The Hitchhiker's Guide to Successful Remote Sensing Deployments in Mongolia

Lehi Alcantara*, Joseph Miera†, Batsaikhan Ariun-Erdene†, Chia-Chi Teng*, and Philip Lundrigan†

*Information Technology & Cybersecurity
Brigham Young University
Email: lehi_alcantara@byu.edu, ccteng@byu.edu

†Electrical and Computer Engineering Department
Brigham Young University
Email: joseph.r.miera@gmail.com, bat@byu.edu, lundrigan@byu.edu

***Abstract*—The health hazard of air pollution in developing countries poses a significant threat of cardiovascular, respiratory, and other diseases. Ulaanbaatar, Mongolia is among cities with the worst polluted air in the world due to the use of coal as the primary heating source in the traditional Mongolian gers where most of the local population resides. Humanitarian groups are looking for ways to improve air quality, but are unable to measure the effects of their solutions. We build a low-cost air quality sensor that can upload data in real-time in remote locations. This newly developed sensor allows for *real-time* air quality monitoring and tracking that was not possible before in such locations. We present the implementation and deployment of this system and share experiences and lessons learned from deploying the sensors in such a unique location.**

***Index Terms*—wireless sensor networks, remote sensing, system applications and experience**

I. Introduction

Poor air quality is a problem that knows no boundaries. It affects populations all over the world [1] [2]. Epidemiological studies show that ambient air pollutants like PM, O_3, SO_2, and NO_2 are contributors to several respiratory problems including: bronchitis, emphysema and asthma [3]. One area of the world that is hit particularly hard by poor air quality is Mongolia.

Traditionally Mongolian families that live in gers (Figure 1) use coal to both cook meals, as well as heat up the gers during the winter months where temperature can drop as low as $-40°$ C. The burning of coal inside the ger causes hazardous amounts of particles that are harmful to the residents of the ger. In the winter months, 80% of Ulaanbaatar's air pollution is caused by households burning raw coal in stoves in ger districts. Mongolia has a population of 3 million people, and in 2016 an estimated 1800 people died from diseases attributable to household air pollution, and more than 1500 people died from diseases due to outdoor air pollution [4].

Organizations such as Deseret International Charities and the Gerhub [5] are looking for ways of improving the ger structure to be more energy-efficient, thus requiring less coal (or no coal) to be burned and improving air quality. Such ideas include building new structures instead of using traditional gers or retrofitting current gers [6]. By making the ger more energy efficient, an electric heater can be used instead of a coal stove. Though these solutions seemingly provide a great benefit to the people living in the gers and the community as a whole, no *indoor* air quality data has been collected in Mongolia to measure the effects of these modifications to gers and quantify the benefits.

Fig. 1. A Mongolian ger where we deployed air quality sensors.

Many commercial air quality sensors exist [7]; however, deploying air quality sensors in this context requires special considerations. First, the cost of the device needs to be as low as possible. Since these projects are typically humanitarian-based, there is not much money to spend on instrumentation. Second, it is vital to be able to monitor the data in real-time. Real-time data allows maintainers to know when the sensors are unplugged or malfunctioning and fix the problem. Without real-time data capability, data collection becomes challenging (you must send someone to every sensor to download the data off of the device), and we would not know of any problems with the sensors until after all the data is collected. Third, most commercial sensors that are real-time use WiFi to upload the data. In the ger districts of Mongolia, there is no

WiFi connectivity as the residents use cellular data through their smartphones instead. To the best of our knowledge, a sensor that fulfills these requirements (an inexpensive real-time sensor that uploads data through cellular) is not commercially available.

To solve this problem, we design, build, and deploy air quality sensors in Mongolia to collect data to help in these humanitarian efforts. The air quality sensor we design costs $200 to build and uses cellular connectivity to upload data in real-time. This sensor has implications beyond Mongolia. Our sensor can be used anywhere WiFi connectivity is not available, such as parks, bus stops, and along roadways, breaking the constraints that other low-cost sensors have. We believe removing the need for WiFi is a necessary step in allowing ubiquitous air quality sensing.

We make several contributions in this paper: First, we present our sensor design and system architecture. Second, since Mongolia offers a unique environment and constraints, we share experiences we had in deploying sensors in a remote location like Mongolia. Our experience goes beyond air quality sensors and can help anyone who is deploying sensors in remote areas.

