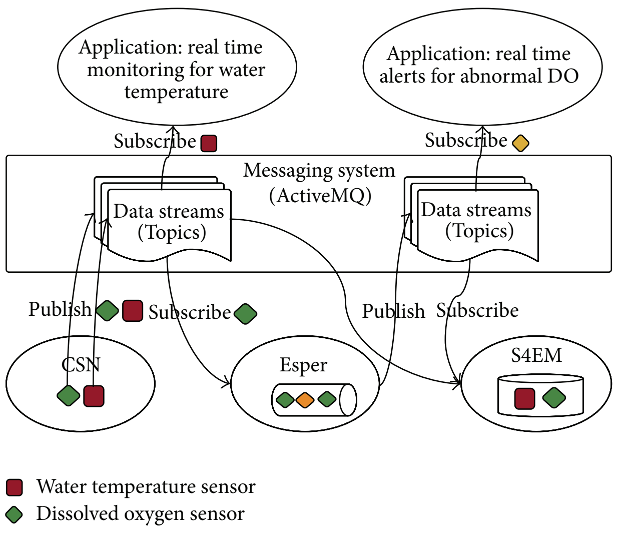

Application: real time monitoring for water temperature
Application: real time alerts for abnormal DO
Subscribe
Subscribe
Messaging system (ActiveMQ)
Data streams (Topics)
Data streams (Topics)
Publish
Subscribe
Publish
Subscribe
CSN
Esper
S4EM
🟥 Water temperature sensor
◆ Dissolved oxygen sensor

Figure 10: The system integration.

In addition, various applications can receive data from the sensor in a real time manner by subscribing the message queue's.

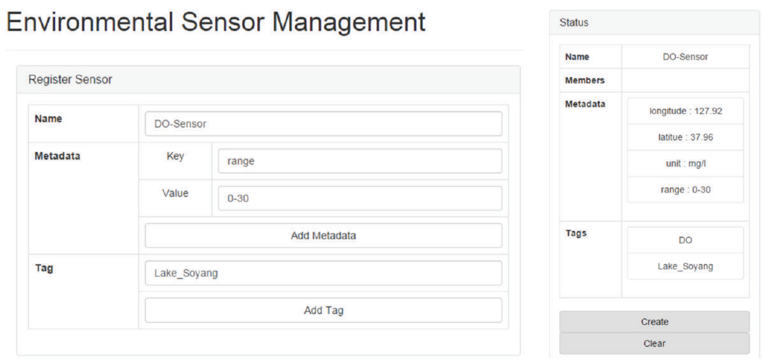
5.2. Management of Environmental Monitoring Infrastructure. The Galilee Monitor Dashboard allows the user to run CSN Sensor Network Middleware easily and conveniently. The dashboard provides the web-based simple and intuitive user interface for most CSN features. Major administration operations available at the web-based user interface are as follows:

(i) Start and shut down the CSN runtime system.

(ii) Manage sensors and sensor networks.

(iii) Search sensors and sensor networks by semantic tags.

(iv) Monitor the status of message queues.


Environmental Sensor Management

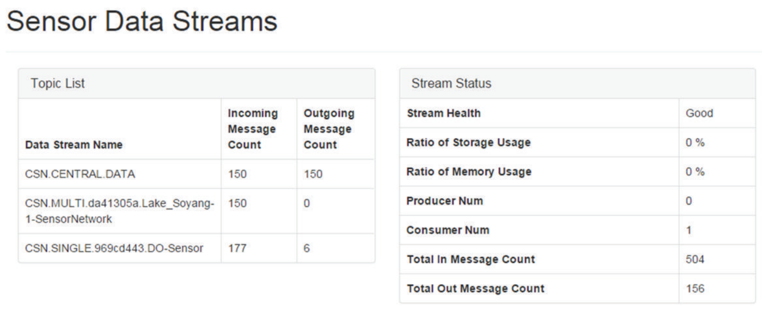
Register Sensor

Name	DO-Sensor	
Metadata	Key	range
	Value	0-30
	Add Metadata	
Tag	Lake_Soyang	
	Add Tag	

Status

Name	DO-Sensor
Members	
Metadata	longitude : 127.92
	latitue : 37.96
	unit : mg/l
	range : 0-30
Tags	DO
	Lake_Soyang

Figure 11: Management window for sensor networks.


