

Air-dropped Sensor Network for Real-time High-fidelity Volcano Monitoring*

Wen-Zhan Song, Renjie Huang,
Mingsen Xu, Andy Ma, Behrooz Shirazi
Washington State University
Vancouver, WA, USA
{songwz, renjie_huang, mingsen_xu,
hama, shirazi}@wsu.edu

Richard LaHusen
Cascades Volcano Observatory
U.S. Geological Survey
Vancouver, WA, USA
rlahusen@usgs.gov

ABSTRACT

This paper presents the design and deployment experience of an air-dropped wireless sensor network for volcano hazard monitoring. The deployment of five stations into the rugged crater of Mount St. Helens only took one hour with a helicopter. The stations communicate with each other through an amplified 802.15.4 radio and establish a self-forming and self-healing multi-hop wireless network. The distance between stations is up to 2 km. Each sensor station collects and delivers real-time continuous seismic, infrasonic, lightning, GPS raw data to a gateway. The main contribution of this paper is the design and evaluation of a robust sensor network to replace data loggers and provide real-time long-term volcano monitoring. The system supports UTC-time synchronized data acquisition with 1ms accuracy, and is online configurable. It has been tested in the lab environment, the outdoor campus and the volcano crater. Despite the heavy rain, snow, and ice as well as gusts exceeding 120 miles per hour, the sensor network has achieved a remarkable packet delivery ratio above 99% with an overall system uptime of about 93.8% over the 1.5 months evaluation period after deployment. Our initial deployment experiences with the system have alleviated the doubts of domain scientists and prove to them that a low-cost sensor network system can support real-time monitoring in extremely harsh environments.

Categories and Subject Descriptors

J.2 [Computer Applications]: Physical Sciences and Engineering - Earth and atmospheric sciences; C.2.1 [Computer Communication Networks]: Network Architecture and Design - Wireless Communications

General Terms

Design, Experimentation, Performance, Reliability

*This work is supported by NASA ESTO AIST program and USGS Volcano Hazard program under the research grant NNX06AE42G.

Copyright 2008 Association for Computing Machinery. ACM acknowledges that this contribution was authored or co-authored by an employee, contractor or affiliate of the U.S. Government. As such, the Government retains a nonexclusive, royalty-free right to publish or reproduce this article, or to allow others to do so, for Government purposes only.

MobiSys'09, June 22–25, 2009, Kraków, Poland.

Copyright 2009 ACM 978-1-60558-566-6/09/06 ...\$5.00.

Keywords

Sensor network, Design and deployment

1. INTRODUCTION

In the last 15 years, volcanic eruptions have killed more than 29,000 people and caused billions of dollars in damage [5]. In the past, in the case of temporary geophysical monitoring deployments, stand-alone data loggers were often used to record signals to a flash memory card or hard drive and later retrieved manually every several weeks, involving significant human effort. Permanent installations, on the other hand, sent their data to an observatory via analog or digital telemetry. The amount of data transmitted is limited by bandwidth of the hardware, hardships of siting telemetry links, and computational and storage limitations at the observatory. As a result, even many threatening and active volcanoes (e.g., Mount St. Helens) maintain networks of less than ten stations. Scientists frequently lack of sufficient real-time and high-fidelity data for volcano analysis and eruption prediction [23]. Wireless sensor networks have the potential to greatly enhance the understanding of volcano activities by allowing large distributed deployments of sensor nodes in difficult-to-reach or hazardous areas. Wireless communication permits sensor nodes to communicate with each other and to a central base station via a smart self-healing multi-hop network, allowing intelligent real-time data reduction as well as retasking the sensor array after deployment. In the past it took USGS (U.S. Geological Survey) several hours, even days to deploy a single monitoring station as it required scientists to enter the crater, dig a hole, bury geophones in the upright position, and precisely orient the directional antenna to a remote gateway (Figure 1 right). A smart sensor network makes the air-drop deployment possible.

An active volcano provides a challenging environment to examine and advance sensor network technology. The crater is a three dimensional environment with very rugged terrain (Figure 1 left), and generates frequent volcanic activities such as rock avalanches, land slides, earthquakes and gas/steam emissions. Volcanic eruptions may even destroy stations. The occasional eruptions, as well as the heavy rain, snow, ice and wind weather conditions pose significant challenges on the network robustness and self-organizing/self-healing ability [12, 13, 32]. Battery is the only reliable energy source, because solar panels do not work during the long rainy season in the Northwest region. Various geophysical and geochemical sensors generate continuous high-fidelity

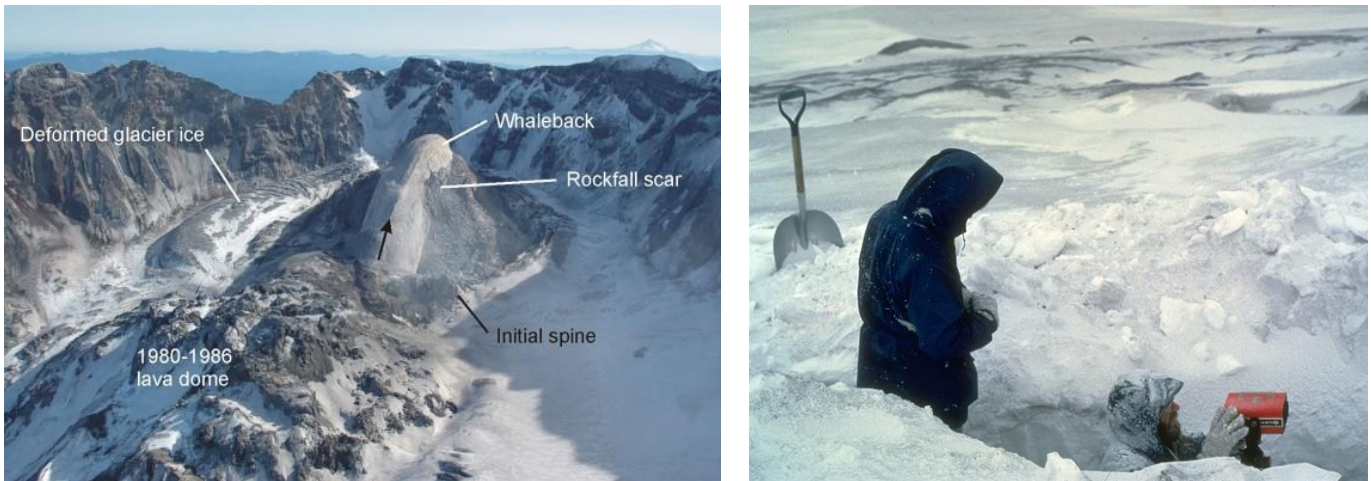


Figure 1: Volcano Crater Environment. (Left) The panorama of the rugged volcano crater with a 1-mile diameter. (Right) USGS scientists worked inside the crater to install an old monitoring station in winter 2005.

data, and there is a compelling need for real-time *raw* data for volcano eruption prediction research. For such a high data rate application, a key challenge is how to collect those high-fidelity data subject to the limited bandwidth available to sensor nodes. In addition to the limited physical bit rate of the radios used on those low power platforms, radio links often experience packet loss due to congestion, interference, and multi-path effects. These problems are exacerbated over multi-hop networking.

In this paper, we present our system design and deployment experiences of a sensor network for real-time high-fidelity volcano monitoring. The design goal is support one-year continuous operation in the volcano environment. To achieve this goal, we designed a comprehensive system with a number of features, such as robust communication stacks, intelligent sensing components, hybrid time synchronization protocols and light-weight network management tools. The system design has been successfully tested in the lab environment, the outdoor campus environment as well as the volcano crater. On October 15, 2008, we air-dropped five monitoring stations into the crater of Mount St. Helens, which took only one hour. These stations formed a multi-hop network and immediately began delivering real-time continuous seismic, infrasonic, lightning, GPS raw data to the control center. Despite the harsh weather conditions (e.g., heavy rain, snow, icing and gusts) in the 1.5 months after deployment, the sensor network achieves a remarkable packet delivery ratio above 99% and the overall system uptime is about 93.8%. Many development and management lessons have been learned and documented. The ground sensor network described here will be integrated with NASA EO-1 sensorweb [4] and USGS science system, to form an Autonomous Space In-situ Sensorweb (OASIS) [1] for earth hazard monitoring. The OASIS system integrates space and in-situ sensors into a semi-closed loop and feeds information into an earth science decision system. More information about the OASIS system can be found at [1].

The main contribution of this paper is the design and deployment of a robust sensor network to replace data loggers and provide real-time continuous volcano monitoring. The

presented evaluation results demonstrated that the network achieves a remarkable reliability despite harsh weather conditions. The data quality meets the demand of USGS scientists. The presented system has broader impact beyond volcano monitoring. Many seismic exploration applications, such as oil field seismic explorations [7], have similar requirements: time-synchronized high-fidelity data acquisition with online configurability.

The rest of this paper is organized as follows. Section 2 presents the system design of hardware and software components. Section 3 describes our campus test experiences and findings. In section 4, we present our field deployment experience on Mount St. Helens and evaluate the performance of the system. Section 5 discusses related work and Section 6 concludes the paper with management lessons and future plans.