The rest of the paper is organized as follows: In Section II, we discuss related projects and air quality sensors. In Section III, we give an overview of our sensor design and system architecture. In Section IV, we share the challenges we faced in deploying our sensors and lessons learned along the way. In Section V, we share our conclusions.

II. Related Work

Many commercially available air quality sensors already exist [7]. Of particular interest are the Purple Air [8] and AirU [9] sensors because they are low-cost (around $250 USD). These sensors have been used in a large number of studies and have been deployed in many places around the world [10]. However, these two sensors use WiFi to report data in real-time. Since WiFi is not a viable option in Mongolia and other remote locations, these sensors are not well suited. Our deployment requires a low-cost stationary sensor that must upload data in real-time.

There have been platforms and architectures designed to handle air quality data, such as EpiFi [11] [12]. EpiFi is a system designed for epidemiological research/study purposes. It focuses on deploying sensors in homes with participants. Since it is geared towards large scale studies, it is too elaborate for what we are trying to achieve. However, our system could easily be integrated into a system like EpiFi.

Other work has been done on sharing experiences and lessons learned related to similar topics. Hnat et al. shared their experience of residential sensing deployments [13]. They share deployment issues including running out of wall sockets, wireless connectivity issues, environmental hazards such as children, pets, and robotic vacuums, participants dropping from the study, and maintenance challenges due to combinations of commercial and custom designed devices. We faced many similar issues. Similarly, Barrenetxea et al. shared their experience with wireless sensor network deployments [14]. We hope to add our experience of deploying air quality sensors in Mongolia to anyone thinking of deploying sensors in remote locations.

Fig. 2. Our air quality sensor, with its cover off, deployed in a ger.

III. Implementation

"Simplicity is prerequisite for reliability."
— Edsger Dijkstra

"Everything should be made as simple as possible, but not simpler."
— Albert Einstein

The goal of our air quality sensor and architecture is to be as simple as possible. We believe that simplicity will lead to a more reliable design. Reliability is very important since these sensors will be deployed far from us in remote locations. We achieve simplicity in two ways: using as few components as necessary and by doing as little custom work as possible. We next describe our air quality sensor design and system architecture.

A. Air Quality Sensor

Our air quality sensor consists of five major sensor components: particulate sensor, CO_2 sensor, real-time clock (RTC), SD card, and Particle Boron. We design a *simple* custom circuit board to connect all of the components together. This is the only custom part of our sensor and only provides the interconnect between different components. Every other component we used can be bought off the shelf. We discuss each component individually:

Particulate and CO_2 sensors. The particulate sensor is the Sensirion SPS30. It measures the number of particles in the air at different sizes. This is denoted by PM_X, where X is the size of the particle in microns. This sensor measures four different sizes of particles: PM_1, $PM_{2.5}$, PM_4, and PM_{10}. The CO_2 sensor is the Sensirion SCD30 and measures the concentration of CO_2 as well as temperature and humidity.

