Storing and Visualizing Sensor Networks Data in Public Clouds

# Aysun Simitci

# Advisor - Munehiro Fukuda CSS 596 University of Washington Bothell

# 6/05/2012

# Introduction

The objective of this sensor data project is to store sensor data in public clouds and to convert raw sensor data into visual formats that can be easily understood by people. The end result should be to make the sensor networks friendlier to users and to create a flexible framework to meet the requirements of different applications.

Cloud computing is the next stage in evolution of the Internet. The cloud in cloud computing provides the means through which everything, from computing power to computing infrastructure, applications, business processes to personal collaboration, can be delivered to you as a service wherever and whenever you need.

Information visualization is at the crossroads between computer science and all other sciences that generate data. Visualization uses computer graphics techniques to take numeric data, collected or synthesized by users, and translates it into images. The images will allow important features in the data to be understood much more readily than from the raw data.

This project is part of the “[Sensor Cloud Integration](http://depts.washington.edu/dslab/SensorCloud/index.html)” research in the Distributed Systems Laboratory, University of Washington Bothell. The “Sensor Cloud Integration” research targets frost protection as an example application. Temperature prediction is the main concern of farmers for protecting their crops from frost damage, and takes advantage of several system components: reading sensor data, running the analysis programs remotely in the cloud, and delivering graphical outputs to remote users.

# Related Work

Existing research on sensor data collection and visualization focuses on using private data stores. Our research differs from them by using publicly available cloud resources.

Shiraishi (2007) focuses on client side processing of sensor data which can be useful if the data is stored in multiple separate databases. In my project, I have only a single data source, but this approach can be used to extend my project later to include multiple cloud databases.

Kim, Ceong, and Jung (2007) present methods for visualizing various data received from environmental sensors into forms of 2D, 3D graphs and animation to manage fish farms in local area. In our project, we collect and visualize sensor data from apple orchards which requires techniques very similar to this project.

Song, Wang, Kagawa, and Raghavan, (2010) present an Internet based urban environment observation system that can real-time monitor environmental changes of temperature, humidity, illumination or air components in urban area. Our project is very similar and instead of the urban sensors we use farm sensors.

Fan and Biagioni (2004) combine sensor databases with geographical information systems (GIS) to visualize sensor data on maps. They show how to color map segments based on sensor data and location. I can incorporate their coloring techniques to show temperature data on maps. Fan and Biagioni have built sensor networks which consist of a series of sensor and communication units ("pods") deployed to monitor rare plants or other endangered species. The environmental data, such as temperature, rainfall, and sunlight, around the plants are sent by the wireless sensor networks to a base station which is able to access the Internet.

Lee, Wang, and Ryu (2010) focus on spatiotemporal query processing and storage of historical data collected from wide area sensor network applications, such as ecology monitoring, environmental monitoring, climate monitoring, and so on. Sensor data is almost always collected as a stream of data. And, real time processing is required on the stream to find statistical information. This paper shows how a large stream of data can be processed efficiently by only storing when data changes. They also have examples of real time visualization screens which can be added to my project in the future.

The idea of uploading summary data to social networks can be used in my project in the future to send sensor information and alerts to sharing networks such as Facebook and Skydrive. Rahman, Saddik, and Gueaieb (2009) propose an open source framework, named SenseFace, which seamlessly incorporates a four-tier network including a Body Sensor Network (BSN), cellular network, Internet and an overlay network consisting of social networks, to pass sensory data from a mobile BSN to the overlay network.

# Sensor Data Cloud Storage

For storing sensor data, we have surveyed existing cloud storage alternatives. Among three clouds storage services (Blobs, Tables, Databases) studied in this project, tables and databases are more appropriate to store sensor data. Blob storage does not provide structured storage and query capabilities required for working with sensor data to create rich analysis and visualizations.

