Live, Continuous Energy Auditing of Miscellaneous Electrical Loads: Methodology and Results

Jorge Ortiz, Jason Trager, Gavin Saldanha, David E. Culler, Paul K. Wright

UC Berkeley

ABSTRACT

We present the architecture, implementation, and results from a live, continuous energy auditing system. By combining QR codes and mobile phones, we are able to iteratively collect building plug-load information and couple it with live, streaming meter data. Our methodology includes the deployment of inexpensive tags (QR codes) as well a smartphone-based auditing application. The mobile phone reads a tag to associate an electrical plug load with meter data and deployment information. We deployed our system in a 141,000 square foot, seven-story building, and tagged over 351 items over 139 rooms. We present the architecture of our system and the results of our initial audit. We also describe our application and future directions of this work.

**Introduction**

Miscellaneous electrical loads (MELs) consume about a third of the energy consumed in buildings in the United States [doe]. According to Lanzisera et al. [LBNL\_MELS], MELs data are typically collected from national surveys of several thousand homes and commercial buildings. The data is statistically disaggregated to estimate energy use and the residual from these models is classified as “other” and referred to as miscellaneous. It consists of end-use electrical loads that are not accounted for in the traditional heating, cooling, ventilation, and lighting. With the recent rise in MELs, there has been more interest in characterizing the energy use of the equipment in this category. Porter et al. [Porter2006] sampled 50 homes, metering 17 devices per home over the period of a week. Moorefield at al. [Moorefield2008] sampled 47 office buildings, metering 10 devices per building for a period of two weeks. In both studies, data was collected at one-minute intervals and analyzed offline.

Although these initial studies provided more insight than ever before, the per-building deployment scale was limited, largely due to cost and technological limitation of the tools available at the time the studies were done. The last study used a plug-load meter called a ‘Watts-up’ with no networking capabilities, which costs in the range of $200-300. This makes sampling every device in the building cost-prohibitive. Also, these meters are not usually networked and contain a limited amount of local memory, so the sampling and collection frequency must be carefully scheduled and managed. You need to sample frequently and long enough to capture the underlying trends but not so quickly that you have to constantly *walk* over and collect the data from each meter. Lanzisera et al. [LBNL\_MELS] used a much cheaper power meter with low-power wireless networking capabilities called ACme [acme]. Although not widely available on the market, they are significantly cheaper than the ‘Watts-up’, costing in the range of $20 in volumes of 10,000. Not only does the cost allow for much large deployments but their wireless networking capabilities remove the need to physically “milk” the meters when space in local memory is exhausted.

Another aspect of deployment setup that makes scaling difficult is the amount of setup work that needs to be done beforehand. Before collecting any meter data, deployment information must be collected and organized; in order to set the context for interpreting them. For example, the analyst must note that power stream X came from meter Y which was attached to device Z of building A, floor B, room C. During this phase, one of the most time-consuming tasks is collecting a list of the each device and categorizing it. This process involves a walk through the building listing each plug-load item, classifying it, and recording some item-specific descriptive information. Typically, this information is written on paper and transferred manually into a database for analysis. The associated database schema should be designed to match the information about the deployment, devices, and meters. It must also capture information about the association between the meter, the item, and its location; allowing the analyst to submit cross-cutting queries for each view.

At large scales, this process also becomes very time consuming and increases with the size of the building and the number of devices that are being metered, since inventory-recording is typically completed by a member of the investigation team; which can only grow so large given a limited budget. The study done by Lanzisera et al. [LBNL\_MELS] consisted of a data-collection team of three members, and they deployed over 455 sensors throughout an entire building. However, the information setup and gathering phase took a several weeks and is discussed in great detail in [cmels].

We argue that information technology should be used to scale this process and that the deployment should be used to extract more value for building occupants. The infrastructure, in the aforementioned studies, was only used for the associated study. It provides *no direct value to building occupants*. We believe that ***energy data services***should be offered to building occupants, provided over the deployed infrastructure. For example, occupants should be allowed to directly observe the energy consumption of their devices or see how much they have consumed over a given time period. We also believe that occupants can help track the evolution of the infrastructure (i.e. meters added, removed, replaced) through these services and they should be indirectly tasked with *distributed consistency maintenance of deployment state* as captured in the accounting system, in return for real-time energy data services.