Sensor Data Streams

Topic List

Data Stream Name	Incoming Message Count	Outgoing Message Count
CSN.CENTRAL.DATA	150	150
CSN.MULTI.da41305a.Lake_Soyang-1-SensorNetwork	150	0
CSN.SINGLE.969cd443.DO-Sensor	177	6

Stream Status

Stream Health	Good
Ratio of Storage Usage	0 %
Ratio of Memory Usage	0 %
Producer Num	0
Consumer Num	1
Total In Message Count	504
Total Out Message Count	156

Figure 12: Status of sensor data streams in messaging system.

Figure 11 shows a snapshot of the user interface window (in fact, only a relevant part of the window for the sake of clarity) for the management of sensor networks. This window consists of two parts: registration (left subwindow) and status (right subwindow). The creation of a new sensor is composed of two steps: (1) enter the name, metadata (a list of key-value pairs), and semantic tags in the registration subwindow and (2) commit by clicking on the "Create" button in the status window. A number of key-value pairs or semantic tags can be entered repeatedly by the "Add Metadata" and "Add Tag" buttons.

Figure 12 shows a snapshot of the status window (in fact, only a relevant part of the window for the sake of clarity) for message queues. The window displays the list of currently operating message queues (e.g., "CSN.SINGLE .969cd443.DO-Sensor") and its status information.

5.3. Event-Driven Sensor Maintenance Support System. The Galilee Event-Driven Sensor Maintenance Support System that is currently the Esper complex event processing engine allows the user to specify sensor error conditions in EPL. EPL provides a rich set of powerful features but is simple and easy to use for simple event programs (called statements). An example of EPL statement is given in Program 1. Please refer to the online Esper reference documentation for the syntax of EPL available at http://www.espertech.com/.

The Esper engine can be configured to send an alert message to a predefined list of addresses by email. Figure 13 shows an example of an alert email message sent by the Esper engine.

5.4. Experiments with Water Quality Monitoring Datasets at Lake Soyang. We conducted some experiments with the datasets that were collected by a simple environmental monitoring system (called mini-Galilee) at Lake Soyang from 2011 to 2012. The mini-Galilee system consisted of the water quality sensing device called Hydrolab MS5 that contained five sensors: water temperature, DO, conductivity, pH, and chlorophyll-a. The mini-Galilee system integrated the sensor system with the MPT-800 device with a 3G cellular communication modem and an embedded computer in order to communicate with the backend data management server. The mini-Galilee system also used a simple relational database for the management of sensor data.

Serious drawbacks with the mini-Galilee system were as follows:

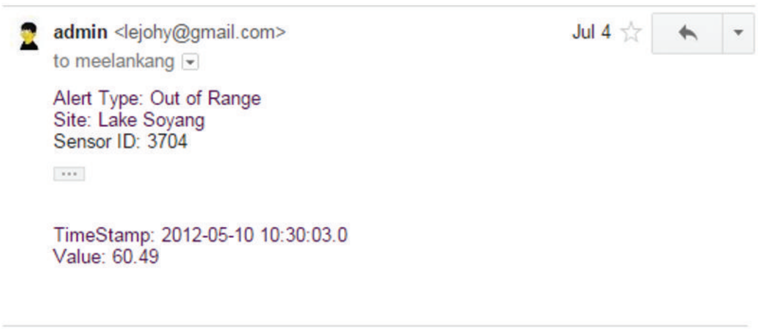
(i) No support for sensor networks: the mini-Galilee system was designed for a single sensor and difficult to extend for the addition of new sensors.