Cloud Databases like [SQL Azure](http://www.windowsazure.com/en-us/home/features/sql-azure/), [Google Cloud SQL](https://developers.google.com/cloud-sql/), and [Amazon RDS](http://aws.amazon.com/rds/) provide data-processing capabilities through queries, transactions and stored procedures that are executed on the server side, and only the results are returned to the app. Cloud provides compute scale that is far bigger than any private computer network. If you have an application that requires data processing over large data sets, then a cloud database is a good choice.

Cloud table storage services like Windows Azure Table Storage and Amazon SimpleDB make sense if you don’t need a relational store or access is limited to a single table at a time and doesn’t require joins. If you have an app that stores and retrieves (scans/filters) large datasets but does not require data processing, then Table Storage is a better choice.

If you have more data than the maximum amount supported by SQL Azure (which is currently 150GB for a single instance), a better option is table storage similar to Windows Azure Table Storage or Amazon SimpleDB. Note that size limitation can be overcome with some data partitioning, but that could drive up the SQL Azure costs. The same space in Windows Azure Table Storage would probably be less expensive and has partitioning built-in by a declared partition key.

Figure 1 shows the price comparison between SQL Azure and Amazon RDS for different database sizes. The prices are taken from SQL Azure and Amazon websites in January 2012. For sizes up to 9GB, SQL Azure provides a cheaper service.

Figure : Cloud Database Price Comparison (January 2012)

Table 1 shows comparison of features between SQL Azure and Amazon RDS. Google Cloud SQL supports only 10GB database sizes and is still experimental.

Table : Feature Comparison of Cloud Databases

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Backup and Restore | Size Scaling | Database Size | Disaster Recovery | Pricing |
| SQL Azure | Needs manual setup | No scaling | Up to 150GB | Second replica is free | Cheaper for sizes up to 10GB |
| Amazon RDS | 8 days backup | Scale up by using larger VMs | Up to 1TB | You have to buy the second replica | Cheaper for larger sizes |

Storing and processing some sensor data will require complex queries and relationships. For example, a complex query might need access to temperature, humidity, light, image, and mapping data to create complex analysis. In addition, the amount of storage space required for sensor data should be within the limits of cloud databases. If not, table storage might be the next best choice but it will require special applications for data analysis.

In summary, storing and processing sensor data in the cloud has many advantages:

* Costs efficiency: Using public cloud databases eliminates server installation and maintenance costs. Any required software updates are done by the cloud service provider.
* Rapid prototyping and development: Since all the necessary server applications and tools are already installed and running, development with public clouds can proceed much faster.
* Global secure access: Security and fast network access is provided by the cloud service provider who might have data centers in every geographical region around the world. Cloud providers also replicate data across different geographical regions to provide data reliability and availability during disasters.

# Sensor Cloud Gateway

In this project, I have designed and implemented the Sensor Cloud Gateway. It is a tool that gathers real time sensor data and forwards it to cloud databases. The input to the gateway is sensor data in text or XML form. So, it can handle many different sensor types. The output is forwarded to SQL Azure cloud database. In the following sections the design and implementation details will be discussed.

## Design and Architecture

Sensor Cloud Gateway utilizes the [Sensor Server and Connector sensor frameworks](http://depts.washington.edu/dslab/SensorCloud/report/connectorManual.pdf), developed by the [Distributed Systems Laboratory](http://depts.washington.edu/dslab/). Figure 2 shows the general design of Sensor Cloud Gateway architecture.



Figure : Sensor Cloud Gateway Architecture

As Figure 2 shows, data is generated by wireless sensors that are grouped into sensor networks. A [wireless sensor network](http://en.wikipedia.org/wiki/Sensor_network) (WSN) consists of spatially distributed autonomous sensors to monitor physical or environmental conditions, such as temperature, sound, vibration, pressure, humidity, motion or pollutants and to cooperatively pass their data through the network to a main location. Sensor networks can be positioned in multiple locations. Real time sensor data is first collected by the sensor server, which is the temporal local store for sensor data. Sensor Cloud Gateway uses the Connector sensor framework and APIs to access data from the Sensor Server. Sensor Cloud Gateway also uses JDBC SQL driver to connect to the SQL Azure database. The sensor data in the database is used to analyze and visualize historical and real-time data.