2. SYSTEM DESIGN

The USGS scientists specified the following system requirements:

- **Synchronized Sampling:** To utilize the temporal and spatial correlation of volcano signals, the earth scientists require that all stations perform synchronized sampling and timestamp recorded signals with precise UTC time, with error no more than one millisecond.
- **Real-time Continuous Raw Data:** There are no existing algorithms or models are widely accepted to predict volcano eruptions, thus the scientists need real-time continuous raw data to study the behavior of volcanoes.
- **One-year Robust Operation:** The sensor network must be able to collect raw data continuously for one year, despite the extreme weather conditions and routine volcanic activities.
- **Online Configurable:** The OASIS system [1] aims to integrate complementary space and in-situ elements and build an interactive, autonomous sensorweb. The

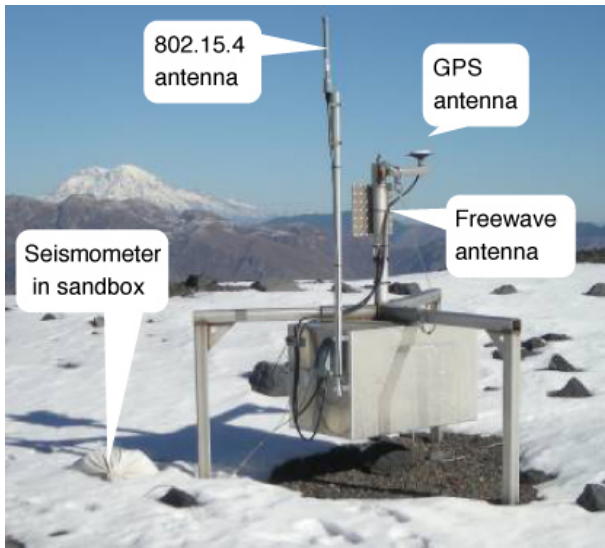


Figure 2: Spider station (sink node) inside volcano crater.

sensor network shall respond to external control from the NASA Earth Observation (EO-1) satellite and USGS science softwares, and adjust its behaviors accordingly. The command and control needs to be delivered reliably in real time.

- **Fast Deployment:** In the past, installing a single monitoring station required significant human intervention, which is not just costly, but also risky. It is strongly preferred that the sensor stations can be air-dropped and self-form a network to reduce the deployment cost and associated safety risks.

To meet these requirements, we comprehensively designed the system architectures, hardware, sensing and networking softwares, such as robust communication stacks, intelligent sensing algorithms, hybrid time synchronization protocols and light-weight network management tools.

2.1 Hardware Design

With the direct involvement of experienced USGS engineers, the hardware design has considered various environment challenges in volcano hazard monitoring. We designed a 3-leg spider station (Figure 2), which is about 4-foot tall including the lifted antenna and weighs about 70 pounds. The spider station is designed to support air-drop deployment and survive in the harsh volcano environment, with the help from our mechanical engineering faculty and students. We mounted a 6 dBi omnidirectional antenna on a PVC steel pipe to get a reasonable line-of-sight. As solar panels do not work well in the Northwest due to the long rainy season, we used heavy AIR-ALKALINE batteries for reliable energy supply. Its weight also helps to stabilize the station in heavy gust. A 900 MHz Freewave radio modem is attached to the sink node to provide long-distance communication to the gateway.

Inside the spider, the core hardware components are encapsulated in a small weather-proof white iron box, with a dimension of $30 \times 20 \text{ cm}^2$ (Figure 3). The white box contains a wireless mote (iMote2), an acquisition board (MDA320CA),

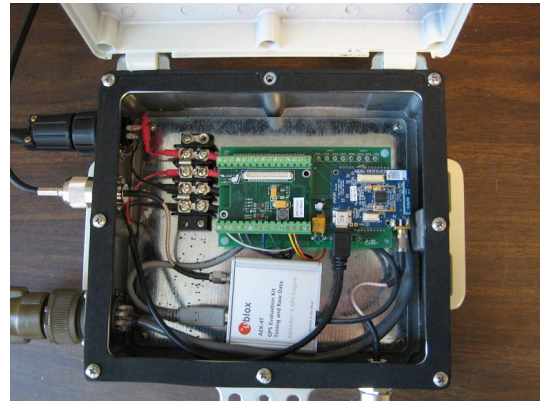


Figure 3: The white box contains iMote2, MDA320CA, U-Blox LEA-4T GPS receiver.

a GPS receiver (LEA-4T) and expansion connectors. We chose iMote2 because it has a good balance between low-power consumption and rich resources: its PXA271 processor can be configured to work from 13 MHz to 416 MHz; it has 256 KB SRAM and 32 MB SDRAM memory space. In addition, it has rich I/O interfaces to support flexible extension, including 2 SPI, 3 UART, and multiple GPIO interfaces. We configured its PXA271 processor to operate in a low voltage (0.85 V) and low frequency (13 MHz) mode in normal operations. A low-power U-Blox LEA-4T L1 GPS receiver is connected to the iMote2 through bluetooth UART interface for raw data capturing, and through GPIO 93 for PPS (pulse-per-second) signal capturing. The GPS is used to measure the ground deformations and timestamp sensor data in UTC time. The accuracy of its time pulse is up to 50 ns, and an accuracy of 15 ns is achievable by using the quantization error information to compensate the granularity of the time pulse. The GPS is also configured to provide UBX-RXM-RAW data for precise ground deformation measurement. The data acquisition board, MDA320CA, is connected to the iMote2 through the SPI interface, acquiring data from seismic, infrasonic and lightning sensors. It provides up to 8 channels of 16-bit analog-to-digital conversion. The device provides a shutdown mode with power dissipation less than $15 \mu\text{W}$. All components are strictly ground power isolated from electromagnetic interference.

The seismic, infrasonic and lightning sensors are connected to the white iron box through coaxial cables. The *seismic sensor*, Silicon Designs Model 1221J-002, is a low-cost MEMS accelerometer for use in zero to medium frequency instrumentation applications that require extremely low noise, and the *2g* version is ideal for seismic applications. The *infrasonic sensor*, Model 1 INCH-D-MV, is a low range differential pressure sensor to record infrasonic, low-frequency ($< 20 \text{ Hz}$) acoustic waves generated during explosive events. This millivolt output pressure sensor is based upon a proprietary technology to reduce output offset or common mode errors. Output offset errors due to temperature changes, warm-up stability, long-term stability, and position sensitivity are all significantly reduced when compared to conventional compensation methods. The *lightning sensor* is a RF pulse detector capable of detecting lightning strikes 10 km away. The entire lightning sensor is built from scratch. It consists of an antenna, MOSFET amplifier and pulse con-

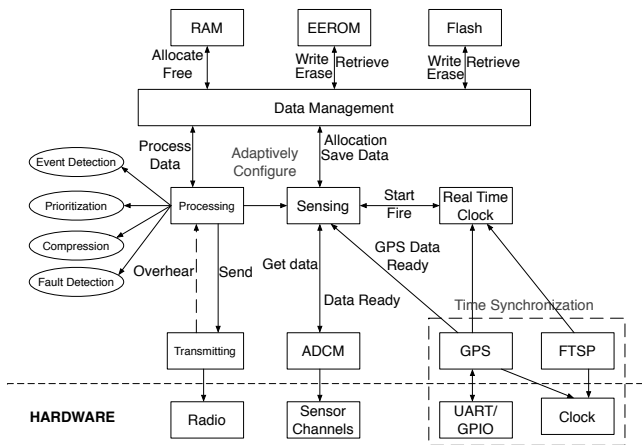


Figure 4: Smart Sensing Component Framework

ditioner. Lightning is detected on seismograms as simultaneous spikes and simultaneous gain-ranging and the typical spike duration is 0.04–0.05 s. Thus lightning strength correlates with both tremor amplitude and magmatic gas content, hence is used for ash detection in this application.

2.2 Smart Sensing

USGS scientists require our network to support high-fidelity synchronized sampling and be online configurable. If the network has to drop packets, they prefer to drop those packets without event information. Naturally, the resource usages shall be driven by the environment and network situations. Figure 4 illustrates the framework of our smart sensing component [21].

2.2.1 UTC Time Stamping and Synchronized Sampling

One requirement from our earth science collaborators is that all stations perform synchronized sampling and timestamp recorded signals with precise UTC time, with error no more than one millisecond. Precise timing is important for utilizing the temporal and spatial correlation of volcano signals. The high-resolution seismic imaging depends on the accuracy of seismic P-wave arrival time differences in the network. Notice that, synchronized sampling is based on time synchronization, but not the same. It means that all sensors in the network sample the environment parameters at the same time point, not just at same interval.