(ii) No support for explicit sensor management in the sensor data repository: the mini-Galilee system used a relational database system to contain the sensor data. However, the database system did not handle sensors or data streams as explicit data objects. It complicated the management of sensors because sensors were not clearly identified in databases.

(iii) No support for the messaging system between sensors and the database system: a socket-based communication system was used to deliver data from sensors

```
@Name("Sensor Data")
create schema sensor_data as (id string, timestamp string, value double)
@Name("Water Temperature Error")
select * value from sensor_data (id = "3704") where value >50 or value <−5
```

Program 1: EPL statements for the detection of "out-of-range" error for water temperature.


admin <lejohy@gmail.com>
to meelankang
Jul 4

Alert Type: Out of Range
Site: Lake Soyang
Sensor ID: 3704
...

TimeStamp: 2012-05-10 10:30:03.0
Value: 60.49

Figure 13: Alert email message.

to the database system. The interaction was implemented in an ad hoc way and difficult to modify.

(iv) No support for the automatic detection of faults or abnormal behaviors in sensors: the maintenance checkup for the sensor was carried out biweekly. When there was a problem with the sensor, it was impossible to detect it until the next maintenance checkup.

Those drawbacks are usually found in many simple environmental monitoring systems. However, these problems are generally difficult to deal with in a systematic way. Our experiment with the datasets at Lake Soyang showed that the current prototype of the Galilee system could alleviate those drawbacks significantly.

Currently, the Galilee Data Portal with the datasets is available to the public at the site (http://gaebasedkleon.appspot.com/) and supports the following:

(i) Simple searching services for sites, variables, and sensor data.

(ii) Data download services.

(iii) Visualization services for sensor data.

Figure 14 shows a snapshot of the data portal window. In the chart, the two lines show water temperature and DO values from February 23, 2012, to February 28, 2012.

6. Related Work

RTEM has been widely used for many real world applications in both various scientific domains and public sectors [8–11]. Therefore, there are a number of RTEM services or infrastructures available and operating. However, RTEM systems for those services or infrastructures are usually developed and customized for their specific application domains. Therefore, such RTEM systems are difficult to modify or to extend for new requirements. As a result, different RTEM systems need to be developed for different applications.

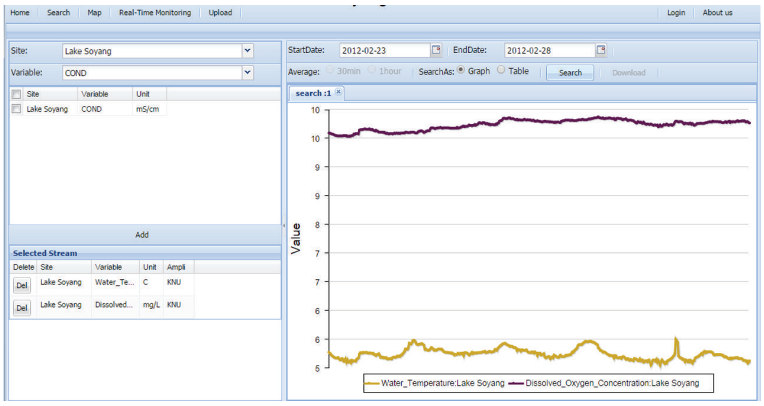
On the other hand, the Galilee RTEM system is based on the data stream-based distributed system model and designed to be general system middleware applicable to a variety of RTEM applications. Therefore, the Galilee system can allow us to develop RTEM applications for various domains without a significant amount of development effort.

There are active research activities on Sensor Cloud to integrate wireless sensor networks and cloud computing [17]. Sensor Cloud addresses important technical issues and has great potentials for future IoT applications. However, Sensor Cloud is not a concrete technical solution but a computing model or concept. It is a very complicated technical approach. However, Galilee is the system middleware intended for real world applications and focused on RTEM.