## Database Design

The sensor cloud database is implemented in Microsoft SQL Azure. Database structures are configured using the Azure management portal web page. Our demonstration sensor network has only temperature sensors. So, the database has one table for temperature data only. A more complex sensor network would have many sensor types with correlated data that will require multiple tables.

Figure 4 shows data fields of the Sensor Data Table. These are the record fields generated by the sensors and collected by the Sensor Server.

The data fields represent:

* MacAddress: Sensor network ID
* TempF: Temperature in Fahrenheit
* RSSI: Received signal strength indication
* Battery: Battery voltage in Volts
* Period: Time between data samples
* Time: Time stamp when the data was sampled

These fields are representative for a temperature sensor. However, these tables can be modified to store all kinds of sensor data. That will require additional tables for different data types. An alternative design is to store sensor data in key-value stores initially, and to move the data to a database in post-processing. That will require additional steps and was not in the scope of my current implementation.

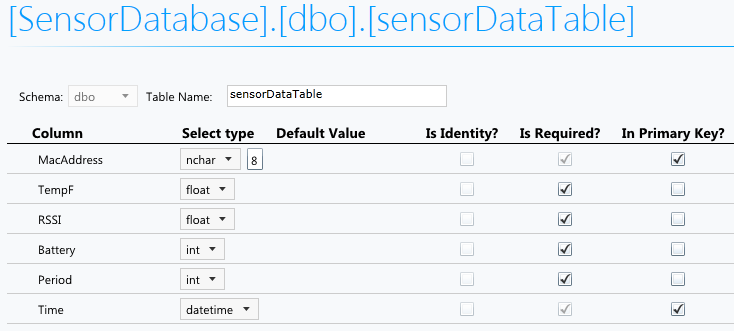


Figure : Sensor Data Table Data Fields

Figure 5 shows the Sensor Information Table which can be used to store metadata about the sensor devices themselves, such as location information and alarm threshold that can be used to generate alarm messages. Such location information can be used to visualize sensor data on maps in the future. The data in this table is independent of the sensor type. Sensor specific information can be stored in separate tables.

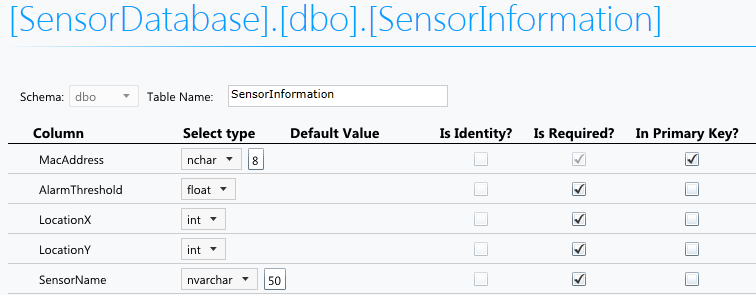
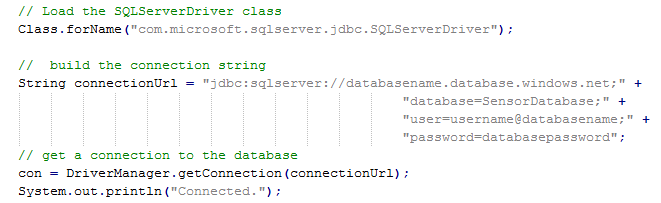


Figure : Sensor Information Table

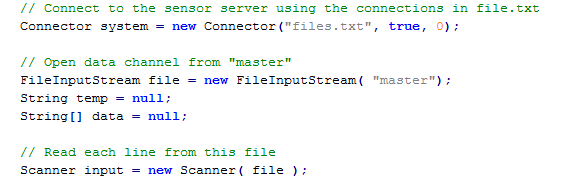
## Coding Tips and Highlights

In this section, highlights from Sensor Cloud Gateway code are presented. Some of these techniques will be useful for people who are going to implement sensor network and cloud integrations.