Time synchronization could be realized either through network time synchronization protocols [10, 19], or through GPS signal processing. While GPS provides high timing accuracy, it may be damaged in the harsh volcano environment. FTSP [19] is a multi-hop time synchronization protocol, which achieves high precision performance by utilizing MAC-layer time stamping and comprehensive error compensation including clock skew estimation. The timing errors of less than 67 microseconds were reported for an 11-hop network of Mica2 nodes in [19]. However, previous research [30] has found that FTSP was not stable in their field deployment. Hence, we developed a hybrid time synchronization approach, Z-SYNC, that combines the merits of both GPS and FTSP: each node is synchronized to the GPS receiver by default; if the GPS signal disappears, then the node will switch to FTSP mode. Three situations can drive a node

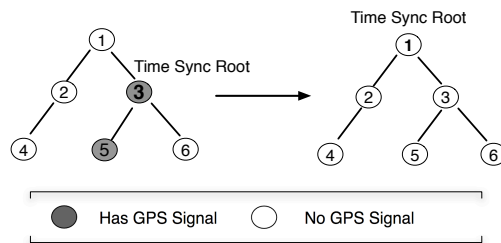


Figure 5: FTSP root election with Z-SYNC: (Left) Node 3 is the TimeSync root; (Right) Node 1 is the TimeSync root.

to switch from GPS mode to FTSP mode: (1) losing PPS signal for more than 10 seconds; (2) losing GPS data for more than 20 seconds; (3) receiving two consecutive invalid GPS data. Later, if valid PPS signal and GPS data is received again, the node switches the synchronization back to the GPS mode immediately. In the original design of FTSP [19], the node with the smallest ID in a multi-hop network is selected as the FTSP root. All other nodes synchronize to the FTSP root. However, the FTSP root might lose GPS signal while some other nodes do not. To solve the problem, we extended the FTSP to support dynamic FTSP root election, as illustrated in Figure 5. If there are GPS nodes in the network, then a non-GPS node synchronizes to the closest one with FTSP protocol. If the whole network loses GPS signals, the node with smallest ID will be elected as the FTSP root.

To support synchronized sampling, we designed a RTC (Real Time Clock) module, which maintains a millisecond resolution timer, with the support of iMote2’s microsecond resolution clock. The timer service provided by RTC is different from the default timer service in TinyOS [6]: it synchronizes its clock value t to the UTC time and fires based on the clock value (instead of interval). In other words, the timer fires when the clock value t satisfies $t \% T = 0$, where T is the sampling interval. For example, if $T = 10$ ms and the timer starts at 20 : 00 : 00 : 422, then the next fire point is 20 : 00 : 00 : 430, not 20 : 00 : 00 : 432. This enables strict synchronized sampling: all sensors in the network may perform periodical sampling at the same time point.

2.2.2 Configurable Sensing

Considering the longevity and remoteness of environment monitoring, online reconfigurability is highly desired in a system deployment. With this feature, users can download the same program to different nodes, then adjust sampling rate, add or delete sensor channels, and configure data processing tasks on any subset of nodes. It naturally supports both self-adaptive configurations and external configurations.

In a smart sensor network, the sampling rate and other sensing parameters shall be adjustable based on environmental conditions and mission needs. For example, the system may use, as a default, a 50 Hz sampling rate for normal data collection. If there is an event detected, it may then increase the sampling rate to get higher fidelity data in a short period. The sensing module in Figure 4 performs synchronized sampling operation and maintains sensing parameters, such as sampling rate, channel (e.g., ADC port number), data

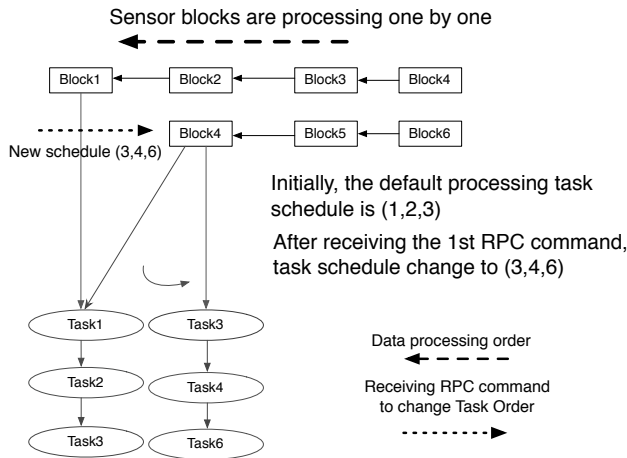


Figure 6: Configurable Processing Tasks

resolution (e.g., 16-bit or 24-bit), sensor status, and reference voltage gain. Currently, our node has 4 sensor channels: the seismic and infrasonic sensors sample at 100 Hz with 16-bit resolution; the lightning sensor samples at 10 Hz rate with 16-bit resolution; the GPS receiver produces about 200 bytes data every 10 seconds. All of these parameters could be tuned according to environmental and resource situations to conserve energy or increase fidelity. This configuration meets the current requirement. In the future, if we want to introduce a new sensor into the system, we only need to attach the sensor to the station and configure a sensing channel for it without the need of reprogramming. If a sensor is broken, its channel can be closed to save energy and bandwidth.

Besides the configurable parameters, it is also useful to configure the data processing tasks for different data types. To support this, each processing algorithm is indexed in a task queue through function pointers. For instance, as illustrated in Figure 6, assuming 6 different tasks have been programmed into the node and the default data processing task list is (1, 2, 3). The user may configure the task list (3, 4, 6) to process some type data. With the future support of the incremental reprogramming over the air, a new task function's binary code could be sent to the node, indexed in the task queue for configurable sensing. With this mechanism we can choose to run different event detection and compression algorithms on different types of data on the fly, without reprogramming the nodes.

As different sensors generate different amounts of data at different time, and different hardware platforms have different amounts of storage space, we designed a sensing data management module to manage sensed data into different storage media based on platform limitations and network situations. If there is enough free RAM space, sensing data will be saved into RAM space instead of flash. Otherwise, the unsent data left in RAM will be moved to flash for later retrieval. In other words, the main function of our data management module is to manage sensed data into an appropriate place for processing and delivery, according to the data priorities and storage availabilities. The data management module hides the implementation details of how a free buffer is allocated and where the sensing data is saved. In our implementation, a list of sensor blocks are allocated ini-

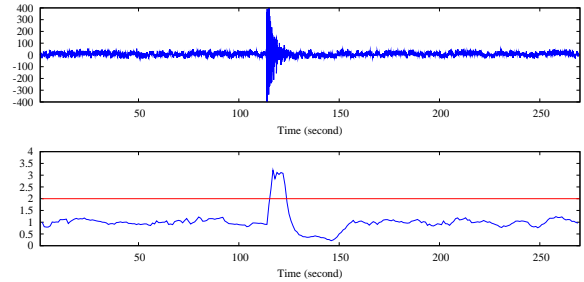


Figure 7: Example of STA/LTA event detection algorithm: (Top) The seismic waveform; (Bottom) The STA/LTA ratio curve. The red line denotes the event triggering threshold 2.

tially. Each of the sensor block has the information of sampling start time, sampling rate, task execution code, status and a raw buffer to save sensing data.

To support reliable online configurations, we have designed a reliable data dissemination protocol and a remote procedure call mechanism, which will be described in section 2.4 and 2.5 respectively.

2.2.3 Situation Awareness

Real-time high-fidelity data collection is constrained by limited network bandwidth. If the network has to drop packets, then it should first drop the packets without critical events information. Hence we designed a situation awareness middleware to detect the environmental conditions. The data during the event period will be assigned higher priority and have more chance to reach the sink through the QoS control in the underlying communication stack.

With the recommendation from earth scientists, we used the STA/LTA (short-term average over long-term average) algorithm [20] on seismic data to locate the earthquake events. To understand STA/LTA algorithm, we need to first introduce the concept of RSAM (Realtime Seismic Amplitude Measurement), which is widely used by seismologists. It is calculated on raw seismic data samples every second. Let m be the number of samples per second, $\{s_t, \dots, s_{t+m-1}\}$ and $\{s_{t-m}, \dots, s_{t-1}\}$ be the raw seismic sample values in the i th and $(i-1)$ th second respectively, then $e_{i-1} = \frac{\sum_{l=t-m}^{t-1} s_l}{m}$ is the average seismic sample value in $(i-1)$ th second. The i th-second RSAM x_i is calculated with the equation:

$$x_i = \frac{\sum_{k=t}^{t+m-1} (s_k - e_{i-1})}{m}$$

In our system, the nodes deliver the RSAM value of seismic and infrasonic data to the sink as required by USGS scientists. The STA or LTA is continuously updated based on the equation:

$$X_i = \frac{\sum_{j=0}^{n-1} x_{i-j}}{n}$$

where x_i is i th-second RSAM; n is the STA or LTA time window size. LTA gives the long term background signal level while the STA responds to short term signal variation. In our implementation, the STA window is 8 seconds; the LTA window is 30 seconds.

The ratio of STA over LTA is constantly monitored. Once the ratio exceeds the trigger threshold, the start of an event

is declared, and the LTA value is frozen so that the reference level is not affected by the event signal. The end of the event is declared when the STA/LTA ratio is below the de-trigger threshold. Figure 7 illustrates the relationship between the seismic waveform and the STA/LTA ratio. It shows that the STA/LTA algorithm with threshold 2 can perform seismic event detection very well.

For lightning data, which has a “strike value” when lightning takes place, a threshold-based trigger approach is applied to detect the event.

2.3 Agile Data Collection Routing

Our data collection routing protocol, MultihopOasis, is developed based on MultihopLQI in TinyOS-1.x [6]. We improved MultihopLQI by considering various practical system issues. Table 1 shows some important parameters we used in routing management. The improvements include:

Table 1: Routing parameters

Parameter	Value
Initial TTL (Time To Live)	31
t_{WDT} (WatchDog check period)	10 min
t_B (Beacon Interval)	10 sec
t_E (The time for a valid Path expire)	$6 t_B$
t_F (The time window to monitor broken link)	15 sec
α (EWMA factor for average LQI)	0.25
Queue space reserved for forwarding packets	66.7%

Link Quality Estimation.

