influxdata®

AN INFLUXDATA CASE STUDY

# How Unix Edge Collects Performance Data to Keep Chemical Plants and Refineries Humming

**Frank Inselbuch**

President, Unix Edge


FEBARUARY 2022 (REVISION 1)

## Company in brief

Based in Houston, Texas, Unix Edge designs, builds and operates Managed Solutions that combine cloud platforms and Edge Devices to deliver tangible business value in the form of actionable, real-time information.

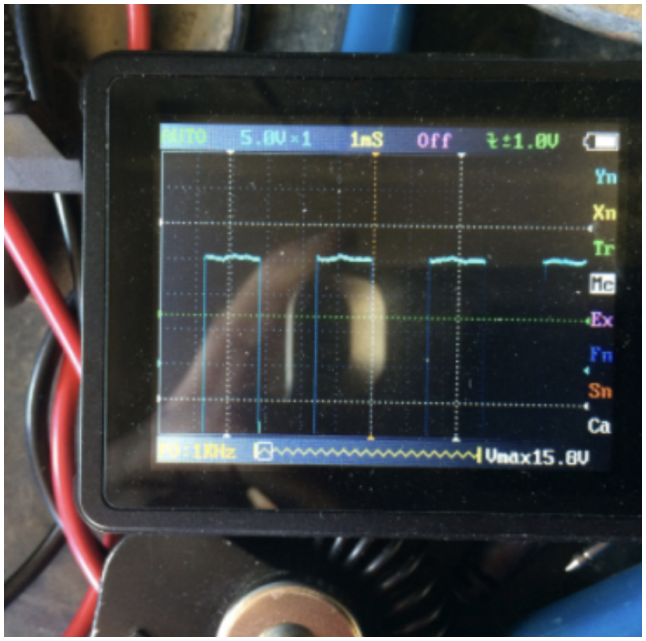
Unix Edge Devices collect virtually any data from any location. They support analog, digital and optical sensors including GPS, temperature, pressure, humidity, hall effect, proximity, and even analog gauges. Support for additional sensors is being added all the time.

Unix Edge Devices are customized for clients' specific needs using off-the-shelf components and are inexpensive, customer installable, with minimal power requirements. Prototypes are built in-house. Unix Edge manages the production of the final units through its network of manufacturing partners.

## About Unix Edge

Unix Edge wanted to build a reliable fleet surveillance solution to collect performance metrics for the industrial machines at their customers' chemical plants and refineries. For the oil and gas industry, real-time remote equipment monitoring is critical given the high cost of machinery downtime and dispersed geographical locations. Unix Edge meet that need by providing specialized Edge Devices to their customers that support analog, digital and optical sensors including GPS, temperature, pressure, humidity, hall effect, proximity, and even analog gauges.

Unix Edge store the data collected in an InfluxDB database and use Grafana to build custom dashboards for their customers, delivering cost savings, added efficiency, and improved safety.




**Measuring signals from field equipment**

*“If I were to describe InfluxDB in three words, I would say: performant, flexible, and cooperative.”*

**Frank Inselbuch**, President

## The business problem

Remote monitoring, diagnostics and surveillance of equipment were once viable only for expensive, critical equipment such as gas turbines on offshore platforms, airplane engines, and anything that might interrupt industrial operations. Today, with the rise of the Industrial Internet of Things (IIoT) and the cloud, along with lower monitoring technology costs, remote monitoring is viable for anything to which a node, or edge device, can be connected.

Unix Edge set out to develop a device for remote monitoring of oil and gas equipment. The remote monitoring business opportunity for this sector is compelling. For example, offshore oil and gas producing platforms host equipment and have to produce their own electric power since they are miles out into the ocean, using a gas turbine. If the turbine goes down, they lose electric power and can't produce oil. The Thunderhorse platform in the Gulf of Mexico alone, depending on the price of oil, could lose $43,000 per minute during downtime. Remote monitoring for such equipment is absolutely critical. That same monitoring technology could be used for less costly industry equipment, such as oil and gas vacuum trucks and compressors.

