Signpost: A Roadmap for City-Scale Sensing

Branden Ghena
Bradford Campbell
Pat Pannuto
{brghena,bradjc,ppannuto}@umich.edu
University of Michigan
Ann Arbor, Michigan

Joshua Adkins
Neal Jackson
Prabal Dutta
{adkins,neal.jackson,prabal}@berkeley.edu
University of California, Berkeley
Berkeley, California

ABSTRACT

City-scale sensing holds the promise of enabling deeper insight into how our urban environments function. Applications such as observing air quality, monitoring traffic flows, and measuring sources of noise pollution can have powerful impacts, allowing city planners and citizen scientists alike to understand and improve their world. However, the path from conceiving applications to implementing them is fraught with many challenges. A successful city-scale deployment requires physical installation, power management, and communications—all challenging tasks standing between a good idea and a realized one, suggesting the need for a platform that enables easy deployment and experimentation of city-scale sensing applications.

To address these basic challenges, we present Signpost, a modular platform for city-scale sensing. Signpost simplifies deployment and installation in cities by removing the need for connection to wired infrastructure and instead harvesting energy from an integrated solar panel. The platform provides the key resources necessary for its pluggable sensor modules to support city-scale applications. Signpost stores excess energy for later use, distributes energy between modules, and provides communication through multiple wireless protocols. It also offers storage for sensor data and allows for local processing in a duty-cycled Linux environment. We explore the goals and design tradeoffs inherent to this focus on modularity and deployability, describe the capabilities of the platform, and present initial results from a prototype Signpost deployment. We believe that Signpost can reduce the difficulty inherent to city-scale deployments, enabling new deployments, insights into urban health, and ultimately improved cities.

TECHCON 2017, Austin, Texas

Figure 1: A deployed Signpost. The Signpost platform mounts to existing street signposts, harvests from a 0.1 m^2 solar panel on the back side, and provides its tenant sensor modules with power, networking, storage, time, location, and high-performance compute.

1 INTRODUCTION

There is growing interest in making cities safer, cleaner, healthier, more sustainable, more responsive, and more efficient—in a word, smarter. And for good reason: applications such as pedestrian route planning based on air quality [3], noise pollution monitoring [5], and gunshot localization [4] can improve the quality of life for the city's inhabitants.

However, enabling city-scale applications such as these requires new sensing, computation, and communication technology to be embedded within cities. These deployments suffer from a combination of costly installation overhead (to get approval or AC mains power access), technical complexity (to implement and debug energy management and networking), and cost (to create a fully-functional, weatherproof device), making it difficult to deploy applications in cities, and particularly challenging to do short-term exploratory research. We argue that a platform that abstracts away the commonly used services required by many smart city sensing applications, does not require hard-wired access to power, and supports isolated, modular, and upgradeable sensing applications will lower the barrier to collecting city-scale data, performing short-term experiments, and creating applications that make cities better.

To evaluate these claims, we design and implement the Signpost platform: a modular, solar energy-harvesting, and signpost-mountable system for enabling easy and ubiquitous installation of smart city applications. By eschewing the need for mains power and wired networking in favor of a built-in solar panel and suite of wireless radios, Signpost can be much more flexibly installed in actual cities. While other projects, such as the Array of Things [1] and SONYC [5], focus on computational ability and specific sensor modalities, we focus on ease-of-deployment and modularity. Our system includes many design decisions aimed at deployability: it mounts to the ubiquitous street signpost pole (Figure 1), uses a vertically-mounted solar panel facing an arbitrary direction, dynamically adapts to available energy, and supports a range of wireless radios for various connectivity options. We also prioritize modularity by including eight plug-in module slots for sensors, processors, and radios which can upgraded over time as application needs and technology options evolve. These modules have access to system-provided services including power, networking, storage, time, location and high-performance compute through API calls to the system.

Signpost stakes two claims: that ease-of-deployment will drive adoption of city-scale sensing and that a modular approach will drive its success. This paper considers tradeoffs of realizing those features and presents the design and implementation of a platform to support them.

2 PROVIDING RESOURCES

In order to reduce the burden for module creators, Signpost provides resources to modules attached to the platform. Figure 2 shows an overview of the system design.