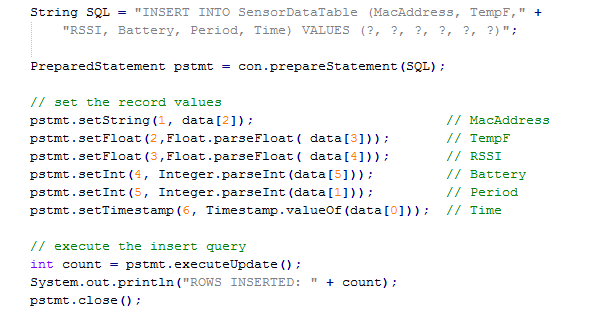
The following code excerpt shows how a connection to the SQL Azure can be made in Java using the JDBC SQL Driver. The connection string contains the database name, username, and password. Notice that, the connection to SQL Azure is exactly same as connecting to a SQL Server. Only the server addresses changes.



The next code sample shows how a connection to the Sensor Server can be made using the Connector class. Connector class simplifies communication to the sensor network by abstracting all data as a stream.



The next code excerpt shows how SQL Prepared Statement class can be used to construct a SQL INSERT query statement to insert a sensor a record to the SQL Azure database using the connection obtained above.



The sensor record fields are set one by one and then the query statement is executed and the number of rows inserted into the database is printed.

# Sensor Data Visualization and Analysis

Figure 6 shows a sample snapshot of the sensor data in the SQL Azure database. Through the Azure portal, you can see real time and historical sensor data. However, this data is in raw format and is difficult to analyze directly.

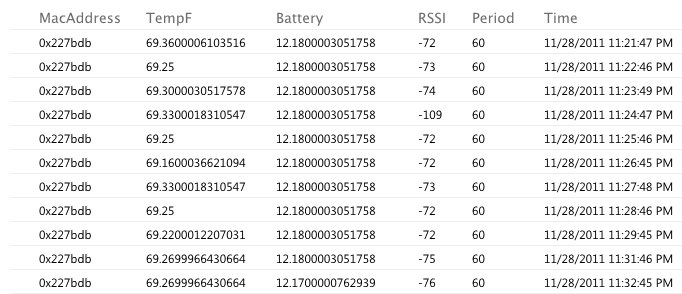


Figure : Sample Data from the SensorDatabase

The sensor software implementation is expanded to include sensor data processing and visualization running as cloud services ([SQL Azure Reporting Services](http://msdn.microsoft.com/en-us/library/windowsazure/gg430130.aspx)). These services are based on the sensor data layout developed in this research. It has visualization capabilities to analyze the data at multiple resolutions and time frames. Existing sensor network research uses local databases and visualization tools (Kim, Ceong, and Jung (2007), Song, Wang, Kagawa, and Raghavan, (2010)). This research differs by utilizing public cloud databases and visualization tools running in the cloud to provide cost efficiency and faster development time.

SQL Azure provides Reporting Services to execute reports in the cloud. The results of these reports can be accessed using application programs or from web pages on desktop PCs or mobile devices (see Figure 7).