**Vacuum Truck**


**Compressor**

## The technical problem

In theory, the challenge of building fleet surveillance IIoT devices seemed to require only a few weeks of effort and a hundred dollars given the low cost of hardware components, open source software, and connectivity. In practice though, it was hard to get all the details right:

- Hardware failure
- Cellular Connectivity
- Data Encryption
- Cybersecurity
- Software Updates
- Device Provisioning
- Vibration
- Temperature
- Backups
- Time Synchronization

The Unix Edge technical journey involved taking measures to mitigate risks and make the system as reliable as it needs to be for production. They learned and experienced in the field, working with the system to perfect it. The decisions they made regarding what platform, software, and pieces of hardware to use—their keys to reliability—were the following, and they set out to find an optimal solution for each:

- Platforms
- Media
- Self-Provisioning
- Code Management & Collaboration
- Redundancy
- Testing
- Cybersecurity
- Power
- Buy, don’t build software (where possible)

## Results

The digital twin is the culmination and end-product of Graphite Energy's time series data management process. They use time series data to create a real-time digital model of a TES unit that is accurate to within about 5% of actual machine performance. Digital twin lets Graphite Energy roll forward and backwards in time to track device performance and is becoming a very powerful part of their predictive toolkit for production optimization.

## The solution

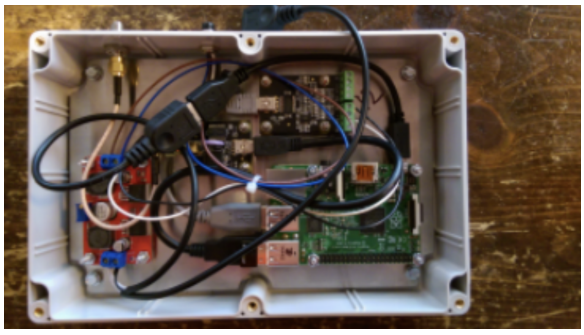
> *"One of the things you want to do when you're building a solution, and this is where InfluxData and InfluxDB have been really terrific, is I don't want to have to do a lot of manual steps when one of my Unix Edge Devices comes online."*

## Why InfluxDB?

Unix Edge chose InfluxDB Time Series Database because it provided the performance and features they wanted, which allowed them to avoid constant configuration:

- Using InfluxDB, they turn the Unix Edge device on, and it will self-identify using the hardware's unique identifier. They don't have to pre-provision, pre-allocate, or pre-configure anything in their schema. To add a new field (if the customer added a new instrument, for example) they don't have to change the schema or do any maintenance. InfluxDB starts collecting the data and allocates the required storage for that new field.
- InfluxDB, given its built-in retention policies, provided control over which data to expire.
- To avoid using a second database, Unix Edge also chose InfluxDB to store metadata.
- For convenience, Unix Edge store both the text representation and integer representation of an integer value.

In its solution, Unix Edge follows a PPP model from Prototype, to Pilot, to Production. Prototypes are rapidly developed in-house for proof of concept. A set of pilot devices, also built in-house, validate the solution in the field over a fixed period of time. Unix Edge's manufacturing partners make the production devices for optimum reliability.