In summary, our work attempts to:

1. Use cheaper, networked sensors and expand the set of acceptable data streams as they become available.
2. Decrease the time it takes to gather and collect MELs information by using cheap/scalable information technology, such as mobile phones, QR codes, and cloud services.
3. Provide energy data services to building occupants.
4. Involve occupants in maintaining a consistent view of the deployment as it evolves.

In the rest of the paper, we describe our architecture and explain our design choices for achieving the aforementioned goals. We also describe the energy services we have built on top of our deployment infrastructure, go through the initial energy audit results we obtained through our system, and discuss plans for future work.

**System design**

Although sufficient for their specific study, we believe that we can further leverage IT to design a system the has the following properties:

* Cost effectiveness
* Scale
* Generality and Ease of Use
* Incrementally deployable

Our architecture consists of four main components: the sensing tier, the data-management tier, the application tier, and the tagging infrastructure. Figure 1 shows the components of our architecture. The sensing tier is shown the ‘plug-loads’ box. Notice the small acmes with QR codes attached to them. These acmes are placed on plug-loads throughout the building. The data management tier is indicated by the box on the upper left hand corner of the figure. The cloud element represents the fact that our data tier resides in a datacenter as a cloud service that is accessible over the network using the mobile phone. The application tier lives on the mobile phone and any computing client with network access (not shown in the figure). Finally, the tagging infrastructure is indicated by the QR codes spread throughout the physical environment. QR codes bridge the physical world with the data management tier and the application.

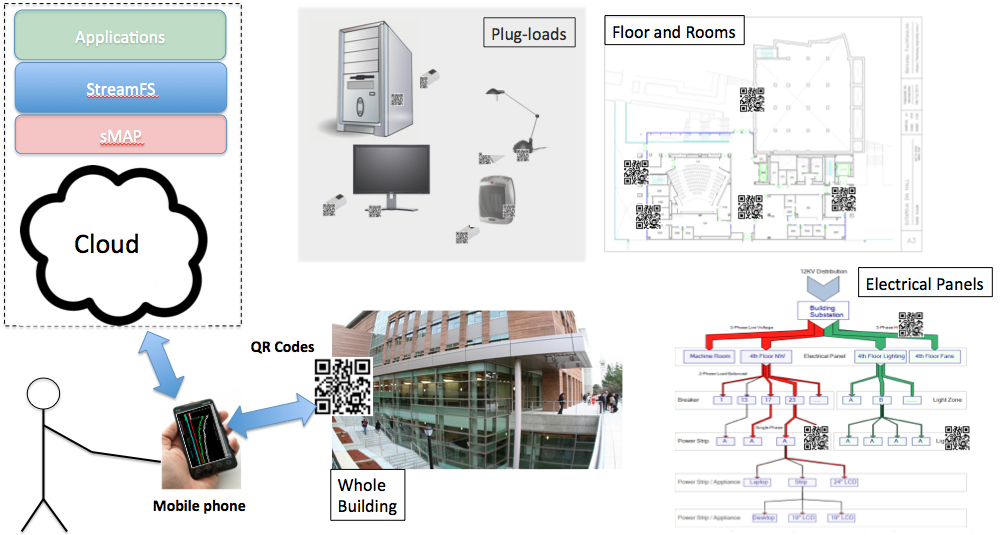


Figure 1. This figure presents the Berkeley mobile auditing system (BMAS) architecture. The figure shows the sensing tier, the data management tier, the application tier, tagging infrastructure that bridge the other three through the application on the mobile phone.