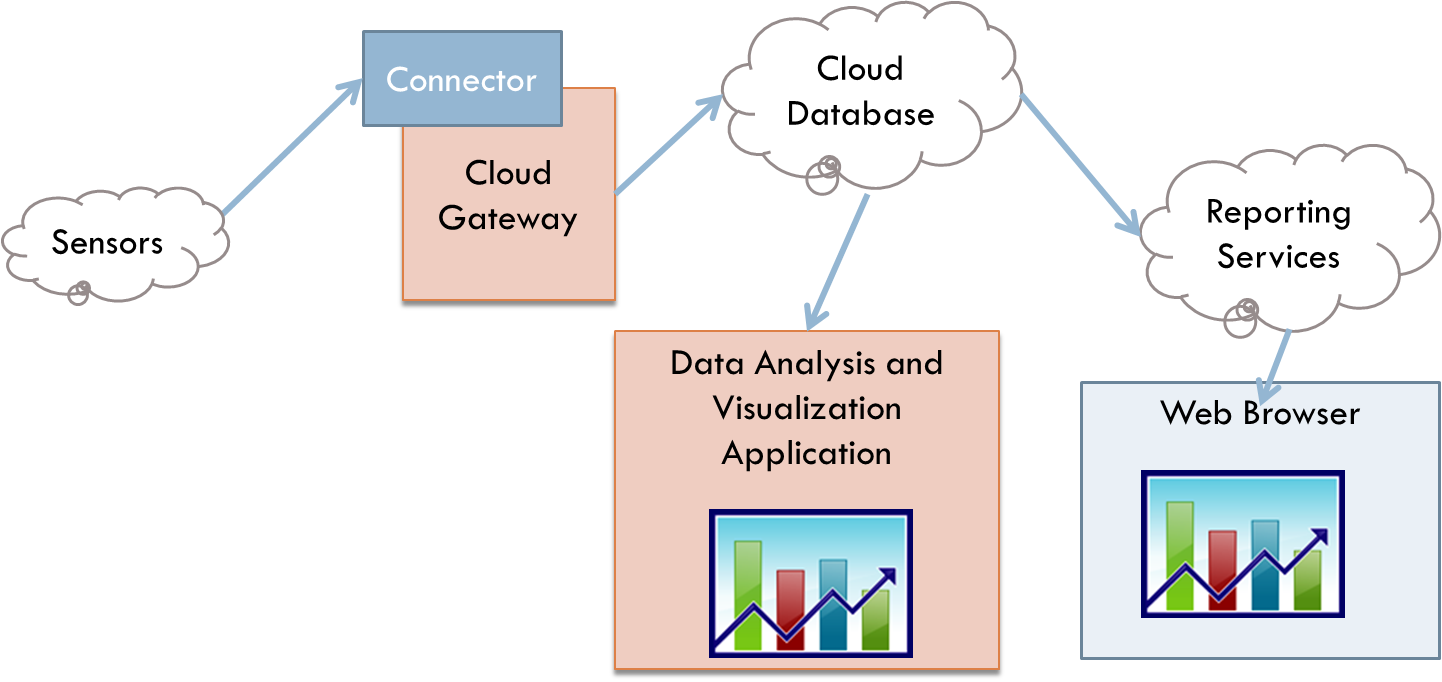


Figure : Cloud Reporting Services for Sensor Data

Using the Windows Azure Management Portal, I added [Reporting Services](http://msdn.microsoft.com/en-us/library/windowsazure/gg430130.aspx) for the Sensor project. To enable the development of reports, I installed Visual Studio 2010 on my development machine. In Visual Studio 2010 there is Reports Application Template to develop projects containing database reports. The projects include a .NET class called [Report Viewer](http://msdn.microsoft.com/en-us/library/ms251671(v=vs.80).aspx). This class can be embedded in applications to visualize database data using tables and graphical charts.

SQL Report Designer in Business Intelligence Development Studio is used to develop to build visualization applications that can run as standalone programs or web pages. SQL Azure provides built-in reporting services that can be set up as web services.

SQL stored procedures are developed to be used by SQL Reporting Services to obtain real time and historical data (Figure 8). In this stored procedure, hourly average temperature values are computed between two dates and return to the caller. Figure 8 shows the stored procedure interface in the Azure portal. This provides a simple and quick way to develop stored procedures directly on a webpage. More complex code can be written in Visual Studio.