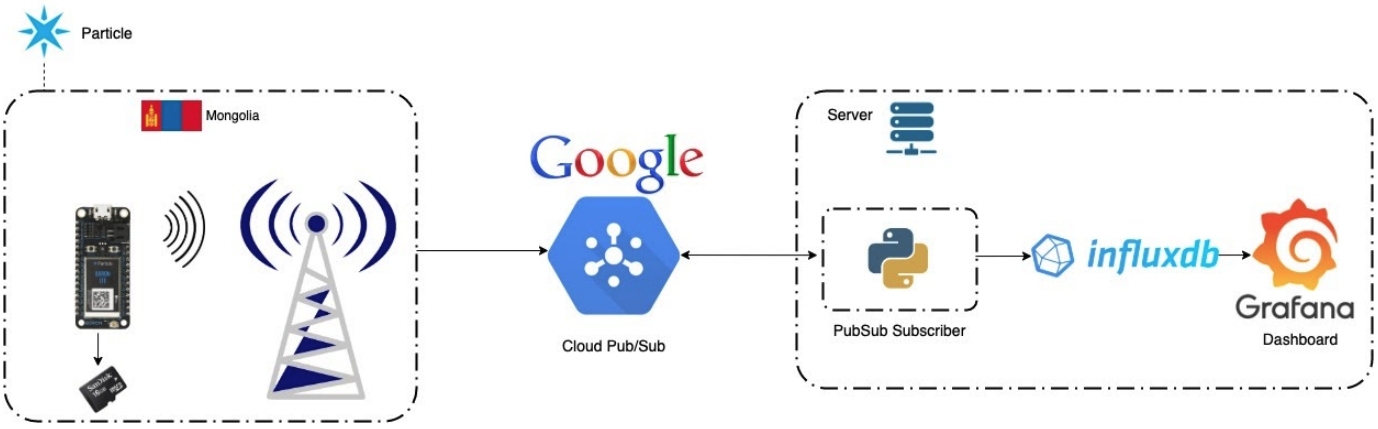

Fig. 3. Air Quality sensors architecture diagram. Sensors readings are published to Particle Cloud, which is then sent to Google's Pub/Sub service. Our script pulls data from the Pub/Sub and writes it to InfluxDB.

Real-time Clock (RTC). The RTC is essential to ensure that our sensor has accurate time even if it is not connected to the Internet. When a sensor is connected to the Internet, it is able to get an accurate time. However, if the sensor gets rebooted and it is not connected to the Internet, it will have the wrong time. Any data that is collected from the time of the reboot to when the sensor connects to the Internet will have the wrong timestamp. This is not a problem if the sensor connects to the Internet instantly after rebooting. However, our sensors are being deployed in a remote location where connectivity might not be good, so we can not expect constant Internet connectivity. Including an RTC improves the reliability of our data.

SD Card. The SD card is used to store data persistently. This is another important design decision to improve the reliability of our sensor and the data it collects. By including persistent storage, our sensor is able to safely store data it has collected while it is waiting to upload the data. Without the SD card, the sensor would store this data in non-persistent storage (i.e., RAM), and if the sensor got rebooted, all the data would be lost. To ensure that no collected data would be lost, we store all readings onto an SD card first. After it has been written to the SD card, then we upload the data to the Internet. For our deployment, we used 16 GB SD cards, which can store years worth of data when sampling every one minute.

Particle Boron. The Boron is a microprocessor that integrates cellular connectivity. There are two types of Borons: 3G/2G and LTE. We select the 3G/2G version because it supports global deployment, whereas the LTE version only supports North America. The Boron includes an embedded SIM card that allows it to work in most countries in the world. We discuss why we selected cellular instead of a different wireless technology in Section IV-B. We program the Boron to communicate with the particulate matter sensor, CO_2 sensor, RTC, and SD card through I2C and SPI interfaces. Particle, the company that makes the Boron, also provides software to easily upload data from the Boron to their cloud, which is done by "publishing" the data. We use this mechanism to send our data from our sensor to the Internet.

Data is collected from the sensors as follows: every minute the Boron reads from each of the sensors and writes this data to the SD card. Every ten minutes, any data on the SD card that has not been previously uploaded gets uploaded. This approach allows for our sensors to store data until it becomes connected to the Internet, which is critical for intermittent connectivity. Our sensor is shown, with the cover off, deployed in a ger in Figure 2.

B. Architecture

Next, we discuss the architecture of our system, which is shown in Figure 3. As stated earlier, the major focus of our architecture is simplicity and reliability. We achieve this by developing as few components as possible ourselves and instead leverage other existing technologies. We want to ensure that no data is lost between publishing the data and storing the data in our database. There are five major components to our architecture.

Data Ingress. As mentioned in the previous section, Particle provides the microprocessor, Boron, and the ability to publish data from the sensor to their cloud service over the cellular network. Using Particle's service allows us to monitor deployed sensors and check on their status. We are also able to push over-the-air updates to the sensors. Particle provides an integration into Google Pub/Sub. Any data that is sent to Particle's servers (through publishing the data) is sent to Google Pub/Sub.

Data Broker. Google Pub/Sub is part of Google Cloud services. It is a message broker that keeps the data in the queue until data is requested (pulled). Data that is pulled from Pub/Sub must be acknowledged before the service will forget about the data. If it is not acknowledged, then it will continue to store that data. This service provides two great benefits. First, it allows us to leverage Google's reliability and scale. If Particle sends data directly to our server and our server is down, then data would be lost. Pub/Sub provides

a safe and reliable place for us to put the data. Second, it decouples the production of the data (sensors uploading data) from the consumption of the data (storing the data in InfluxDB). Though Pub/Sub is not required for a system like this to work, we believe the scalability and reliability benefit outweigh the cost.