Energy. Energy storage and collection are key to sensor deployment, and an obvious choice to provide on Signpost. A connection to AC mains would provide a relatively unlimited supply of energy for modules, enabling high-power sensing and computation. However, it would also limit possible deployment locations and require costly and time-consuming installation by city utility workers. Running off of stored energy in batteries is sufficient for many short-term sensor network deployments, but battery replacement is not scalable for large, geographically distributed deployments. Instead, harvesting energy from solar panels creates an acceptable middle ground. A solar panel the size of Signpost ($0.1\ m^2$) can be expected to produce several Watts of power, averaged over an entire day. A battery is still necessary for such a system in order to provide stability over nights and cloudy days.

Networking. While it is possible to run some applications entirely locally on the Signpost, many will want to upload data to the cloud for further use. Updating applications on the modules also requires a network connection. Rather than having each module solve networking problems on its own, Signpost can provide communications for all modules. However, no one communication method is suitable for all needs. Providing a cellular connection may be ideal for short-term reliable, high-throughput connections, but is too costly in energy for constant use. Long-range, license-free solutions such as LoRaWAN provide lower-power communications with a range of several kilometers, but correspondingly low throughput as well, on the order of several kbps. Finally, local short-range connections such as Bluetooth Low Energy are useful for direct interaction with nearby individuals, allowing for public interaction with deployments. Signpost provides all of these options, allowing the choice to be made at runtime based on available resources and application requirements.

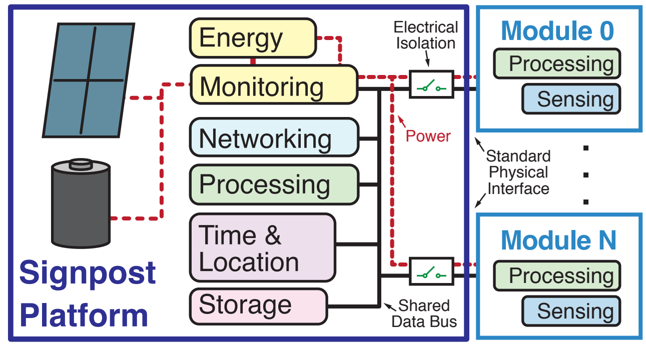

Figure 2: Signpost platform design. Signpost monitors and distributes energy to connected modules and provides common services like networking, processing, and storage, allowing modules to focus on their core sensing application.

Processing. In nearly any sensing system, data must be processed, batched, transformed, and analyzed. While much of this processing could be done on a microcontroller, providing a Linux environment allows module creators to use the languages and libraries they are accustomed to. Unfortunately, existing Linux compute modules are all too high power for continuous operation on an energy-harvesting system like Signpost, but treating a Linux computer as a heavily duty-cycled coprocessor is sufficient for many processing needs. Data can be batched by the sensor module and then periodically processed.

Other Resources. Centralized storage allows module data to be batched for processing or transmission without requiring each module to implement a storage method. Time and location can both be provided with a single centralized GPS unit. They are useful not just as sensor metadata, but also for synchronization and localization applications. Finally, Signpost also provides an installation method for modules, utilizing off-the-shelf waterproof enclosures that can be connected to the platform.

Service	System Call	Description
Init	`i2c_address = module_init(**api_handles)`	Initialize module
Network	`response = network_post(url, request)`	HTTP POST data to URL
	`network_advertise(buf, len)`	Advertise data over BLE
	`network_send_bytes(destination, buf, len)`	Send via best available medium
Storage	`record = storage_write(buf, len)`	Store data
Energy	`energy_info = energy_query( )`	Request module energy use
	`energy_set_warning(threshold, callback)`	Receive energy usage warning
	`energy_set_duty_cycle(duty_cycle)`	Request duty cycling of module
Processing	`processing_call_rpc(path, buf, len, callback)`	Run code on Linux compute
Messaging	`messaging_subscribe(callback)`	Receive message from a module
	`messaging_send(module_id, buf, len)`	Send message to another module
Time	`time_info = get_time( )`	Request current time and date
	`time_info = get_time_of_next_pps( )`	Request time at next PPS edge
Location	`location_info = get_location( )`	Request location

Table 1: Signpost API examples. Abstract versions of several Signpost API calls for each system service are shown. Providing a high-level API to applications enables easier development for domain experts.

3 SUPPORTING MODULARITY

Signpost supports sensor modules, which connect through standard electrical and mechanical interfaces, have access to system resources, and perform the actual city-scale applications.

Each module is independently powered by the system. To enable various energy apportionment policies, Signpost is able to monitor the energy usage of each module individually, disable modules using more than their allocated share, and track the energy available to the system as a whole. Shared resources on the system, such as the Linux coprocessor and radios, are also monitored so that their energy use can be attributed to the application using them.