MultihopLQI is a distance vector routing protocol, using LQI (link quality indication) as the routing metric. In CC2420 radio, the LQI measurement is a characterization of the strength and/or quality of a received packet. However, some previous works have reported that LQI fluctuates over the time. Our experiments showed that, instead of using each beacon’s LQI, the average LQI is better to reflect packet delivery ratio. We apply the Exponentially Weighted Moving-Average (EWMA) on LQI values in each time frame (window) and calculate, $lqi(t) = (1-\alpha) \times lqi(t) + \alpha \times lqi(t-1)$, where α is 0.25.

Count-to-Infinity Problem.

MultihopLQI, as a distance vector routing protocol, intrinsically has the count-to-infinity problem. In our design, each packet has a TTL field (5 bits) which is initialized to 31 at the source node. TTL decreases 1 each time the packet traverses an intermediate node. When TTL reaches 0, the packet is discarded. Moreover, each sensor node maintains a history message queue. Each entry records a triple (sourceID, seqno, TTL) of a forwarded packet in history. When forwarding a packet, a node looks up its history queue. If there exists an entry that has the same source address and sequence number, but a different TTL, then a loop is detected, otherwise a duplicated packet is received. TTL is necessary to differentiate route loops from packet duplications. If a loop is detected, the routing protocol will invalidate the path and re-discover a new path immediately.

Agile Network Self-forming and Self-healing.

When a node detects a loop, or it does not receive route beacon from its parent for more than 6 beacon periods, or all

packet transmissions in last 15 seconds fail, it will invalidate its link to its current parent. In standard distance vector routing protocol, each node periodically broadcasts beacon packets to update route information. The node with invalid parent must wait new beacons to re-discover a new path to the sink. During this waiting period, the node has to wait and may discard some packets due to buffer overflow. The time delay is proportional to the length of the beacon period. A short beacon period can mitigate the problem, but it has the disadvantage of introducing more beaconing overhead, which is not desirable. To makes the network agile to link status changes, we make a few changes to enable fast self-healing: (1) When a node’s parent becomes invalid, it invalidates the parent immediately and issues a beacon packet indicating infinite path cost. Additionally, if a node receives a beacon from its current parent containing bad link news, it also invalidates its current parent and sends out a beacon message to notify other nodes immediately. (2) If a sensor node with an invalid parent receives a beacon and discovers a new parent, it will broadcast a beacon of the new route information immediately. In that way, the network will take a short time to self-form or self-heal. This is especially effective during the network start-up.

As mentioned above, when a node detects a loop, beacon timeout, or maximum successive transmission failures, it will invalidate its current parent and wait until a new path is discovered. While a reactive approach for fast network healing is proposed above, we also utilize a proactive approach, multi-path backup, further reduce the recovery delay. If a node’s parent becomes invalid, it searches for an alternative in its route table first. In our design, a neighbor management module caches all valid paths toward the sink proactively. Once a received beacon indicates a valid path toward the sink, neighbor management module deposits the route information to the route table. In a dense network, a node may have more neighbor nodes than the route table size. Our replacement strategy not only considers path cost, but also consider the liveness. Each routing table entry has a *liveness* field. If an entry is updated by a beacon, its *liveness* is reset to a maximum value; if a update timer fires, the *liveness* value will decrease by 1; if its *liveness* is decreased to 0, then that entry becomes invalid. When the routing component looks up an alternative path to the sink, the *live* path with lowest cost will be chosen, without waiting beacons to rediscover a path.

Asymmetric Link Problems.

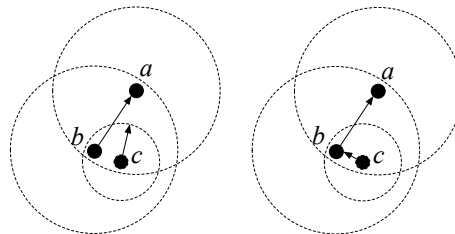


Figure 8: The difference between MultihopLQI (Left) and MultihopOasis (Right) with the existence of asymmetric links. The dash line denotes transmission range.

Asymmetric links commonly exist in wireless network, due to many factors such as the irregularity of radio transmission range and antenna orientations. As illustrated in Figure 8, node c can hear from node a , but node a cannot hear from node c . Asymmetric link can cause complete node failure: although a child node can receive beacon packets from its parent, its own packets may never reach its parent. With ARQ (Automatic Retransmission Request) mechanism in the link layer, the child node will retransmit the unacknowledged packet and degrade the network throughput. MultihopOasis addresses the asymmetric link problem by monitoring link status. If its link loss ratio during a time window exceeds the threshold, the child node switches to an alternative parent as described in previous paragraph, and puts the old parent into blacklist to avoid switch oscillation. As illustrated in Figure 8, with MultihopLQI, node c chooses node a as parent, but its packets never get an acknowledgement; with MultihopOasis, node c will switch parent to node b after maximum successive transmission failures.

2.4 Reliable Data Dissemination

As introduced earlier, the presented sensor network is part of an autonomous space in-situ sensor web, hence interaction with external components is an important requirement. For example, a science software may generate control commands to adjust sensing parameters in real-time when some monitored area becomes more active. Those feedback commands require 100% reliable delivery in the correct order. Moreover, the dissemination process should terminate in acceptable time window. To our best knowledge, the existing dissemination protocols either do not guarantee 100% reliability or do not terminate [14, 18] in short time (e.g., based on gossip). So we designed the *Cascades* protocol for reliable and fast data dissemination. The *Cascades* protocol ensures 100% reliability by utilizing the parent-children relationship in a data collection tree: each node monitors whether its children have received the broadcast messages through snooping. Each node rebroadcasts periodically until successful reception is confirmed by all children. The broadcast flow does not depend on the data collection tree structure. A snooped new message from neighbor nodes will be accepted and rebroadcasted. Therefore, it is possible that a node first hears new data from its children, before it hears it from its parent. In other words, it is a fast opportunistic data dissemination protocol. In addition, *Cascades* performs a reactive fetch mechanism [29] if there is gap in packet sequence, which denotes missing packets.

There might be multiple control clients on the Internet, hence one key challenge of reliable broadcast is the consistency of sequence number. In addition, there might be multiple sink nodes in the sensor network, and a sensor node could be rebooted. All those facts could cause the sequence number in a network to be inconsistent, then the nodes cannot differentiate between old and new commands. We solved this problem by maintaining the sequence number in SerialForwarder, a TinyOS tool modified to support packets forwarding between Internet and multiple sink nodes. In addition, *Cascades* resets its sequence number if it is idle for 10 minutes. With this mechanism in place, even if a node's sequence lost synchronization, it can re-synchronize with the SerialForwarder after 10 minutes. More details can be found in [22].

2.5 Network Management and Robustness

Network management is more challenging in sensor networks than in traditional networks, due to the limited computation, storage and communication resources, and diverse application requirements. The system robustness is also particularly important, because those sensors typically work in hostile environments, it has to rely on itself for recovery. In this section, we present a light-weighted sensor network management mechanism [33], and a comprehensive software watchdog to increase network robustness.

2.5.1 Network Control and Status Report

Visibility into network failures is an important network management requirement, as knowing the cause of network failures can help to correct bugs and improve the system. We developed a transparent and light-weighted RPC (Remote Procedure Call) mechanism based on [31], which has little overhead on sensor nodes. It allows a PC to access the exported functions and any global variables of a statically-compiled program on sensor nodes at run time. At pre-compilation time, a RPC server stub is automatically generated and added to a nesC application. All information necessary for a RPC client to use RPC is parsed from the application source code and exported to an XML file by a Perl program. At run-time, the RPC client imports all information from the XML file. Figure 9 illustrates the process.

The RPC mechanism gives users great flexibility to read/write system variables and run any exported functions. Operations such as set/get sampling rate, beacon interval, power level, radio channel, event report threshold are provided to remote clients. In addition, RPC also helped us to solve some rare, but challenging, system bugs. For example, we found that occasionally some nodes forward packets for others, but they stopped sending their own packets. It could be because the sampling timer stops triggering, or the data management in sensing component corrupts. The problem only happens occasionally and is hard to repeat. So traditional debugging approaches fail to find out the cause. However, with the visibility and flexibility of RPC, we were able to narrow down the scope of suspicious code and identify the problem.

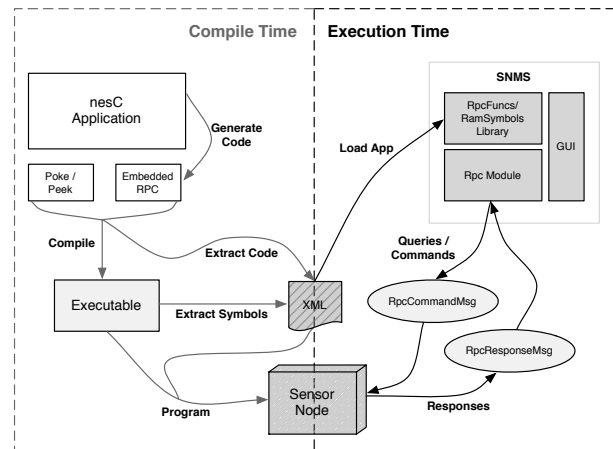


Figure 9: The RPC solution to reduce the overhead of sensor node management.