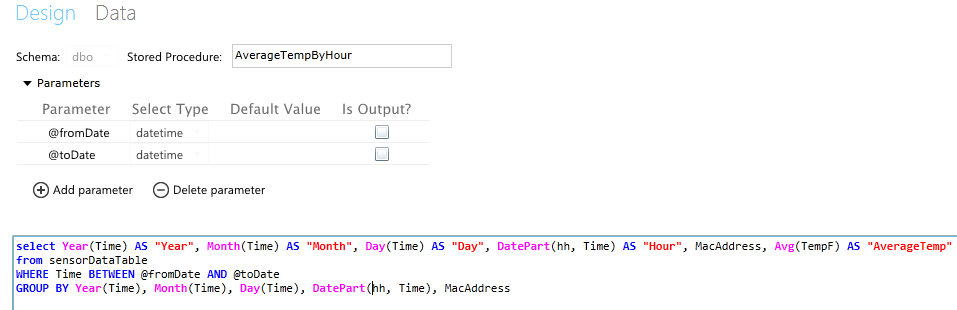


Figure : Stored Procedure for Average Temperature by Hour

# Execution Output

Visual Reports can be developed in the visual studio report designer. These reports can be embedded in applications or uploaded to SQL reporting services. Screen designs given below are examples to show possibilities with data and are not prescriptive.

In Figure 9, in the report designer, data charts are constructed using the data results from the stored procedures as data sources on the left. The design is saved in a .rdl file which can be compiled as an application or uploaded to the web service.

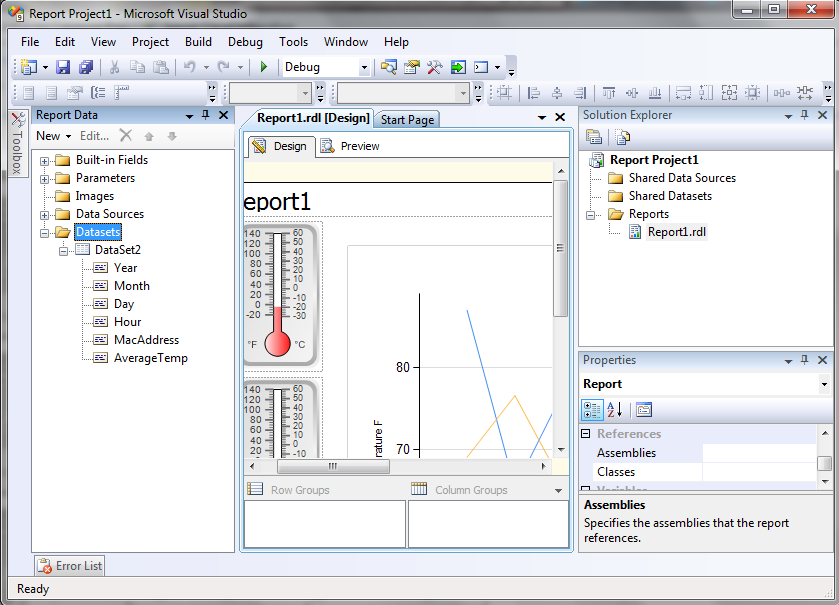


Figure : Visual Studio Report Designer

Figure 10 shows a report example that has data for hourly average temperature between two dates (3/15/2012 to 4/01/2012). The graph on the right shows average temperatures over the time range. The gadgets on the left show the overall minimum for each temperature sensor. Report Designer provides many kinds of these customizable gadgets.

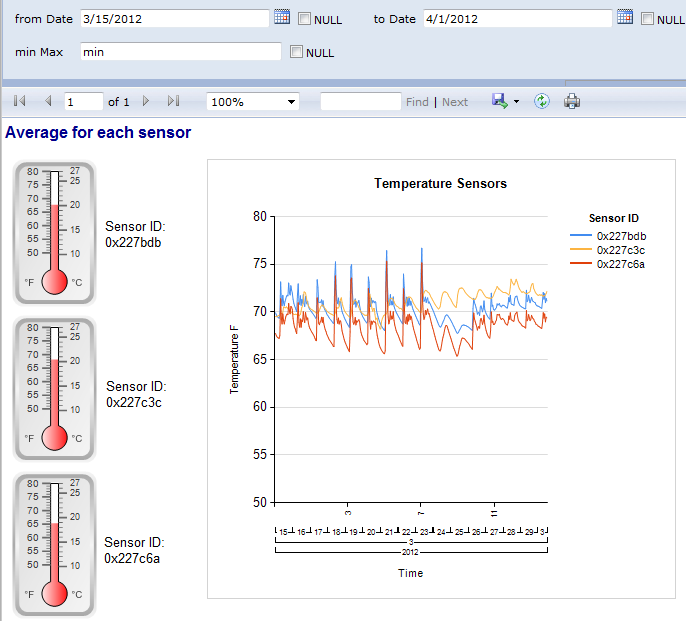


Figure : Sensor Temperature Report

Figure 11 shows average network signal strength between two dates (3/1/2012 - 4/24/2012). On the left, there is one gadget for every sensor with its ID, showing the average network signal strength for the date range. On the right, the chart shows the average network signal for each day and for each sensor.