In order to use system resources, each module is connected to a shared data bus, implemented as a multi-master I^2C bus, over which it can send commands and requests. Dedicated USB connections from each module to the Linux coprocessor are also included in the design, allowing high-speed communication if so desired for data processing. The platform is capable of isolating modules from the buses if necessary.

To aid in creating modules for the Signpost platform, a software library is provided that abstracts away discovering system services, implements the over-the-wire protocol, and provides a simple API for using the provided resources as shown in Table 1. The current implementation is written for the Tock operating system [2], but the library is written on top of a hardware abstraction layer, allowing it to be easily ported to other platforms such as mbed or Arduino.

4 THE REALIZED PLATFORM

The Signpost platform, with hardware shown in Figure 3, is defined by the Power Module, Control Module, Backplane, and Radio Module. A full Signpost has six general module slots, one of which is taken by the Radio Module, leaving five for sensing capabilities. The size of the entire system, including a case, is 0.429 m by 0.300 m.

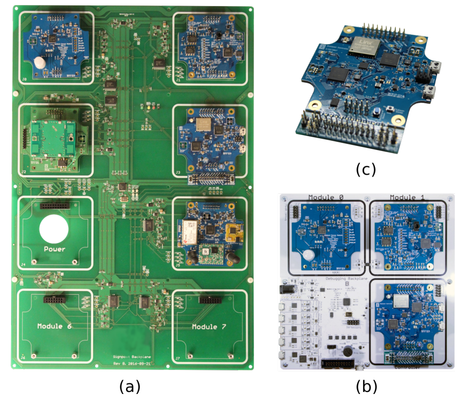

Figure 3: A full Backplane (a), Development Backplane (b) and Control Module (c). The Backplane serves as the Signpost interconnect, while the smaller Development Backplane is the desktop equivalent, enabling easy module and application creation and testing. The Control Module manages Signpost energy and provides services to sensor modules.

The Backplane has physical and electrical connections for modules, signal routing between modules, and isolation hardware. There are eight slots where modules can be connected. Two are special-purpose for the system, and the remaining six are standard interfaces for modules. All connections to modules are individually isolated and buffered by the Backplane in order to prevent electrical issues on one module from affecting the entire system. A smaller development version of the Backplane has also been created for use in creating and debugging modules.

The Power Module is responsible for energy management, monitoring, and distribution. Energy is harvested from the attached 0.39 m by 0.27 m solar panel and stored in a 9 Ah lithium polymer battery back. Power rails for each module provide 5 V at up to 1.5 A of draw, monitored by fuel gauges with a precision of 244 nAh. In addition, the Power Module provides watchdog hardware capable of resetting the entire system in the event of software failures.

The Control Module handles system tasks, such as managing the energy policy, assigning module addresses, and monitoring for system faults. It also provides time, location, storage, and processing services to the modules. Processing is provided by an attached Intel Edison Linux computer.

The Radio Module provides networking services to Signpost through Cellular, LoRaWAN, and BLE radios. The energy use of each radio is individually monitored, allowing communications costs to be charged to application modules. Providing three communications interfaces allows Signpost to make decisions about which radio to use based on quality of service, latency, throughput, and energy requirements.