We also enable the sensor network to report important events or node status periodically for network health diagnosis, such as battery voltage, buffer status, and neighbor RSSI and LQI. The control client is then able to monitor network connectivity and health. This information is particularly useful during field deployment and evaluations. We realize this through an EventReport component, which provides users the flexibility to monitor different levels of system status. It uses a mechanism similar to *dbg()* in software debugging. Users can customize the emergency level of events. Events with emergency level below the report threshold will be filtered out to reduce the communication cost. The report threshold is configurable during run-time using the RPC command. It allows users to control the amount of status report information based on network traffic situations. By default, no filters are set for reported events so that all of the events are sent to the client.

2.5.2 System Design for Robustness

In the reality of software engineering, it is almost impossible to find and fix all bugs. A distributed wireless sensor network is especially hard to debug, although the community has developed various debugging tools. But volcanologists expect our system is capable of running continuously on St. Helens Volcano for at least one year. It is, therefore, crucial to have an exception handling mechanism that allows it to recover automatically from software and hardware failures. We exploit the benefits of watchdog. The iMote2's hardware watchdog can restart the node if there is any dead loop, illegal memory access, stack overflow, or division by zero. We further developed a software watchdog to enable self-recovery from unexpected logic errors. Those errors do not necessarily cause sensor nodes to die, but make them unstable. For example, if the communication protocol or the message queue corrupts due to whatever reasons, a node may no longer accept new messages. In our system, if no messages are sent or received for 10 minutes, then the node will be rebooted. The control center can also send a RPC command to reboot a node if necessary.

In addition, we also made some improvements in TinyOS-1.x. TinyOS is an event-driven system and does not support multi-threading. In recommended practice of TinyOS-1.x, a up-layer component cannot send another message to a low-layer component until the *sendDone* event of the previous message is received. This could cause a long delay between two consecutive sends. Similar situation happens during receiving. We enabled pipelined sending/receive approach through buffering messages in sending/receiving queues at each layer. Also, we found and solved some problems of the CC2420 radio driver in TinyOS-1.x: (1) Under intense network contention, the radio stack may receive a packet with valid CRC, but with the length that is smaller or bigger than the real packet length, due to some problems in radio software or hardware. Those corrupted packets not only waste network bandwidth, but also cause the control client to die. We suppressed those corrupted packets by performing a sanity check on the packet length field. (2) Sometime, when *sendDone* fails, the packet header (e.g. AM type) may also be corrupted. This causes the message pointer to return to a wrong up-layer component in TinyOS. We avoided this through backing up the packet header before sending. When a *sendDone* event is received, the packet header is restored. (3) Sometimes the *sendDone* event for the same packet may

be signaled twice, perhaps due to some race conditions. Our queue management module is resilient to such unexpected situations and ignores the duplicated events.

3. CAMPUS OUTDOOR TEST

Before deploying the sensor network into the crater of Mount St. Helens, we conducted a 3-month outdoor test on the university campus. The system setup of the campus test was almost the same as that of the field deployment. The campus test exposed many software and hardware design problems, which have been posted to the TestNote page of our lab's wiki [1].

Originally, we planned to use a 4-foot tall 12 dBi omnidirectional antenna to extend the transmission range of CC2420 radio. However, the transmission range was only about 25 meters, which is even less than without it. The reason will be explained later. Then we began to seek an amplifier to increase the radio transmission power. Unfortunately, there are no commercial off-the-shelf 802.15.4 radio amplifier to amplify input signals under 0 dBm (the maximum transmission power of CC2420 radio). Eventually, we found a company who could customize its radio amplifier SmartAmp to support -3 dBm input amplification. It did improve the signal strength significantly. The commercial RF measurement device, Anritsu S332D Site Master, showed that the output power was close to -10 dBm in one meter distance. However, the network packet loss ratio was very high if two nodes were separated more than 50 meters. We made various attempts, such as replacing a new iMote2 or replacing a cable. Finally, we realized that it might be because the signal strength was high, but signal quality (e.g., signal noise ratio) was low, hence we developed a packet-level LQI measurement program called TOSBaseLQI (based on TOSBase in TinyOS). Indeed, we found that the packet LQI was below 70 (typically, above 90 is good, 110 is the maximum), though RSSI was about -30 dBm (which is far above the CC2420 receiver sensitivity threshold). We learned a lesson from this: signal strength does not necessarily reflect signal quality. It is actually not surprising because link quality is correlated with signal noise ratio, not the signal strength. After we knew the true reason, we began to reduce the potential noise sources between iMote2 and the amplifier. Eventually, we found that a L-connector between the iMote2 and the amplifier was the problem. The L-connector did not attenuate the signal much, but it added noise which was also amplified. Small changes made big differences. After the L-connector was removed, the LQI of received packets increased to $90 \sim 106$, even at a distance of more than 400 meters. Without the quantitative measurements of link quality using TOSBaseLQI, we would think it was normal, since low link reliability is the common assumption in the wireless networking community. In fact, it may also be caused by a bad RF design, like the example given above. This experience also taught us that LQI was a more reliable and accurate link metric, although some research [26] appreciated RSSI. After we learned this lesson, we began to record the LQI of each link to a database.

Then we found another problem: the sink node's LQI dropped from 105 to about 70 during 1-7 PM everyday, while other nodes did not have this problem, although they shared exactly same hardware configuration. We wondered whether it was a defect of certain hardware components that made the difference and tried to replace various hardware

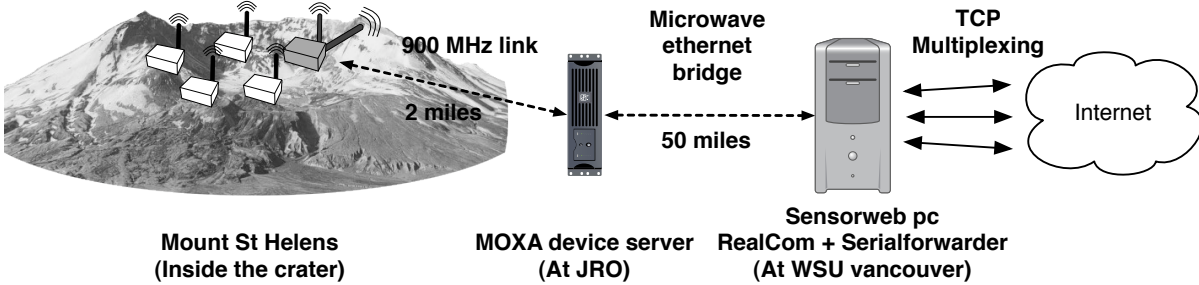


Figure 10: The system configuration of the air-dropped volcano monitoring sensor network

components in the sink node. By chance we found that, when we replaced the cable that connected the amplifier and the antenna with a short thin one, with mark BELDEN 8262 M17/155-00001 MIL-C-17 16428 2137 19:22 ROHS, the LQI did not drop during 1-7PM. The thick cable, with mark LMR@-400-ULTRAFLEX COAXIAL CABLE TIMES MICROWAVE SYSTEMS, typically has higher quality. Then we wondered whether it was because a long cable introduced more noises than a short one. However, a short thick cable did not change the LQI fluctuation situation. Finally, we replaced the long thick cable with a long thin cable to solve the problem. Although we found this solution to make the link quality stable, this problem is still a mystery to us: why does this phenomenon only appear at one node and only between certain hours of the day?

In the lab test, we set up a GPS antenna outside of the building and linked its signals into the room with a cable, and everything worked well. But sensor nodes became unstable on time synchronization after we moved to the outdoor campus. It took us several days to find the reason. In the lab test, the nodes were close to the building wall, hence they received at most 8 satellite signals. When the nodes were placed in the open field, they could receive more than 13 satellite signals at the same time, which caused GPS data buffer overflow and triggered the watchdog to restart nodes frequently, thus unable to synchronize to GPS. We fix it by increasing the data buffer size.

During the campus outdoor test, we also learned that we should have paid more attention to the connection between components, including software interface and hardware interface. Besides the example of the L-connector, there are several problems caused by interconnection in the system. For instance, the sensor board is connected to the iMote2 through a interconnection board. But during a movement, it will occasionally become loose. When the sensor board is loose, the data acquired is a straight line of `0xFFFF` value.

4. FIELD DEPLOYMENT

4.1 Deployment Experience and System Configuration

On October 15th 2008, we successfully air-dropped 5 stations into the crater of Mount St. Helens and streamed data to the Internet. The deployment map is illustrated in Figure 11. First, the sink node with ID 10 was lifted up and dropped into the crater and it immediately showed up on our

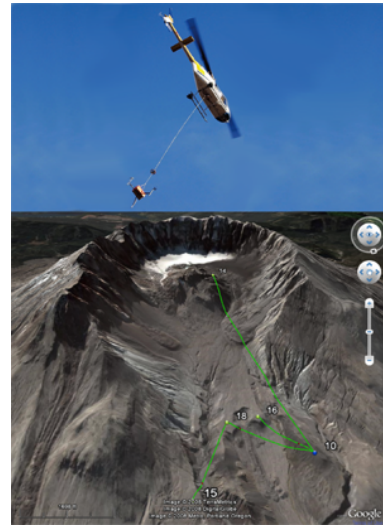


Figure 11: Air-drop deployment and the panorama of the network

monitor. Then, when the second node with ID 14 was lifted up and toward the crater, we were excited to see that node 14 had linked the remaining ground stations at the preparation site to node 10 inside crater. The distance between the preparation site and the crater is about 4 miles. The radio transmission range was remarkably long and beyond our expectations. The same situation repeated during the rest of the deployment. Our optimized data collection routing protocol worked remarkably well and formed the network immediately. In our routing protocol, once a node discovers its parent, it broadcasts a route beacon immediately. This helps to build the network immediately in that short transition period. If the distance vector is broadcasted periodically, this might not be observed because the helicopter took less than 1 minute to fly from the preparation site to the crater. We have made several other improvements on the routing protocols as described in previous sections. After all nodes are deployed, we found that node 16's seismic data plotted a straight line. From our experiences in campus test, we recognized that this meant that the seismic sensor was broken. We believe that the sandbox containing the seismic sensor was spinning too hard during the helicopter lift, causing the connection to break. We returned the crater

and fixed the problem. This was the only problem we encountered during the air-drop deployment.