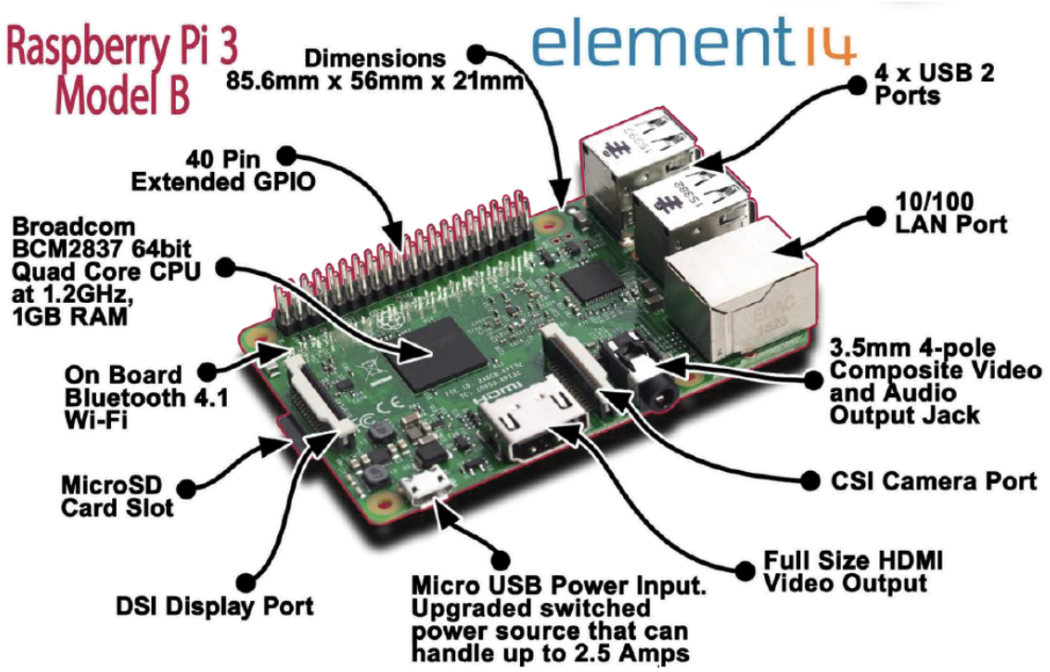


**Unix Edge Pilot Device**

In developing their devices, Unix Edge addressed each of their keys to reliability:

**Platform (Edge)**

After considering different single-board computers, Unix Edge chose the flexible and powerful Raspberry Pi 3, due to its widespread use (over 10 million in use and 100 hundred million users).




**Platform (Backend)**

In the backend, Unix Edge wanted to be on a cloud platform. They chose Linux because it is familiar, reliable, and mature with no license fee, and they found that the IIoT is making classic Linux and C programming skills once again highly relevant and applicable for device reliability. So after considering Amazon Web services (AWS) and MS Azure but finding them costly, they chose a company called Linode which runs Linux Ubuntu 16.04 LTS for their two nodes and does backups: one hosts the production machine, and one hosts their development machine and web server.

**Platform (Backend)**

In the backend, Unix Edge wanted to be on a cloud platform. They chose Linux because it is familiar, reliable, and mature with no license fee, and they found that the IIoT is making classic Linux and C programming skills once again highly relevant and applicable for device reliability. So after considering Amazon Web services (AWS) and MS Azure but finding them costly, they chose a company called Linode which runs Linux Ubuntu 16.04 LTS for their two nodes and does backups: one hosts the production machine, and one hosts their development machine and web server.

**Media**

For media inside the machines, because the Raspberry Pi uses microSD cards, Unix Edge selected high-speed name-brand cards (SanDisk Ultra 8GB-Class 10 and Lexar 633x). Since SD cards generally do not tolerate an unlimited number of reads and writes, they followed MicroSD Card best practices, such as preventing OS Swapping, using SD only for data that must survive a reboot, and using RAM disk for logs and temporary files.

**Self-Provisioning**

Unix Edge wanted their devices to “self provision”: Data flows from the device automatically. Metadata can be added later to indicate which truck or equipment type is emitting the data.