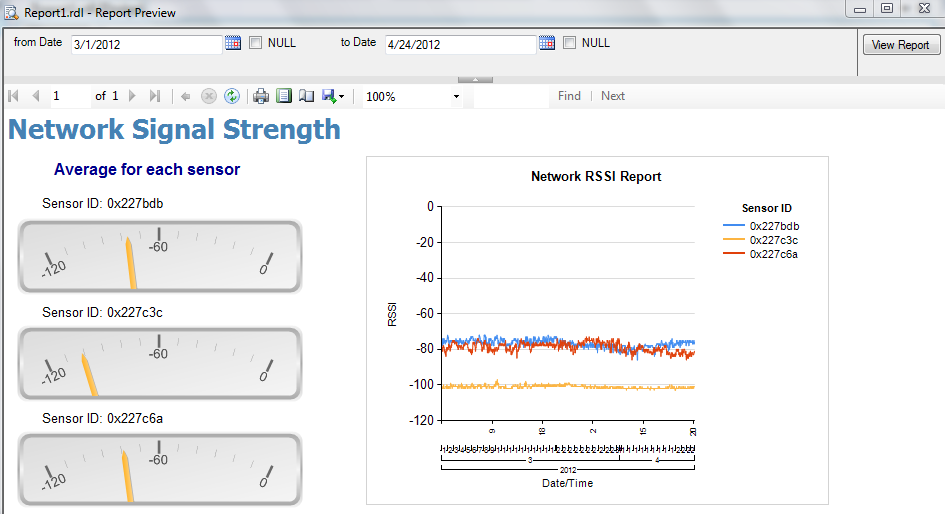


Figure : Network Signal Strength

Figure 12 shows battery voltage between two dates (1/1/2012 - 4/24/2012). On the left, there is one gadget for every sensor with its ID, showing the average battery voltage over date range. The color red represents that the battery is low, yellow represent that battery needs to be replaced, and green represents that battery for that sensor is good. On the right, the chart shows the average battery voltage for each day and for each sensor.