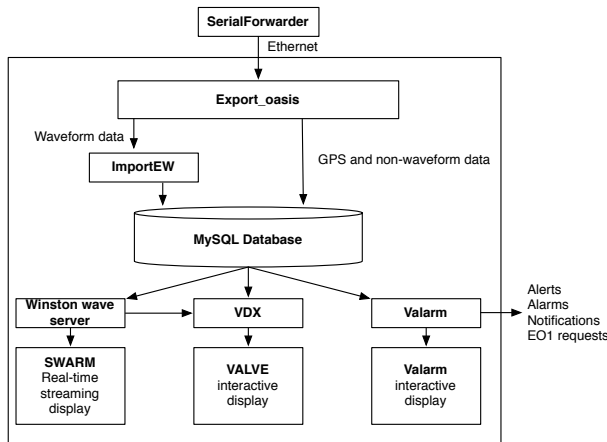


Figure 12: The network management and data manipulation tools

Figure 10 illustrates the system configurations. The rugged crater has a diameter of around 1 mile. During the deployment, the air-dropped nodes immediately formed a data collection tree, and delivered real-time sensing data to the sink node through the multi-hop network. The sink node then relayed the data to gateway (e.g., MOXA device server) at JRO through a Freewave radio modem. Then the gateway relayed the data stream to our sensorweb research lab through a microwave link of 50 miles. In the lab, a SerialForwarder java tool forwards data between the sensor network and the Internet. Multiple control clients may connect to it, access the sensor data stream and control the network in real-time. Figure 12 illustrates the software architecture of network management and data manipulation tools. ExportOasis imports the data stream from seismic, infrasonic, lightning sensors, and GPS, as well as RSAM, battery voltage and LQI data, into a MySQL database. Wave data (seismic and infrasonic) is reformatted into standard form for storage by ImportEW before being logged into the MySQL database. VALVE (Volcano Analysis and Visualization Environment) is a web application that allows on-demand visualization of history sensing data from any location on the Internet. VDX (Valve Data Exchange) handles the interaction (e.g., requests and responses) between the VALVE web application and the Winston Wave Server. SWARM (Seismic Wave Analysis and Realtime Monitor) is similar to VALVE, but has the ability to show the real-time waveform data stream. SWARM and VALVE are existing USGS data manipulation tools, which supports zoom in/out and can draw spectrum graph of raw seismic/infrasonic data. Valarm is a volcano alarm system, which can automatically identify earthquake events from the raw data stream. Once an event is triggered, Valarm can send the event report via email or short messages to corresponding scientists in charge. Valarm is developed in our OASIS [1] project.

Besides that, we developed a realtime network monitor tool for monitoring and debugging. It can display the network topology and the status of each node, and visualize the real-time sensing data in an oscilloscope view. Moreover, it can record events from sensor nodes and send email alerts

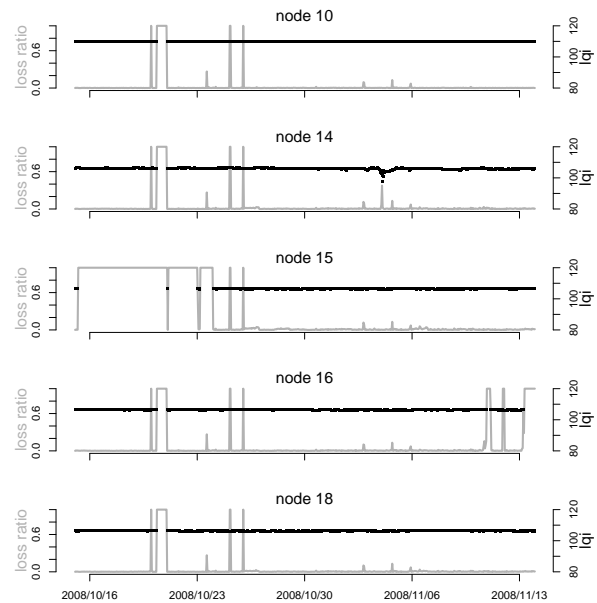


Figure 13: Hourly loss ratio (gray) and parent-lqi (black) for all the nodes during the first month of deployment.

to users if a node stops sending packet for a certain period. It also provides users the interface to issue user commands to adjust network behavior.

4.2 System Evaluations and Findings

In this section, we analyze the deployed system performance and compare it with existing USGS volcano monitoring stations in Mount St. Helens.

4.2.1 Network Evaluations and Findings

We computed the loss ratio from the data logged in the database, and found the end-to-end overall data delivery ratio as high as 91.7%. Notice that this loss ratio is not the sensor network packet delivery ratio, but the end-to-end overall system delivery ratio. If we do not count the failures of data servers, java tools, Internet and hardware, the sensor network itself achieves a remarkable 99% packet delivery ratio over the 1.5-month evaluation period. Figure 13 illustrates the hourly loss ratio for all the nodes during the first month of deployment.

In Figure 13, we can see that node 15 was down for one week. About 6 hours after the deployment, node 15 disappeared. A USGS engineer went to the Mount St Helens for another mission five days later, and found the station was tipped over (Figure 14). He landed, placed the station in an upright position, and then weighted it with big rocks in the hope that it wouldn't be tipped over again. However, only two hours after he left, node 15 disappeared again. We were wondering if the node was tipped over again, but no more helicopters were traveling to Mount St Helens. We had to find another solution to investigate what happened there. After having explored many possibilities we decided that 3 persons would do a 6-hour hike to the crater to see what was going on. They discovered that node 15 was still on its legs but the voltage regulator had been damaged. After the problem was fixed, node 15 has stayed alive. The problems



Figure 14: Node 15 was tipped over after the first week of deployment.

with node 15 made us realize the weather conditions inside the crater of Mount St Helens are very tough. Nodes experienced high temperature variations, heavy gusts, rains and even snows during the second half month of the deployment. Very strong gusts were recorded by our infrasonic sensors. The infrasonic sensors are essentially pressure sensors, and hence are able to capture those heavy gusts (Figure 15). We checked the weather conditions during that time and found that wind speed reached 120 miles per hour. We can also see that data from all the nodes was lost around 2008/10/20 for about 20 hours. The reason is that ExportOasis, the data importer of the VALVE database, cannot handle daylight saving time changes, because that server was running an old WinXP version without the new daylight saving patch. However, the network still worked normally during that period. We know this because we also logged data timestamp to another database.

In Figure 13, when the loss ratio of all the nodes are 100% at the same time, it means there was an error on the database server or Internet; when only one node has a high loss ratio, it means that this node had some troubles. Node 16 around 2008/11/11 had a relatively low packet delivery ratio. From our data log, it had been restarted frequently. We found the battery voltage level is close to the lowest permissible limit, which perhaps explains why. Before deployment, USGS engineers intended to put the used batteries to all nodes to test our network self-healing ability and stability. Node 16 is one of them and probably run close to the energy depletion status.

Figure 16 shows the uptime of each sensor node and the database server. The uptime was calculated hour by hour using the assumption that if a node is sending data during that hour (even with some loss) then it is up for that hour. The overall network uptime is about 93.8%. The gaps in a solid line denotes either the database server was down or there was no data from the station. From Figure 16, we can see that all stations achieved a high percentage uptime except node 15 due to the troubles described earlier. The other four nodes' uptime is about 100%.

Overall, we were excited to see that, during the whole evaluation period, no nodes died and the system successfully recovered from numerous challenges.

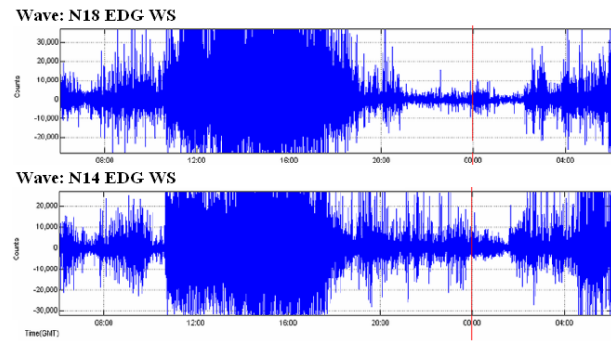


Figure 15: Heavy gusts observed by the infrasonic sensor of node 18 and 14.

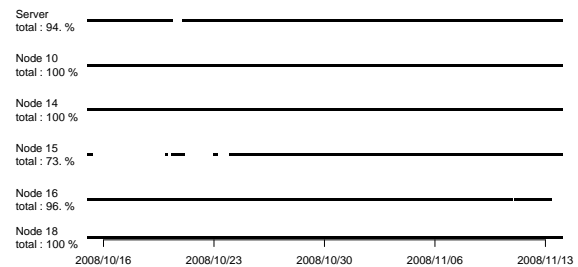


Figure 16: Individual uptime of sensor nodes and data logger server.