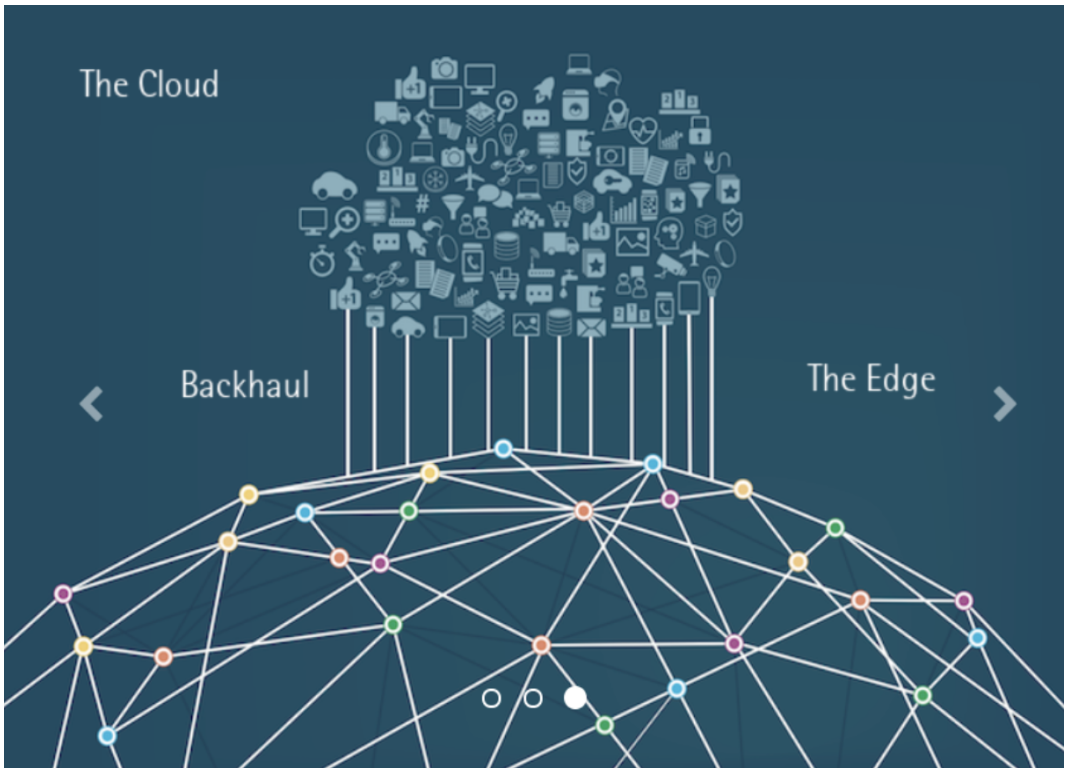
Each node uniquely identifies itself. Unix Edge initially used the MAC address of eth0, but to support Raspberry devices that don't have an ethernet adapter, they switched their solution to use the CPU ID, enabling it to run on nodes that have an ethernet 0 and those that don't. The CPU ID now serves as the single tag in the database (the unique identifier).

**Code Management and Collaboration**

Collaboration and code management (electrical engineer, installers, data scientists, vendors, and partners) were critical. With installations and team members in multiple locations, Unix Edge use Slack to communicate and share files such as equipment images, to show the model number and customer's identifier for a given machine and then configure that metadata to line it up with the hardware’s unique identifier.

**Redundancy**

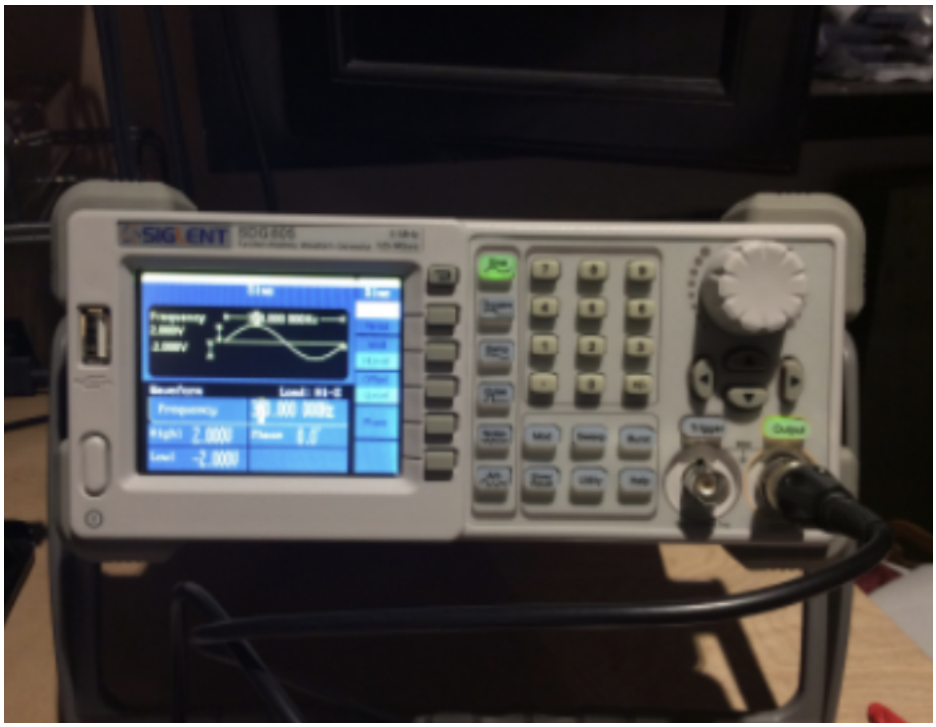
Since GPS signals are not consistently available, such as when a customer's equipment doesn't have a clear view of the sky, Unix Edge use GPS information from two different sources: one from the cellular towers and one from satellites in the sky. To determine which position from two different sources is more accurate, Unix Edge put a certain amount of logic on top of their data collection. To log into the devices whenever they're online, Unix Edge implemented remote access to them using a reverse tunnel, which eliminates the need to have a public IP on the devices. They can also send an SMS message to devices to prompt them to reboot. The devices update themselves using Git commands.