Subscriber. This is a Python script that pulls data from Google Pub/Sub, transforms the data, and saves the data to InfluxDB. If the data insertion is successful, the script sends an acknowledgment to Pub/Sub, notifying it that the data can be deleted. This is the only custom component of the architecture that we created.

Storage. For data storage, we use InfluxDB [15]. InfluxDB is a popular open-source database that is designed for time-series data, such as sensor measurements. It provides good scalability as well as other features when working with time-series data.

Visualization. We use Grafana [16] to visualize data that is stored in InfluxDB. Grafana is an open-source visualization tool that integrates with many different databases. It allows you to set up dashboards and graphs to monitor data. It is essential that we monitor the collected data to ensure that all sensors are uploading data correctly and that the sensor readings look reasonable. If a sensor is not uploading data or the sensor readings are not reasonable, we must determine the reason or remotely debug the sensor. We provide a public dashboard for people in Mongolia to monitor the air quality, as well as private dashboards for debugging purposes.

The flow of data goes as follows. Every 10 minutes, sensors publish data to the Particle cloud. Our sensor will continue to resend the data until it has received an acknowledgment from Particle that the data has been received successfully. Particle then forwards this data to Google's Pub/Sub service. This is set up through an integration between Particle and Google. Our Pub/Sub subscriber checks Pub/Sub for new data. When there is new data, it pulls the data, and inserts the data into InfluxDB, sending an acknowledgment to Pub/Sub that the data was inserted successfully. Data can be viewed using Grafana to check its validity and make sure sensors are uploading data properly. We develop a few dashboards to analyze and compare the data being sent from the air quality sensors. One dashboard was publicly created for the purpose of sharing the data with the people in Mongolia, as shown in Figure 4. We use Grafana to send us alerts whenever there is missing data.

IV. Challenges

In September 2019, we deployed 50 air quality sensors in gers in Ulaanbaatar, Mongolia. Since that time, we have collected over 5.5 million air quality samples. While designing, deploying, and maintaining, we ran into numerous challenges. We summarize some of the problems we faced below as well as the lessons learned in dealing with these challenges. These challenges are not unique to us or this type of deployment and can be generalized to all IoT sensor deployments.

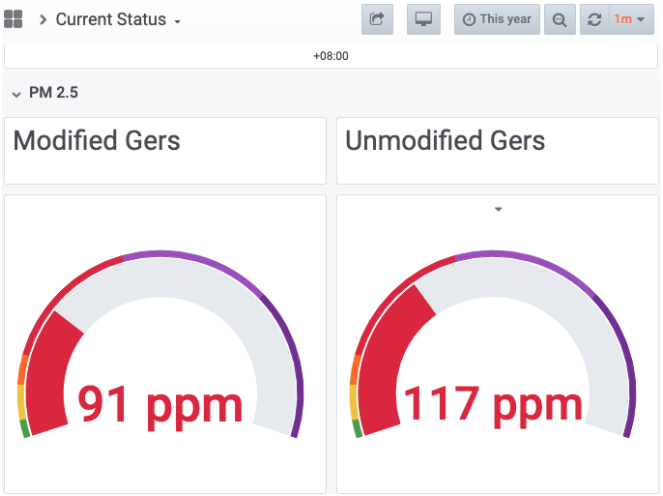

Fig. 4. Grafana public dashboard.

A. Unplugged Sensors

Since we initially deployed our sensors, there have been several instances when deployed sensors were sending air quality data and suddenly stopped. We were perplexed as to why this was happening. We eventually discovered that the participants were unplugging the sensors. This quickly became a common occurrence. We looked through our collected data to determine how many times our sensors were getting unplugged. On average, 22% of our sensors are being unplugged very often—some on average multiple times a day. These results are shown in Figure 5. We did not anticipate that our sensors would be unplugged that often and had not accounted for this in our design.

Lessons Learned. From this experience and data, we learned two lessons. First, we should have designed the sensor to require as little from the participant as possible. By using their energy, we depend on them to keep it plugged in at all times. Put another way, following Murphy's law, "whatever can go wrong, will go wrong" when deploying a sensor in a home. We could have, for example, used a small battery to keep the sensor powered while it is unplugged temporarily. Second, when deploying the sensors, we used one of the participant's power plugs. Given the nature of gers, there are no set power outlets to plug into. Each home is wired depending on the residents' abilities and preferences. This makes power plugs more transient and sparse compared to a traditional American home. In retrospect, we should have provided power strips so that we would not use one of the few open outlets in the ger.