OGC's Sensor Web Enablement (SWE) includes open interfaces, data standards, and service models for a variety of sensor related applications [26]. There are reference implementations such as 52°North and PULSENet [27, 28]. Although the SWE technology is comprehensive and addresses challenging issues in managing sensors and their data, SWE requires application developers to understand a significant amount of technical knowledge and system administrators to manage a substantial scale of system infrastructure. On the other hand, the Galilee system is a very lightweight approach based on the simple data stream-based distributed system model.

Integrating the complex event processing (CEP) technology into wireless sensor networks is recently an active research area [29] where various promising ideas and techniques are being studied. In Galilee, CEP is applied only for the specific issue: intelligent sensor maintenance.

There are a number of cloud databases available [30–32]. Currently, the Galilee system uses Google Datastore for the



Site: Lake Soyang
Variable: COND

Site	Variable	Unit
Lake Soyang	COND	mS/cm

StartDate: 2012-02-23 EndDate: 2012-02-28
Average: ( ) 30min ( ) 1hour SearchAs: [✓ Graph] ( ) Table

Selected Stream

Delete	Site	Variable	Unit	Ampli
Del	Lake Soyang	Water_Te...	C	KNU
Del	Lake Soyang	Dissolved...	mg/L	KNU

Figure 14: Data portal window.

Sensor Data Repository because it is a PaaS service. However, Google Datastore can be easily replaced by another NoSQL database such as MongoDB on cloud infrastructures because the dependency of the Galilee system on Google Datastore is limited.

7. Conclusions and Future Work

Sensor-based RTEM is crucial for environmental research because it allows scientists to observe and study dynamic, rare, or abrupt phenomena in the environment that are almost impossible to capture by human-involved sporadic or periodic monitoring. Due to technical advances in sensors, wireless communication, and embedded computing, sensor-based RTEM now becomes an effective and important approach to environmental sciences.

However, due to the lack of a well-defined distributed system model for RTEM, conventional RTEM systems are usually developed and customized in a domain-specific way. Therefore, it is usually challenging to build RTEM systems, to apply them for other different applications, to extend them for new requirements, and to integrate them with other application systems.

In this paper, we first proposed a simple distributed system model for RTEM intended to simplify and to conceptualize the design of RTEM systems for various application domains. We explained how every system component and their interactions in RTEM systems are simply modeled according to the concept of data streams in this system model. Furthermore, we discussed how the simple data stream-based approach enables the designs of major system components and their integrations to be conceptual and independent of application characteristics. Therefore, the data stream-based distributed system model facilitates the development of general RTEM middleware applicable for a variety of applications.

We also presented the RTEM middleware called Galilee whose design is based on the data stream-based distributed system model. The Galilee system consists of three middleware systems: the sensor network middleware called CSN, the sensor data repository called S4EM, and the complex event processing engine called Esper. We showed how the designs of these system components were logically specified and understood according to the data stream-based system model. Furthermore, most design features are independent of application characteristics. As a result, the Galilee middleware system for RTEM is simple and intuitive to understand and to manage. It is also reliable and extensible to support a variety of RTEM applications.

We tested the current prototype implementation of the Galilee system with the datasets that were collected by a simple environmental monitoring system (called mini-Galilee) at Lake Soyang from 2011 to 2012. Based on our experiences with the current Galilee prototype implementation, we expect that the Galilee system will facilitate the development and maintenance of various RTEM application systems significantly once it is fully developed.

The Galilee project is at an early stage and therefore there will be lots of future work. First, we plan to test the Galilee runtime system for real world field monitoring in the Weather Information Service Engine (WISE) project [33]. Technically, we will add security supports to the Galilee runtime system because the current Galilee implementation has little support for security. In addition, more advanced administration services will be added in the near future.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This work was funded by the Weather Information Service Engine (WISE) project under Grant KMIPA-2012-0001. The authors would like to thank the PRAGMA community (http://www.pragma-grid.net/) and the GLEON community (http://www.gleon.org/) for sharing their knowledge and experiences with them.