**Supporting Multiple Backhaul Options (WiFi, 2G, 3G, 4G, BTLE)**

### Testing

Through testing, Unix Edge resolved hardware problems such as faulty GPS antennas, bad power converters (the devices needed to run with various battery levels in trucks and compressors). The Unix Edge test bench includes signal generator, power supply, and oscilloscopes to test different scenarios and build a robust platform. They also maintain a handwritten physical log.




**Signal Generator to Simulate Signals from Field Equipment**

### Cybersecurity

Following hacking attempts on their system, Unix Edge implemented what they called Fail2ban, which temporarily bans an IP address from which multiple login attempts are made. They also avoid storing any Personally Identifiable Information (PII).

**Power**

Unix Edge designed their devices to anticipate power failures. They planned for a broad range of supply power voltages and use buck/boost voltage converters. Unix Edge track the voltage to detect failing batteries, prevent unavailability of equipment, identify undesirable operator behavior, and prevent erosion of confidence in the solution.

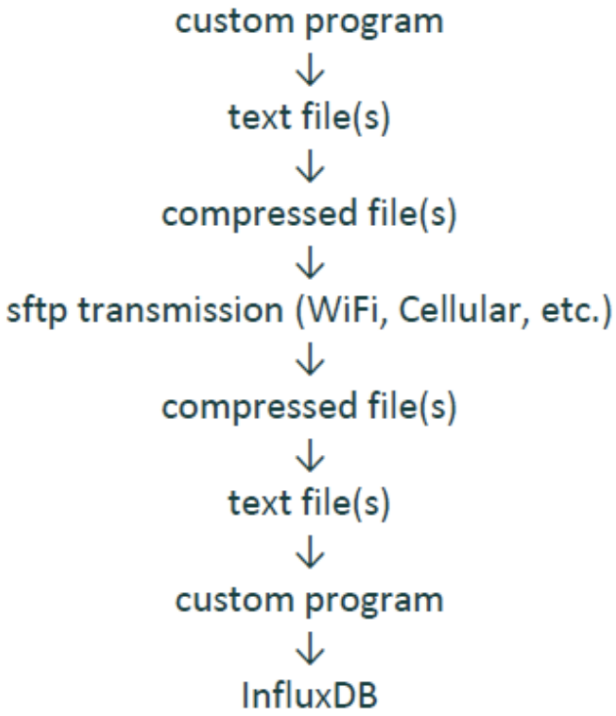
**Software**

Unix Edge opted to buy and use in-production software (commercial or open-source) to reduce the amount of custom code they would need and to have a more reliable solution. Apart from InfluxDB for storage, their software choices included Grafana for visualization. They write code only when required: for data analytics that deliver business value (such as whether their device is on, running, or idling and for how long); for user interface tweaks (customizing Grafana); and for generating Excel reports in customers' precise format, using the Python module XlsxWriter.

## Technical architecture

> *"The performance of InfluxDB has been tremendous. And that's what's required, right? Because if you're going to have a user-interface that's reading data directly from the database and the database is not responsive, then your user-interface will be unresponsive."*

**InfluxDB is the Only Database in Unix Edge Architecture**




**Here's how the data flows:**

- A Unix Edge custom program reads their sensors.
- Text files are used to store this data since text files persist through a data or telecom outage or a machine reboot.
- Unix Edge then generate and compress the text files. Encryption in the compression protects the data. They track how much data they are sending.
- They send the data over to sftp and then send it over the telecom layer (Wi-Fi, SigFox, or cellular).
- Once they get the data, the compressed file arrives then gets decompressed.
- Finally, text files get processed by a custom program written in Python which inserts the data into InfluxDB.