B. Cellular vs LoRa

A critical part of our whole system is the ability to upload data from the sensor to the Internet. There are many different approaches to this. The conventional method is to use existing WiFi architecture, as used by Purple Air and AirU. While this is a good approach for some contexts, WiFi is not widely deployed in all parts of the world, so we can not require it

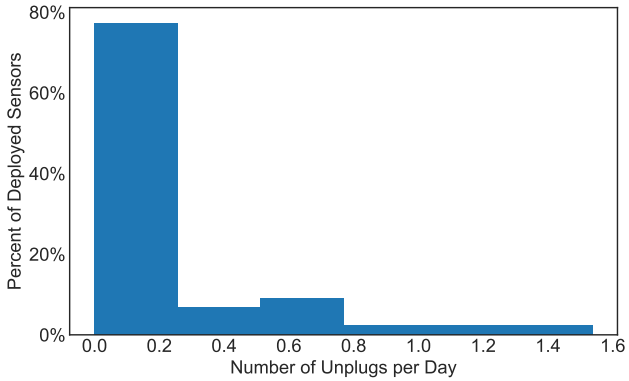

Fig. 5. A histogram showing the amount of times deployed sensors get unplugged.

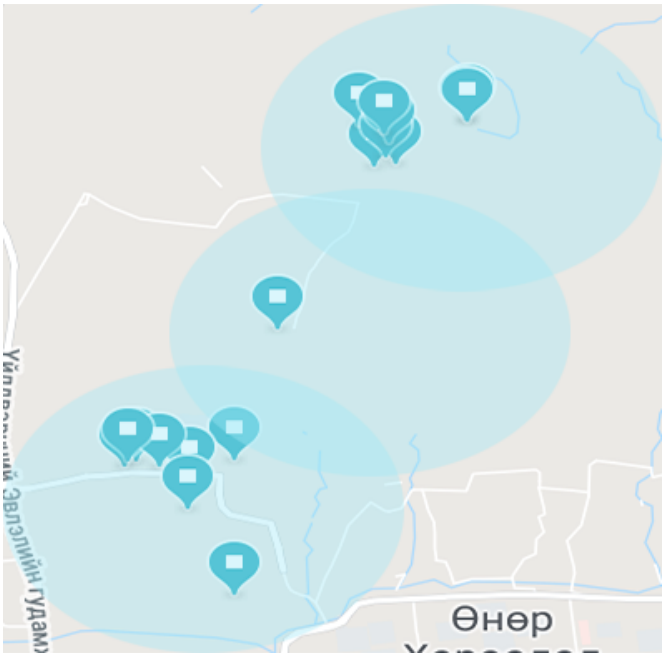

Fig. 6. Potential locations to place LoRa gateways to get complete coverage of the area where we are deploying sensors.

for our sensor to work. Using a cellular network is another option for uploading data. While this is more widely available than WiFi, the cellular hardware is more expensive, and using the cellular network costs money. A third option is to use a different wireless protocol like LoRa [17]. LoRa is a relatively new wireless protocol that is designed to have long-range, but with little power usage. LoRa clients upload data to LoRa gateways, which are connected to the Internet. It is designed to get similar ranges as cellular but without the extra cost.

In designing our sensor, we explored using LoRa because it could potentially be lower cost than using cellular. However, after investigating this option further, there are two critical drawbacks to using LoRa when deploying in remote areas. First, since there is no existing LoRa infrastructure in Mongolia, LoRa gateways must be deployed. In some parts of the world (mainly Europe), LoRa gateways are available and free to use by anyone [18]; however, in Mongolia, such a network does not exist. To deploy LoRa gateways, it requires knowledge about where you are deploying and to get maximum coverage, a site survey. In a typical LoRa network, range depends on many factors such as indoor/outdoor gateways, the type of antenna used, etc. On average, in an urban environment with an outdoor gateway, you can expect up to 2 to 3 km wide coverage [19]. Looking into deploying outdoor gateways with LoRa in the area we were planning on deploying sensors, it would require at least three gateways to get coverage of the area. Figure 6 demonstrates potential locations to place LoRa gateways. Since we are not local to Mongolia, it is difficult to plan for a LoRa deployment beforehand.