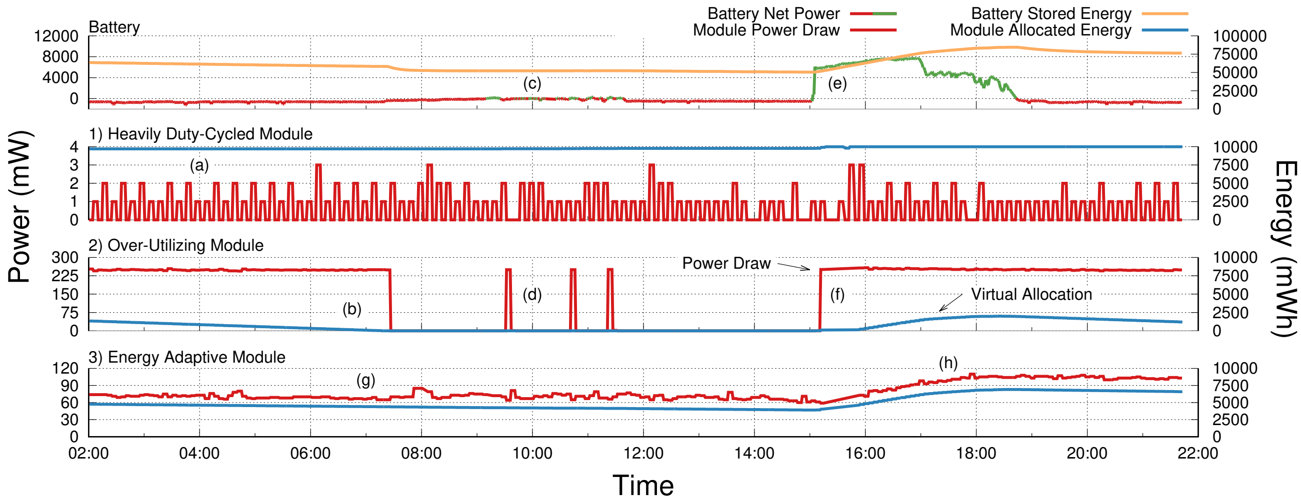

Figure 4: Effect of energy distribution policy on module operation and system energy. Shown is the operation of three modules operating for a day under a simple energy distribution policy. Each module employs a different management technique for its allocated energy. Module 1 asks the platform to duty cycle it and is turned on once every 10 minutes, evident in the duty cycling events (a); in this way it is able to achieve extremely low average power with little developer effort. Module 2 consistently draws a high amount of power, causing the platform to cut it off when it has used all of its share (b). When the battery is charging slowly (c), it is enabled several times, only to quickly use its entire allocation and get disabled (d). Eventually the Signpost is in more direct sunlight (e) and enough energy is distributed to the module for it to remain enabled (f). Module 3 adapts the amount of energy it uses to the amount of energy it has remaining in its allocation. This can be observed in the lowering average power through the night (g) and the increasing average power as the battery charges (h). These situations show that Signpost has the facilities to handle energy distribution and management in a multi user system, and that the APIs it provides can be successfully used by modules to handle the varying available energy inherent to energy harvesting systems.

4.1 Evaluation

We evaluate Signpost, deploying it outdoors with three attached sensor modules for a 20-hour period. We demonstrate an example energy policy, wherein each module is granted a virtual battery, with incoming energy equally distributed between virtual batteries. Modules can use a software API to query for their available energy and request to be duty cycled. Figure 4 shows the affect of the energy distribution policy on three modules. Each module uses a differing amount of intelligence in deciding when to perform sensing activities, resulting in differing amounts of runtime.

5 CONCLUSIONS

In this paper, we describe the goals and design of Signpost, a solar energy-harvesting modular platform designed to enable city-scale deployments. By providing energy, networking, storage, processing, time, and location services, Signpost aims to allow developers to focus on the sensing application they actually care about rather than the engineering details of making it deployable, allowing experimentation with new applications and modalities, new frameworks for distributed city-scale applications, and new insights into the functionality of our urban world.

6 ACKNOWLEDGMENTS

This work was supported in part by the TerraSwarm Research Center, one of six centers supported by the STARnet phase of the Focus Center Research Program (FCRP), a Semiconductor Research Corporation program sponsored by MARCO and DARPA. This material is based upon work supported by the National Science Foundation Graduate Research Fellowship Program under grant number DGE-1256260. This work partially supported by generous gifts from Intel and Texas Instruments.

REFERENCES

[1] 2016. Array of Things. (2016). https://arrayofthings.github.io/ Accessed 2016-04-10.
[2] 2017. Tock Operating System. https://www.tockos.org/. (2017).
[3] Yun Cheng, Xiucheng Li, Zhijun Li, Shouxu Jiang, Yilong Li, Ji Jia, and Xiaofan Jiang. 2014. AirCloud: a cloud-based air-quality monitoring system for everyone. In *Proceedings of the 12th ACM Conference on Embedded Network Sensor Systems*. ACM, 251–265.
[4] A. Ledeczi, P. Volgyesi, M. Maroti, G. Simon, G. Balogh, A. Nadas, B. Kusy, S. Dora, and G. Pap. 2005. Multiple simultaneous acoustic source localization in urban terrain. In *IPSN 2005. Fourth International Symposium on Information Processing in Sensor Networks, 2005.* 491–496. DOI: https://doi.org/10.1109/IPSN.2005.1440982
[5] Charlie Mydlarz, Justin Salamon, and Juan Pablo Bello. 2017. The implementation of low-cost urban acoustic monitoring devices. *Applied Acoustics* 117 (2017), 207–218.