## What's next for Unix Edge?

Unix Edge are considering using Telegraf given its many input and output plugins, as well as using the MQTT protocol to support buffering or storing forward where the data would collect and persist.

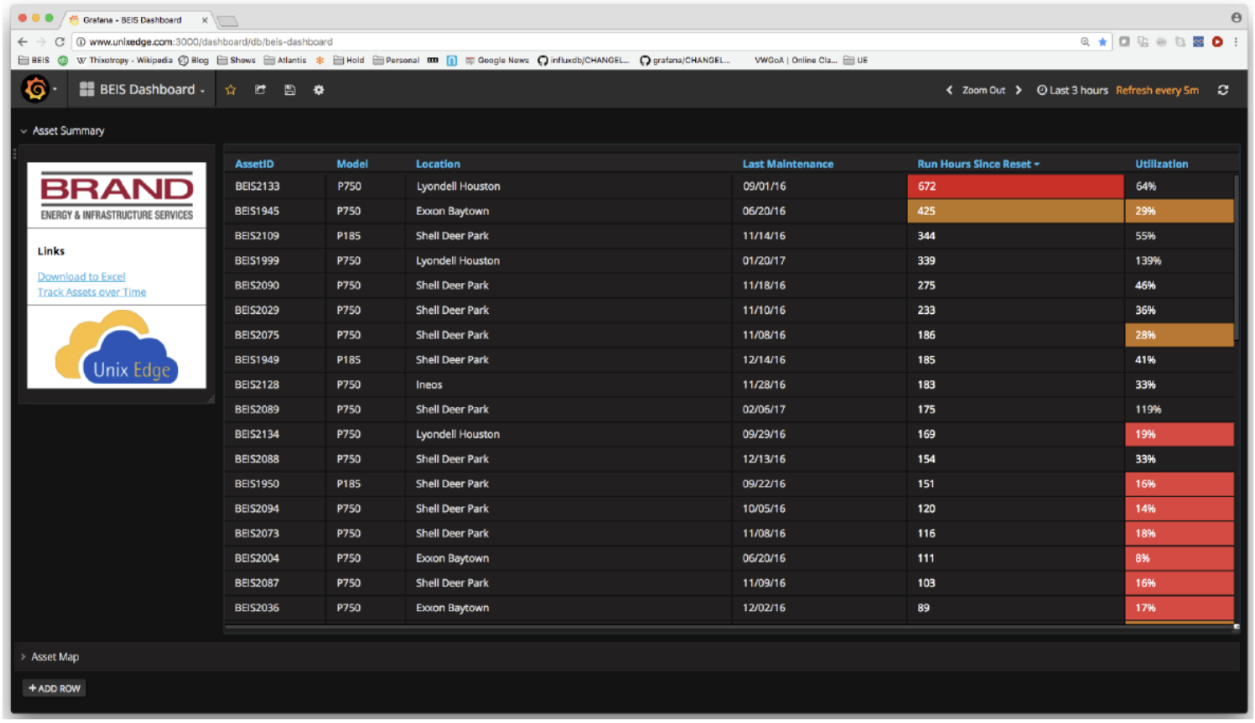
## Results

> *What I like to do is to take the technology and solve a real problem. It's quite satisfying to know that your system is providing value, saving the customer money, saving the environment, protecting people, and keeping them safe."*

Unix Edge managed cloud solutions require zero in-house IT resources so their customers can focus on their core business. Unix Edge's business model is a monthly subscription fee per truck, per asset, per compressor, per machine. Unix Edge shows customers how they are achieving cost savings for them by sparing them unnecessary maintenance, reducing fuel consumption, enabling more efficiency, and improving safety.