References

[1] J. Artiola, I. Pepper, and M. Brusseau, *Environmental Monitoring and Characterization*, Elsevier Academic Press, Amsterdam, The Netherlands, 2004.

[2] W. Joe, J. Lee, and K. Jeong, "CSN: the conceptually manageable sensor network," *International Journal of Distributed Sensor Networks*, vol. 2015, Article ID 720861, 17 pages, 2015.

[3] R. Bhargavi, V. Vaidehi, P. T. V. Bhuvaneswari, P. Balamuralidhar, and M. Girish Chandra, "Complex event processing for object tracking and intrusion detection in wireless sensor networks," in *Proceedings of the 11th International Conference on Control, Automation, Robotics and Vision (ICARCV '10)*, pp. 848–853, Singapore, December 2010.

[4] Z. Cui, M. Jiang, K. Jeong, and B. Kim, "A cloud database service approach to the management of sensor data," in *Proceedings of the 5th International Conference on Information Science and Applications (ICISA '14)*, pp. 1–4, Seoul, Republic of Korea, May 2014.

[5] P. Corke, T. Wark, R. Jurdak, W. Hu, P. Valencia, and D. Moore, "Environmental wireless sensor networks," *Proceedings of the IEEE*, vol. 98, no. 11, pp. 1903–1917, 2010.

[6] G. Barrenetxea, F. Ingelrest, G. Schaefer, and M. Vetterli, "Wireless sensor networks for environmental monitoring: the sensorscope experience," in *Proceedings of the International Zurich Seminar on Communications (IZS '08)*, pp. 98–101, IEEE, Zurich, Switzerland, March 2008.

[7] J. K. Hart and K. Martinez, "Environmental sensor networks: a revolution in the earth system science?" *Earth-Science Reviews*, vol. 78, no. 3-4, pp. 177–191, 2006.

[8] Y. Xue, B. Ramamurthy, and M. Burbach, "A two-tier wireless sensor network infrastructure for large-scale real-time groundwater monitoring," in *Proceedings of the 35th Annual IEEE Conference on Local Computer Networks (LCN '10)*, pp. 874–881, IEEE, Denver, Colo, USA, October 2010.

[9] M. A. Nasirudin, U. N. Za'bah, and O. Sidek, "Fresh water real-time monitoring system based on wireless sensor network and GSM," in *Proceedings of the 2nd IEEE International Conference on Open Systems (ICOS '11)*, pp. 354–357, IEEE, Langkawi, Malaysia, September 2011.

[10] E. R. Vivoni and R. Camilli, "Real-time streaming of environmental field data," *Computers & Geosciences*, vol. 29, no. 4, pp. 457–468, 2003.

[11] X. Song, C. Wang, M. Kagawa, and V. Raghavan, "Real-time monitoring portal for urban environment using sensor web technology," in *Proceedings of the 18th International Conference on Geoinformatics*, pp. 1–5, IEEE, Beijing, China, June 2010.

[12] J. S. Horsburgh, D. G. Tarboton, D. R. Maidment, and I. Zaslavsky, "A relational model for environmental and water resources data," *Water Resources Research*, vol. 44, no. 5, 2008.

[13] W. K. Michener and J. W. Brunt, Eds., *Ecological Data: Design, Management and Processing*, John Wiley & Sons, 2009.

[14] G. Cugola and A. Margara, "Processing flows of information: from data stream to complex event processing," *ACM Computing Surveys*, vol. 44, no. 3, article 15, 2012.

[15] Jboss.org, "Drools—Drools—Business Rules Management System (Java, Open Source)," 2015, http://www.jboss.org/drools/droolsfusion.html.