**Cost effectiveness**

To achieve cost effectiveness we examined way of cutting the cost of each component of the architecture. The plug-load meter we chose are the same ones used by Lansizera et al. [LBNL\_MELS], where the large bulk cost was in the range of $20 per device. The price per meter was slightly higher since the bulk number of manufactured meters was much lower than 10,000. However, the cost per meter was still in the range of $50. The data tier was set up in Amazon EC2 as a large unix-based instance, which costs $0.36/hour. The application tier is free. We provide a code base that can be downloaded through your android phone. The tagging tier is also virtually free. Tags can be printed on demand from QR-Code generation sites. We also provide a site that automatically generates tags.

**Scale**

Cost plays a large role in scalability. Although not a technically fundamental limitation, it prohibits deployments from getting too large. The decision to use Acmes allows us to deploy many sensors and stratify a much large set of plug-loads. By choosing to run the data management tier in the cloud we are able to scale our data tier up or down. The cost scales with the number of the machines we spawn to manage the underlying data and support applications. We ran a service called StreamFS [sfs] as the main data management service upon which we built our android application.

Choosing the mobile phone as the application delivery and system interaction interface allows you to scale up the number of simultaneous inputs to the system for collecting deployment information. It removes the information-to-paper step and lets you directly input deployment information with your phone. It also allows you to involve building occupants by having them participate in the input process with their personal devices and serves as a good platform for personalized energy services. Finally, QR codes are easy to print and can be generated and printed by building occupants themselves. Since they are cheap they are easily replaceable and they can be placed throughput the building.

**Generality and ease of use**

Generality can be achieved with the right abstractions and interfaces. For this application we chose to use sMAP [smap] and StreamFS [sfs] in the data management tier, QR codes in the tagging tier, and the mobile phone in the application tier. sMAP provides a generalized, uniform interface for accessing physical data from sensors. StreamFS provides a generalized interface for organizing sensor data and coupling the organizational structure with streaming data directly. It also provide aggregation facilities that were useful for building from of the initial energy services. Both provide HTTP/RESTful APIs, simplifying the integration processes between both layers and with the application. Figure 2 shows the StreamFS architecture.

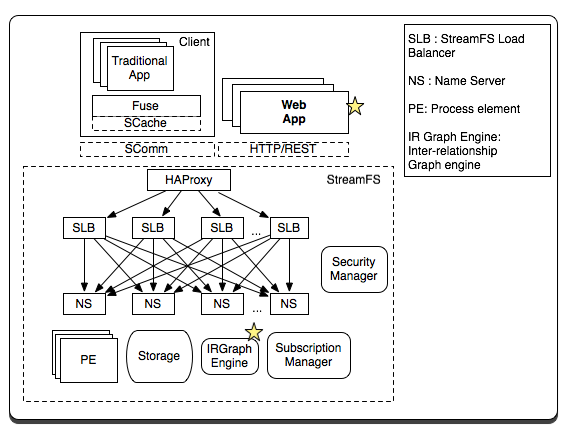


Figure 2. This figure shows the components of the StreamFS system. The main component of the system that interacted with our application was through the HTTP/REST interface. Our application also made use of the inter-relationship graph component as a tool for building energy-management services. Both are starred in the diagram.

The mobile phone served the main platform for facilitating interaction between users and the infrastructure. By using the on-board camera as a generic scanning device we are able to bridge the physical world with information about the physical world as represented in the data management tier.

**Incrementally deployable**

Finally, we aim for the architecture to allow incremental deployment so that it can grow over time to capture more than just plug-load information. sMAP generalizes the notion of sensor data into the umbrella that encompasses all physical data. StreamFS couples this notion with semantic information by allowing users to name their streams according to categorical, systematic, spatial or other semantic relationships and we make direct use of that in our mobile application. StreamFS allows updates of the underlying representation and organization of the data and both follow the principal of horizontal scalability – if you need more compute or storage, simply add more servers incrementally on EC2 or your favorite cloud-service provider. Storage grows with the amount of data collected, compute power grows with the number and complexity of running applications.

QR codes are also incrementally deployable. The more items you have the more QR codes you print and register with the underlying system. Furthermore, as smart phones become more and more popular, you will have more occupants interacting with the infrastructure through the application. All of the pieces are meant to build up as demand increases on multiple fronts: more data, more compute, more users, more applications.