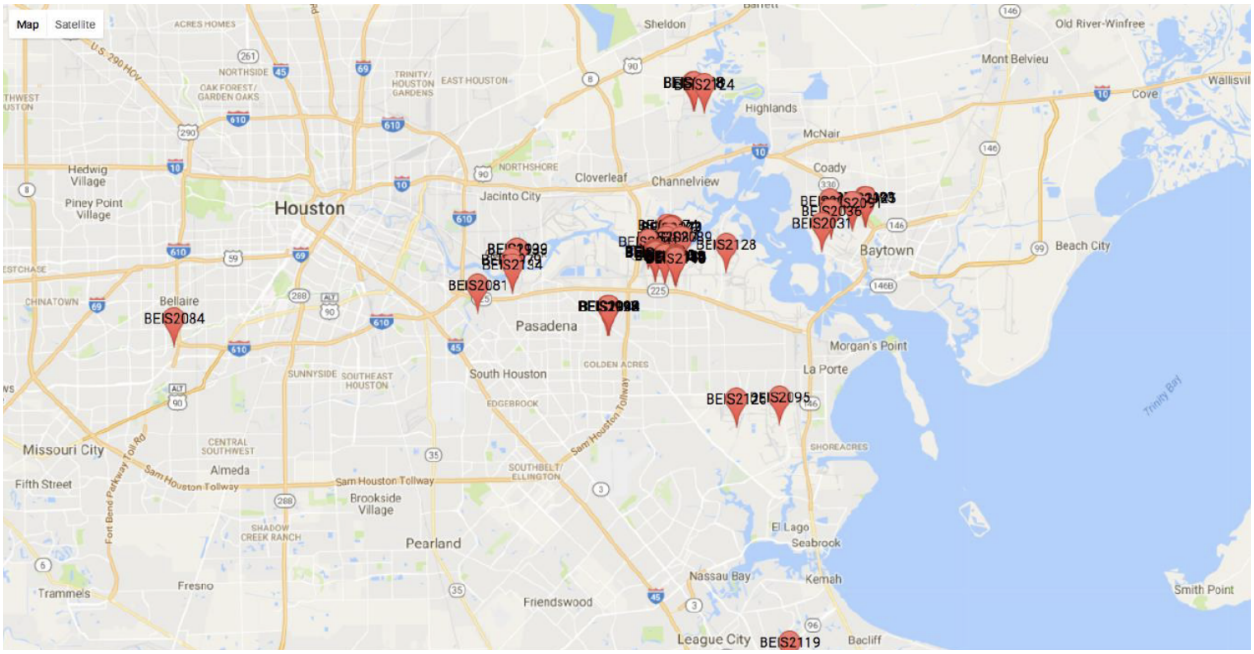
The Unix Edge main dashboard lists their machines, their location, and model number, and color- codes maintenance status and aggressive usage—based on data directly from InfluxDB. Run Hours Since Reset is done by an integration of data under a curve (accumulator). If the machine is on, they keep accumulating the time that it is on, by using interpolated values from InfluxDB, so that they can reach the granularity required for maximum accuracy.

## Main Dashboard - No Code




For monitoring arrival or departure of a piece of equipment to a maintenance facility, or if that equipment gets assigned to a particular customer, Unix Edge store that data in InfluxDB and track equipment via GPS using the Google Maps API. Using InfluxDB, they keep track of the metadata history and the most recent time-stamped values. Sensors, switches, and levers show what activity the truck is performing and enable precision billing based on activities performed.