The second issue with deploying LoRa gateways is its infrastructural reliability. If a gateway goes down, all of the sensors that are communicating with that gateway are no longer able to upload data. Also, because of how critical the gateway is to the LoRa network, the gateway should be placed in a location where it has reliable power, good height (for maximum coverage), and restricted access. Without these, a LoRa deployment is not going to work reliably. Given these requirements, this was not a reasonable approach for deploying sensors in remote locations.

Lesson Learned. From doing a thorough analysis, we learn that simplicity is the way to go. Though adding LoRa would be cheaper in the long run, having to manage our own network would be brittle and error-prone because there would be one or two points of failure. By using cellular, we are using a network that is well provisioned and managed. Each sensor is self-contained and does not depend on another sensor. Because of the remote location of our deployment, reliability is one of the most important aspects of our deployment.

C. Management Tools

In developing our system and deploying our sensors, it quickly became apparent that collecting data and storing the data is not enough. A system like this requires many management tools. For example, it is important to be able to check on the status of sensors, debug sensors, visualize data, get notified when a sensor has gone offline, etc. Some of these tools exist already, such as Grafana, but many of them do not. Even the tools that do exist are geared towards people with computer backgrounds. This increases the complexity of the system and puts a more significant burden on the maintainers of the system. We break these management tools into different categories and discuss them individually.

Metadata Management. When deploying sensors in a real environment, there is a lot of additional data about each sensor, which we call metadata. This is data that is not being collected by the sensor but is still necessary to understand the data that the sensors are collecting. This includes the GPS location of the sensor, household contact information for the ger the sensor is deployed in, the firmware version the sensor is on, relevant information about the family (such as if someone smokes in the house), etc. It is challenging to know

where to put this data once it has been collected and how to correlate it with the sensor measurements. It becomes even more challenging when this information changes. For example, if a sensor is broken and needs to be replaced with a different sensor, metadata must be updated in a timely manner to reflect this change. If not, data will be attributed to the wrong GPS location or household. To the best of our knowledge, there are no existing tools to help sensor deployers and maintainers deal with these challenges. Ensuring that all metadata is accurate and up to date requires a lot of manual work by a deployer and is often overlooked when planning out a sensing deployment.

Visualization and Data Monitoring Tools. Since there is so much data being generated from our air quality sensors, we need a way to aggregate all that data stored in InfluxDB database and make some sense of it. As stated in Section III-B, we use Grafana as that representation layer. It enables us to create different dashboards used internally, as well as making it available to the public to be able to see the air quality data. Grafana allows us to check on the state of a sensor and view its historical data. However, it quickly became untenable to check on all of our sensors to see if they were functioning properly. Therefore, we looked into ways to alert us when a sensor went offline. Grafana has alerting functionality out of the box that we could utilize in order to send alert notifications to us. While this helped with sensors that went offline, we quickly realized that it did not help with another class of problems: missing data and bad data. Missing data is when not all of the sensors of the device are working properly. For example, with our sensor, the CO_2 sensor is reporting data, but the PM sensor might be malfunctioning and not reporting data. Bad data is when a sensor is reporting data, but the data is not in an acceptable range for the sensor. Though it is possible to use Grafana to inspect the data manually, this is not scalable to large deployments of sensors. Better tools must be created to help deal with large deployments of sensors.

Over-the-air (OTA) Updates. OTA updates quickly became an invaluable tool after we deployed the sensors and realized there was a critical bug in the system that we had not found in our testing. Being able to upgrade software on the device remotely is a critical part of such a system. Luckily, Particle provides the ability to update devices remotely, so we did not have to write this component ourselves.

Lesson Learned. For any successful long-term deployment of sensors, collecting data is not enough. Data collection must be coupled with tools that help manage and monitor the sensors. We have highlighted a few tools that exist to aid in this area. However, there is a lot left to be desired. In particular, there are significant gaps in metadata management tools and data monitoring tools. We firmly believe for a deployment of sensors to be successful, such tools, as outlined above, must exist.

V. Conclusion

In this paper, we present a low-cost air quality sensor that we develop and deploy in Mongolia. This sensor removes the constraints of WiFi and allows for real-time air quality data monitoring in remote locations. We present our system architecture for processing and storing the data. We share our experiences and lessons learned while deploying and managing the sensors. These lessons are valuable to other researchers who would deploy sensors in remote locations.