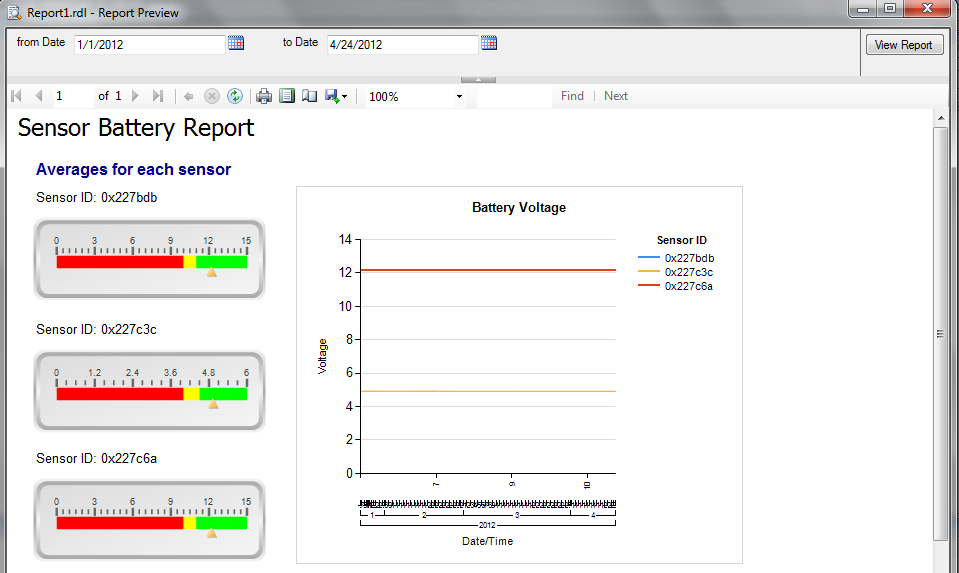


Figure : Sensor Battery Report

# Future Work

# For demonstration, I have developed three data visualization pages. Number and type of these pages can be increased to include new visualization tools and gadgets to show data in different formats and scales.

Accessing cloud databases adds network delays more than any local database. Performance impact of cloud access needs to be analyzed, including database operation and query times.

Sensor data will provide more information when it is combined with data from other data sources. For example, multiple data sources like weather information and mapping data can be processed together to show information on graphs with location data.

# Conclusions

Processing sensor data requires complex queries and relationships, such as correlating data between multiple sensor types and searches over indexed data. In addition, the amount of storage space required for sensor data is usually within the limits of cloud databases. For example, one temperature sensor used only 20MB of space after running six months to store temperature data every minute. Since cloud database limits are much bigger than this size, and there are numerous existing development and visualization tools available for them, cloud databases are a good choice for storing sensor data. If the data size limits are beyond the capabilities of databases, table storage can be used. However, that will require post-processing for data analysis.

Sensor database implemented in SQL Azure was simple to develop and execute. With a cloud database like SQL Azure, the user does not have to install and maintain a database server and related applications. The data is easily accessible through JDBC drivers. This sensor data framework will be the basis of the future sensor data visualization cloud service.

In summary, storing and processing sensor data in the cloud has many advantages:

* Costs efficiency
* Rapid prototyping and development
* Global secure access

Public Clouds are controlled by external vendors. That introduces some risks, such as availability (network) problems, interface changes, cost increase, service outage or closure. Even though these risks also exist on private networks, in the cloud you have to trust another service provider to mitigate the risks. Some customers might think their own processes are better than a separate provider.

# References

Fan, F. & Biagioni, E. S. (2004) “*An Approach to Data Visualization and Interpretation for Sensor Networks*” Proceedings of the 37th Hawaii International Conference on System Sciences.

Hu, Y. & Li, D. & He, X. & Sun, T. & Han, Y. ( 2009). “*The Implementation of Wireless Sensor Network Visualization Platform based on Wetland Monitoring*” Second International Conference on Intelligent Networks and Intelligent Systems

*Kim, E. K. & Ceong, H. & Jung , S.-J. (2007) “Environmental Sensor Data Visualization for Aquafarm Monitoring” Frontiers in the Convergence of Bioscience and Information Technologies*

Lee, K. Y. & Wang, L. & Ryu, K. H. (2010) “*A System Architecture for Monitoring Sensor Data Stream*” Database/Bioinformatics Laboratory, Chungbuk National University, Korea

Miche, S. & Salehi, A. &, et al, (2009). “*Environmental Monitoring 2.0”. 25th International Conference on Data Engineering (ICDE 2009*), Shanghai, China.

Rahman, M.A. El Saddik, A. Gueaieb, W. (2009) “*Data Visualization: From Body Sensor Network to Social Networks*” Robotic and Sensors Environments, 2009. ROSE 2009. IEEE International Workshop

Shiraishi, Y.,(2007). “*A User-centric Approach for Interactive Visualization and Mapping of Geo-sensor Data*”. Networked Sensing Systems, 2007. INSS '07. Fourth International Conference on. 134 – 137. Tokyo Univ., Chiba

Song, X. & Wang, C. & Kagawa, M. & Raghavan, V. (2010) “*Real-time Monitoring Portal for Urban Environment Using Sensor Web Technology*”. 18th International Conference on Geoinformatics.

Talukder, A. & Panangadan, A. (2009) “*Online Visualization of Adaptive Distributed Sensor Webs*” Aerospace conference, IEEE