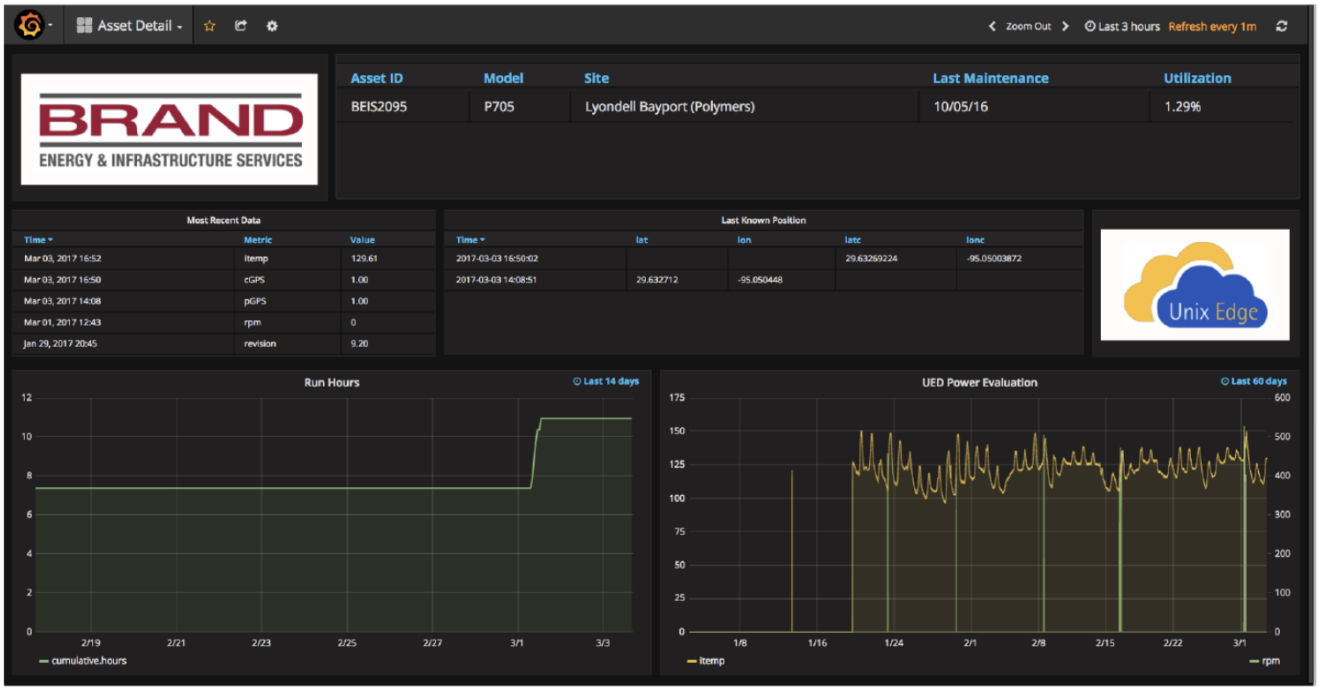
## Tracking Customer Equipment Based on GPS Location




For vacuum trucks, seven types of data—GPS coordinates, ambient temperature, bed height, lever position, switch position, engine RPMs, ignition state—get collected into Unix Edge Devices, then get compressed, password-protected, and sent to the cloud.

With regard to bed height in particular, data spikes—spurious values from the ultrasonic sensor that measures truck bed heights—can occur when trucks are bouncing around. To discard unreasonable sensor values, Unix Edge is in the process of using Kapacitor to implement, in production, the Tukey Outlier algorithm (based on mathematician John Tukey's work).

## Asset Detail - No Code




## Vacuum Truck Dashboard

For compressors, Unix Edge monitor RPMs, battery voltage, and GPS position—such metrics help customers understand component efficiency and comply with safety regulations. For example, compressors' diesel engines (old simple engines that don't have onboard diagnostics) are monitored with a sensor placed on the engine alternator, using an oscilloscope. Fluctuating RPMs could indicate an engine that's not running well, a slipping belt, or an engine left to run overnight, which can result in diesel fuel waste and safety violations.

Unix Edge also implemented a safety blinking light for compressors to advise a delay in starting the engine after the compressor has been lifted by cranes, which results in the tank's diesel fuel to get shaken about and have particulates in it. That way, Unix Edge saves customers fuel filter changes and delivers savings.

Using InfluxDB, Unix Edge is fulfilling its mission of building managed solutions that combine cloud platforms and Edge Devices to deliver tangible business value.

## About InfluxData

InfluxData is the creator of InfluxDB, the leading time series platform. We empower developers and organizations, such as Cisco, IBM, Lego, Siemens, and Tesla, to build transformative IoT, analytics and monitoring applications. Our technology is purpose-built to handle the massive volumes of time-stamped data produced by sensors, applications and computer infrastructure. Easy to start and scale, InfluxDB gives developers time to focus on the features and functionalities that give their apps a competitive edge. InfluxData is headquartered in San Francisco, with a workforce distributed throughout the U.S. and across Europe. For more information, visit influxdata.com and follow us @InfluxDB.