4.2.2 Data Evaluations and Findings

One important aspect to evaluate a data collection system is the data quality. To assess the quality of our data we compare it with other data sources from the volcano. USGS owns 4 types of measurement stations on Mount St Helens: (1) Dual frequency GPS with store-and-forward telemetry when polled. (2) Short period seismic stations with geophones and analogue telemetry. (3) Broadband seismic stations with digital telemetry (24-bit ADC, the gold standard in seismic monitoring). (4) Microphones for explosion detection added to the short period seismic stations.

A quick comparison between the capabilities of existing USGS stations and our Oasis station can be found in Table 2. To differentiate our station from the USGS station, we called our sensor node Oasis station in the following, named after our project name OASIS [1].

Table 2: Comparison between monitoring stations.

Station / Carac	Digital	GPS	Seismic	Ultrasonic	Lightning
Dual-freq GPS(1)	yes	yes	no	no	no
Short period(2,4)	no	no	yes	(no,yes)	no
Broadband(3)	yes	no	yes	no	no
Oasis station	yes	yes	yes	yes	yes

Station / Carac	Real-time	Self-healing	Power Cons	Money Cost
Dual-freq GPS(1)	no	no	++	++
Short period(2,4)	no	no	+++	++
Broadband(3)	no	no	+++	+++
Oasis station	yes	yes	+	+

Scientific value of the data: USGS volcanologists were satisfied with the data quality (e.g., signal noise ratio) overall. For example, a magnitude 1 earthquake on the volcano was detected on 11/04/2008. In Figure 17, we can see the detected event data by our Oasis station and existing USGS stations. The SEP station is one of the most widespread instrument for volcano monitoring in USGS. We can see that the signal quality of the Oasis station is better than that of the SEP station. Both the SEP station and the Oasis station gave lower signal-noise-ratio than the really expensive broadband station VALT. The broadband station uses completely different technology and the equipment and installation cost is significantly higher: a station costs more than \$10K and requires several days labor cost to install. There is one, and only one, in Mount St. Helens. For the purpose of earthquake event detection and warning, we can see that our system was able to accurately detect earthquake events with precise timestamp.

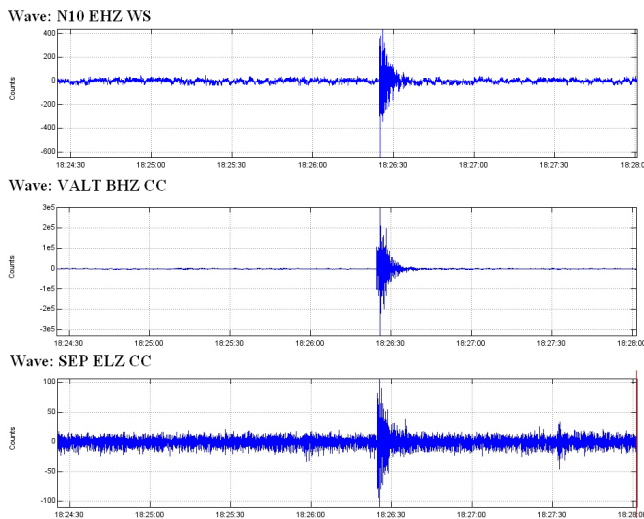


Figure 17: Waveform of detected earthquake on three types of stations: (N10) Oasis station node 10, (VALT) USGS broadband seismic station, (SEP) USGS typical short period seismic station.

Cost of the data: The cost includes the equipment and deployment cost. The gold-standard broadband seismic station costs more than \$10K, while Oasis station costs about \$2K (which includes GPS, infrasonic and lightning sensors, besides seismic sensor). Moreover, all old stations require human on-site intervention to become operational, while our 5 stations were dropped by helicopter. The turn around time for each station was only 10–15 minutes, hence deploying all 5 stations only took 1 hour. In the past, it took USGS several hours, even a day, to deploy a single monitoring station, as it requires scientists to enter the crater, dig a hole, bury geophones in the upright position, and orient the directional antenna to the gateway. The sensor networking technology allows the fast deployment of sensors into dangerous or non-human accessible spots. Also, the maintenance cost and the energy cost of our system is also orders of magnitude lower than all the old stations, which use telemetry for data acquisition and require higher energy consumptions. Table 2 summarizes the major differences. The money cost counts the hardware and deployment labor cost.

The low deployment and maintenance cost is also contributed by the self-organizing and self-healing mesh networking technology. Smart mesh networking not only enables the flexible deployment in locations without line-of-sight, but also provides better fault-tolerance and longer network lifetime. None of the old station technology have the smart networking capability. The sink node is still a bottleneck of the system as it relays the data from the whole network to the gateway. In our future larger-scale deployment, we plan to use multiple sink nodes: if the active sink dies or its bandwidth is saturated, another node can also become a sink to forward data to gateway.

Another advantage of a wireless sensor network, against the old monitoring technology, is the in-network processing capabilities. Instead of processing data in a central computer, the data processing algorithms, like seismic event detection, characterization, timing and localization can be done in the network, and alarms can be triggered when too many seismic events are detected or when eruption is detected. Having such high-level information available in the network has numerous advantages. It allows the network to reduce its load and energy consumption by only sending event information and important data instead of continuous raw data. We are still in the preliminary explorative phase for such in-network detection and characterization algorithms. At this point, volcanologists are demanding continuous raw data our system provides. Their current work involves designing event detection and classification algorithms using the data our system provides.

5. RELATED WORKS

The first-generation sensor network deployments typically have low-duty-cycle and low-data-rate application characteristics. In [8], a tiered sensing system was deployed on the UC James San Jacinto Mountains Reserve to collect dense environmental and ecological data about populations of rare species and their habitats within a mountain stream ecosystem and the surrounding conifer forests and meadows. In the tiered system architecture, data collected at numerous, inexpensive sensor nodes is aggregated by larger, more capable and more expensive nodes. In 2002-2003, researchers used sensor network to monitor the habitat of the Leach’s Storm Petrel [27] at Great Duck Island. It is a representative application with low sampling and bandwidth demands. The results shed light on a number of design issues from selection of power sources to optimizations of routing decisions. On a smaller scale, a sensor network was deployed on a single redwood tree [28] using 33 nodes to cover roughly 50 meters. With this deployment researchers were able to map the differences in the microclimate over a single tree. Those deployments has made valuable contributions in establishing sensor network as a viable platform for habit monitoring and developing essential components for the research community. Other habitat monitoring deployments include tracking Cane Toad populations [24], the movements of zebras [15] and turtles [2].

Some recent deployments involved high-data-rate signals with high-frequency sampling, like our application requirements. In [17], a 64-node sensor network was designed to monitor the structural health of the long-span Golden Bridge. Ambient vibration of the structure is monitored and is used to determine the health status of the structure. Signal processing is used to increase the quality of the sample. Similar

as us, the system includes a protocol for reliable command dissemination, and improvements to software components for data pipelining, jitter control, and high-frequency sampling. Because the nodes are almost on a line in this deployment, the bidirectional antenna is used to provide longer connectivity and reliable communication link. BriMon [9] designs a sensor network system for long term health monitoring of railway bridges and reporting when and where maintenance operations are needed. It uses a simple time synchronization scheme and multiple channel communication. It uses an event detection mechanism to trigger data collection in response to an oncoming train. When volcanology becomes mature enough and can accurately identify events, the data delivery upon event detected may also be designed to save bandwidth and energy. At this point, USGS scientists highly demand real-time continuous raw data.

Recently, researchers also tried sensor network deployment in rugged terrains and under harsh environmental conditions, similar as our deployment. Researchers developed the FireWxNet [11], a multi-tiered portable wireless system for monitoring weather conditions in rugged wildland fire environments. FireWxNet provided the fire fighting community with the ability to measure and view fire and weather conditions over a wide range of locations and elevations within forest fires. The task of deploying the in-situ network was particularly severe, given the rugged mountainous and forested terrain over which FireWxNet was spread. The network covered a unique topology which had not been studied before, ranging from sharp elevational differences to a fairly wide coverage area spanning about 160 square Kilometers. In [3], a seismic network comprises 50 broadband seismic stations along a 500km line across Mexico from Acapulco to Tampico. Stations were placed roughly every 5 km and linked by 802.11 radios with some repeater stations. In this wide area deployment, the network is vulnerable and disruptive, hence a Disruption Tolerant Shell (DTS) mechanism is used to handle network breaks and avoid data loss. In our application, we used the low-power 802.15.4 radio with amplifier, which is sufficient to cover the volcano crater with 1-mile diameter. In the future large-scale volcano deployment, the network disruptions may be a severe issue as well, then we may consider to apply the similar mechanism in transport layer.