References

[1] D. Carrington and M. Taylor. (2018) Air pollution is the 'new tobacco', warns who head. [Online]. Available: https://www.theguardian.com/environment/2018/oct/27/air-pollution-is-the-new-tobacco-warns-who-head

[2] S. Sengupta. (2018) Air pollution is shortening your life. here's how much. [Online]. Available: https://www.nytimes.com/2018/08/22/climate/air-pollution-deaths.html

[3] N. A. B. Mabahwi, O. L. H. Leh, and D. Omar, "Human health and wellbeing: Human health effect of air pollution," *Procedia - Social and Behavioral Sciences*, vol. 153, pp. 221 – 229, 2014, aMER International Conference on Quality of Life, AicQoL2014KotaKinabalu, The Pacific Sutera Hotel, Sutera Harbour, Kota Kinabalu, Sabah, Malaysia, 4-5 January 2014. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1877042814054986

[4] B. of the World Health Organization. (2019) Bulletin of the world health organization. [Online]. Available: https://www.who.int/bulletin/volumes/97/2/19-020219/en/

[5] (2018) Gerhub. [Online]. Available: https://gerhub.org

[6] (2019) A ger with less brr. [Online]. Available: https://magazine.byu.edu/article/a-ger-with-less-brr/

[7] U. S. E. P. Agency. (2020) Evaluation of emerging air pollution sensor performance. [Online]. Available: https://www.epa.gov/air-sensor-toolbox/evaluation-emerging-air-pollution-sensor-performance

[8] PurpleAir. Purpleair: Real time air quality monitoring. [Online]. Available: https://www2.purpleair.com

[9] K. Kelly and P.-E. Gaillardon. (2017) Airu at the university of utah. [Online]. Available: https://airu.coe.utah.edu/about/

[10] S. Hegde, K. T. Min, J. Moore, P. Lundrigan, N. Patwari, S. Collingwood, A. Balch, and K. E. Kelly. (2017) Household indoor particulate matter measurement using a network of low-cost sensors. [Online]. Available: http://www.aaqr.org/files/article/9041/AAQR-19-01-LCS-0046_accepted.pdf

[11] P. Lundrigan, K. T. Min, N. Patwari, S. K. Kasera, K. Kelly, J. Moore, M. Meyer, S. C. Collingwood, F. Nkoy, B. Stone, and K. Sward, "Epifi: An in-home iot architecture for epidemiological deployments," in *2018 IEEE 43rd Conference on Local Computer Networks Workshops (LCN Workshops)*, Oct 2018, pp. 30–37.

[12] J. Moore, P. Goffin, M. Meyer, P. Lundrigan, N. Patwari, K. Sward, and J. Wiese, "Managing in-home environments through sensing, annotating, and visualizing air quality data," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 2, no. 3, p. 128, 2018.

[13] T. W. Hnat, V. Srinivasan, J. Lu, T. I. Sookoor, R. Dawson, J. Stankovic, and K. Whitehouse, "The hitchhiker's guide to successful residential sensing deployments," in *Proceedings of the 9th ACM Conference on Embedded Networked Sensor Systems*, ser. SenSys '11. New York, NY, USA: Association for Computing Machinery, 2011, p. 232–245. [Online]. Available: https://doi.org/10.1145/2070942.2070966

[14] G. Barrenetxea, F. Ingelrest, G. Schaefer, and M. Vetterli, "The hitchhiker's guide to successful wireless sensor network deployments," in *Proceedings of the 6th ACM Conference on Embedded Network Sensor Systems*, ser. SenSys '08. New York, NY, USA: Association for Computing Machinery, 2008, p. 43–56. [Online]. Available: https://doi.org/10.1145/1460412.1460418

[15] [Online]. Available: https://github.com/influxdata/influxdb

[16] [Online]. Available: https://github.com/grafana/grafana

[17] [Online]. Available: https://www.lora-alliance.org/

[18] [Online]. Available: https://www.thethingsnetwork.org

[19] J. Petajajarvi, K. Mikhaylov, A. Roivainen, T. Hanninen, and M. Pettissalo, "On the coverage of lpwans: range evaluation and channel attenuation model for lora technology," in *2015 14th International Conference on ITS Telecommunications (ITST)*, Dec 2015, pp. 55–59.