[16] G. Thonet, P. Allard-Jacquin, and P. Colle, "ZigBee—WiFi coexistence," Schneider Electric White Paper and Test Report, 2008.

[17] S. K. Dash, J. P. Sahoo, S. Mohapatra, and S. P. Pati, "Sensor-cloud: assimilation of wireless sensor network and the cloud," in *Advances in Computer Science and Information Technology. Networks and Communications*, vol. 84 of *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, pp. 455–464, Springer, Berlin, Germany, 2012.

[18] C. D. M. A. Qualcomm, Technologies. Advanced Modem Receivers-System Level Performance , internal documentation, 2008.

[19] R. C. Cazan, L. V. Chirila, and A. Marichescu, "Remote terminal unit's Intelligence Evolution," in *Proceedings of the IEEE International Conference on Automation, Quality and Testing, Robotics (AQTR '08)*, vol. 3, pp. 300–303, IEEE, Cluj-Napoca, Romania, May 2008.

[20] S. Cox, "Observations and measurements," Open Geospatial Consortium Best Practices Document, Open Geospatial Consortium, 2006.

[21] L. A. Winslow, B. J. Benson, K. E. Chiu, P. C. Hanson, and T. K. Kratz, "Vega: a flexible data model for environmental time series data," in *Proceedings of the Environmental Information Management Conference*, pp. 10–11, Albuquerque, NM, USA, September 2008.

[22] T. K. Kratz, P. Arzberger, B. J. Benson et al., "Toward a global lake ecological observatory network," *Publications of the Karelian Institute*, vol. 145, pp. 51–63, 2006.

[23] M. Hamdaqa and L. Tahvildari, "Cloud computing uncovered: a research landscape," *Advances in Computers*, vol. 86, pp. 41–85, 2012.

[24] W. Voorsluys, J. Broberg, and R. Buyya, "Introduction to cloud computing," in *Cloud Computing: Principles and Paradigms*, pp. 1–44, Wiley, 2011.

[25] Activemq.apache.org, Apache ActiveMQ—Index, October 2015, http://activemq.apache.org/.

[26] M. Botts, G. Percivall, C. Reed, and J. Davidson, "OGC sensor web enablement: overview and high level architecture," in *GeoSensor Networks*, vol. 4540 of *Lecture Notes in Computer Science*, pp. 175–190, Springer, Berlin, Germany, 2008.

[27] A. Sliwinski, I. Simonis, A. Remke, U. Streit, and A. Wytzisk, "Boosting the OGC sensor web enablement initiative by open source web services—the case of 52° North," in *Symposium flir Angewandte Geographische lnformationstechnologie (AGIT '05)*, Salzburg, Austria, July 2005.

[28] S. M. Fairgrieve, J. A. Makuch, and S. R. Falke, "PULSENet: an implementation of sensor web standards," in *Proceedings of the International Symposium on Collaborative Technologies and Systems (CTS '09)*, pp. 64–75, IEEE, Baltimore, Md, USA, May 2009.

[29] J. Dunkel, "On complex event processing for sensor networks," in *Proceedings of the International Symposium on Autonomous Decentralized Systems (ISADS '09)*, pp. 1–6, IEEE, Athens, Greece, March 2009.

[30] F. Ickert, M. D. Del Fabro, E. C. de Almeida, and S. Scherzinger, "NoSQL data model evaluation on app engine datastore," in *Proceedings of the 28th Brazilian Symposium on Databases*, pp. 1–6, Recife, Brazil, 2013.

[31] R. Cattell, "Scalable SQL and NoSQL data stores," *ACM SIGMOD Record*, vol. 39, no. 4, pp. 12–27, 2010.

[32] K. Kaur and R. Rani, "Modeling and querying data in NoSQL databases," in *Proceedings of the IEEE International Conference on Big Data*, pp. 1–7, Silicon Valley, Calif, USA, October 2013.

[33] WISE, http://www.wise2020.org/.