Perhaps the most relevant prior research was that done by Welsh and his colleagues [30], who deployed a sensor network on an active volcano in South America to monitor seismic activity related to volcanic eruptions. They did a science-centric evaluation of a 19-day sensor network deployment at the flank of Reventador, an active volcano in Ecuador. They used an event-detection algorithm to trigger on interesting volcanic activity and initiate reliable data transfer to the base station. During the deployment, the network recorded 229 earthquakes, eruptions, and other seismoacoustic events. However, they found their event detection accuracy was only 1%, which justifies the requirement of real-time continuous raw data delivery by USGS. Their network reliability and uptime was relatively low: the mean node uptime was only 69%. This work reveals many hard lessons in volcano monitoring, which greatly helped us to avoid that in our design and deployment. Our work differs from them [30] in many aspects. Our system is the first air-dropped sensor network for long-term volcano monitoring, driven by USGS demands. The goal is to replace existing data logger or telemetry sys-

tems and reduce the deployment risk, time and cost. Second, our system was designed to survive, unattended, for one year and collects raw data continuously. Such longevity and data yield brings with it significant challenges because we must ensure that the system is robust even under harsh environmental conditions and resilient to software and hardware failures. Overcoming these challenges pervade our design. Third, our system was co-designed with USGS scientists to exacting standards – perhaps the most important of which was the need to do time synchronized data acquisition across the sensor network and the need to be online configurable due to the longevity and remoteness of volcano monitoring.

6. CONCLUSIONS AND FUTURE PLANS

Besides the design, test and deployment lessons, we have also learned enormous education and management lessons. For example, when implementing this system, most of our effort was placed on software design and verification. We did not test the radio transmission range of our spider station until we began our campus test. The tough hardware problems exposed in the campus test were time-consuming to investigate and solve, and forced us to postpone our deployment plan. This experience taught us that hardware shall also be verified as early as possible. The learning curve of TinyOS-1.x is sharp. One main reason is that there is no comprehensive learning materials. Instead of describing the rules in TinyOS, it is more effective to explain how it works. For example, we initially had problems to understand how the wiring, event and task works, because it is different from the meaning in other systems. After we discovered that TinyOS code is first preprocessed to generate a C file called *app.c* before compilation, we can easily understand those rules or keywords by examining how they are converted from nesC code to *app.c*.

Aiming to replace data loggers for volcano monitoring, our system design mainly focused on achieving real-time high-fidelity, online reconfigurability, and a high-degree of robustness. The high data yield of our deployed system proves its robustness. The presented system design and deployment experience clears the doubts of domain scientists and proves that the low-cost sensor network system can work in extremely harsh environments. Our next plan is to integrate a localized TDMA MAC protocol [25] and light-weighted compression algorithm [16], and deploy a larger scale sensor network with multiple sinks and multiple channels for real-time volcano monitoring in summer/fall 2009.

Acknowledgments

Many thanks to our shepherd, Maria Ebling, whose suggestions and guidance greatly improved the paper.

7. REFERENCES

- [1] OASIS: Optimized Autonomous Space In-Situ Sensor Web. <http://sensorweb.vancouver.wsu.edu>.
- [2] <http://prisms.cs.umass.edu/dome/turtlenet>.
- [3] <http://research.cens.ucla.edu/areas/2007/Seismic/>.
- [4] <http://sensorwebs.jpl.nasa.gov>.
- [5] <http://www.pbs.org/wnet/nature/forces/lava.html>.
- [6] TinyOS. <http://www.tinyos.net/tinyos-1.x>.
- [7] ION Inc. <http://www.iongeo.com/>

- [8] A. Cerpa, J. Elson, D. Estrin, L. Girod, M. Hamilton, and J. Zhao. Habitat monitoring: Application driver for wireless communications technology. In *1st ACM SIGCOMM Workshop on data communication in Latin America and the Caribbean*, April 2001.
- [9] K. Chebrolu, B. Raman, N. Mishra, P. K. Valiveti, and R. Kumar. Brimon: A sensor network system for railway bridge monitoring. In *The 6th Annual International Conference on Mobile Systems, Applications and Services (MobiSys)*, June 2008.
- [10] S. Ganeriwal, R. Kumar, and M. B. Srivastava. Timing-sync protocol for sensor networks. In *Proc. 1st ACM conference on Embedded networked sensor systems (SenSys)*, November 2003.
- [11] C. Hartung, R. Han, C. Seielstad, and S. Holbrook. Firewxnet: A multi-tiered portable wireless system for monitoring weather conditions in wildland fire environments. In *The 4th International Conference on Mobile Systems, Applications, and Services (MobiSys)*, June 2006.
- [12] D. Herbert, V. Sundaram, Y.-H. Lu, S. Bagchi, and Z. Li. Adaptive correctness monitoring for wireless sensor networks using hierarchical distributed run-time invariant checking. *ACM Trans. Auton. Adapt. Syst.*, 2(3), September 2007.
- [13] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister. System architecture directions for networked sensors. *SIGOPS Oper. Syst. Rev.*, 34(5):93–104, December 2000.
- [14] J. Hui and D. Culler. The dynamic behavior of a data dissemination protocol for network programming at scale. In *Proc. 2nd ACM Conference on Embedded Networked Sensor Systems (SenSys)*, November 2004.
- [15] P. Juang, H. Oki, Y. Wang, M. Martonosi, L. Peh, and D. Rubenstein. Energy efficient computing for wildlife tracking: Design tradeoffs and early experiences with zebranet. In *Proc. 10th Tenth International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, October 2002.
- [16] A. Kiely, M. Xu, W.-Z. Song, R. Huang, and B. Shirazi. Adaptive linear filtering compression on realtime sensor networks. In *The 7th Annual IEEE International Conference on Pervasive Computing and Communications (PerCom)*, March 2009.
- [17] S. Kim, S. Pakzad, D. Culler, J. Demmel, G. Fenves, S. Glaser, and M. Turon. Wireless sensor networks for structural health monitoring. In *Proc. 4th ACM conference on Embedded networked sensor systems (SenSys)*, November 2006.
- [18] P. Levis, N. Patel, D. Culler, and S. Shenker. Trickle: A self-regulating algorithm for code propagation and maintenance in wireless sensor networks. In *Proc. 1st USENIX/ACM Symposium on Networked Systems Design and Implementation (NSDI)*, November 2004.
- [19] M. Maróti, B. Kusy, G. Simon, and A. Lédeczi. The flooding time synchronization protocol. In *Proc. 2nd ACM conference on Embedded networked sensor systems (SenSys)*, November 2004.
- [20] T. L. Murray and E. T. Endo. A real-time seismic-amplitude measurement system (rsam). volume 1966 of *USGS Bulletin*, pages 5–10. 1992.
- [21] Y. Peng, R. Lahusen, B. Shirazi, and W. Song. Design of smart sensing component for volcano monitoring. In *The 4th IET International Conference on Intelligent Environments (IE)*, July 2008.
- [22] Y. Peng, W. Song, R. Huang, M. Xu, and B. Shirazi. Cacades: a reliable dissemination protocol for data collection sensor network. In *IEEE Aerospace Conference*, March 2009.
- [23] R. Scarpa and R. I. Tilling. *Monitoring and Mitigation of Volcano Hazards*. Springer, 1 edition, January 1996.
- [24] S. Shukla, S. Shukla, N. Bulusu, N. Bulusu, S. Jha, and S. Jha. Cane-toad monitoring in kakadu national park using wireless sensor networks. In *Networks Research Workshop*, July 2004.
- [25] W.-Z. Song, R. Huang, B. Shirazi, and R. Lahusen. TreeMAC: Localized tdma mac protocol for high-throughput and fairness in sensor networks. In *The 7th Annual IEEE International Conference on Pervasive Computing and Communications (PerCom)*, March 2009.
- [26] K. Srinivasan and P. Levis. RSSI is under appreciated. In *Proc. 3rd Workshop on Embedded Networked Sensors (EmNets)*, May 2006.
- [27] R. Szewczyk, J. Polastre, A. Mainwaring, J. Anderson, and D. Culler. Analysis of a large scale habitat monitoring application. In *Proc. 2nd ACM Conference on Embedded Networked Sensor Systems (SenSys)*, November 2004.
- [28] G. Tolle, J. Polastre, R. Szewczyk, D. Culler, N. Turner, K. Tu, S. Burgess, T. Dawson, P. Buonadonna, D. Gay, and W. Hong. A macroscope in the redwoods. In *Proc. 3rd ACM Conference on Embedded Networked Sensor Systems (SenSys)*, November 2005.
- [29] C.-Y. Wan, A. T. Campbell, and L. Krishnamurthy. PSFQ: a reliable transport protocol for wireless sensor networks. In *Proc. 1st ACM International workshop on Wireless sensor networks and applications*, September 2002.
- [30] G. Werner-Allen, K. Lorincz, J. Johnson, J. Lees, and M. Welsh. Fidelity and yield in a volcano monitoring sensor network. In *Proc. 7th USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, November 2006.
- [31] K. Whitehouse, G. Tolle, J. Taneja, C. Sharp, S. Kim, J. Jeong, J. Hui, P. Dutta, and D. Culler. Marionette: Using rpc for interactive development and debugging of wireless embedded networks. In *The 5th International Conference on Information Processing in Sensor Networks (IPSN)*, April 2006.
- [32] A. Woo, T. Tong, and D. Culler. Taming the underlying challenges of reliable multihop routing in sensor networks. In *Proc. 1st ACM conference on Embedded networked sensor systems (SenSys)*, November 2003.
- [33] F. Yuan, W.-Z. Song, N. Peterson, Y. Peng, L. Wang, and B. Shirazi. Lightweight sensor network management system design. In *The 4th IEEE International Workshop on Sensor Networks and Systems for Pervasive Computing*, March 2008.