Finally, the input protocol implemented by the mobile application incrementally adds components to the infrastructure in small units that include the meter, the item, and the location of the meter/item combination. We will go through each step of the protocol in the upcoming sections.

**Berkeley mobile auditing system (BMAS)**

We call our system the Berkeley mobile auditing system (BMAS). We implemented a version of the application using the aforementioned components. The main pieces consist of an android-based application, acme sensors, QR codes, and sMAP and StreamFS running in the cloud.

**Metering devices**

In our deployment we used Acme power meters [acme] to take measurement of devices. Acmes transmit data periodically to a local IPv6 router. We set up a network of Acmes in various parts of the building, had them report their data at one-minute intervals to an sMAP server in the cloud. Our architecture does not restrict you to using only Acme meters although they provide many benefits based on price-point and manageability.

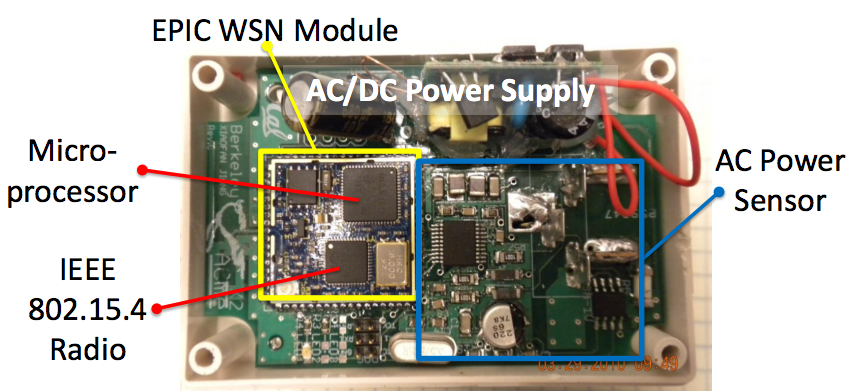


Figure 3. Acme power meter. The hardware includes a microprocessor running an entire IPv6 stack, an AC power meter, and a low-power wireless radio.

**Cloud-storage tier**

As mentioned, our cloud storage tier consists of sMAP machines and StreamFS machines. sMAP provides uniform representation and access for retrieving physical data and StreamFS organizes that information into a common namespace, capturing the inter-relationships that are interpreted by the mobile application. StreamFS manages metadata like a filesystem; organizing the data into a hierarchical namespace with file and folders. It also supports the Unix “pipe” facility and symbolic linking. The main differences are that each of the files and folders are taggable with queryable metadata, streams can be subscribed to so the application can attain the data in real time, the data is automatically aggregated, and historical queries can be posed on the stream data.

These features are main benefit of using StreamFS. When constructing the namespace for a building we wish to refer to the spatial hierarchy and the sensors within that hierarchy. For example, the root of the namespace is the building, followed by each of the floors and rooms within them. Inside each room or space we included a folder for each metering device. The contents of the folder are the collection of streams produced by the metering device. The main purpose of this is to capture the deployment context so that we can interpret the data in the appropriate context. We transform the physical deployment to a structured, traversable namespace as shown in Figure 4.

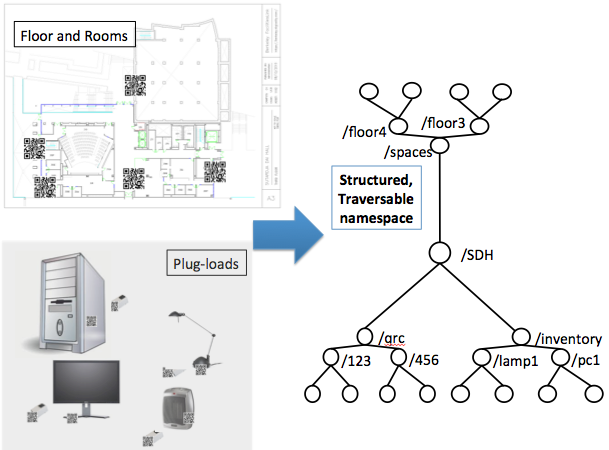


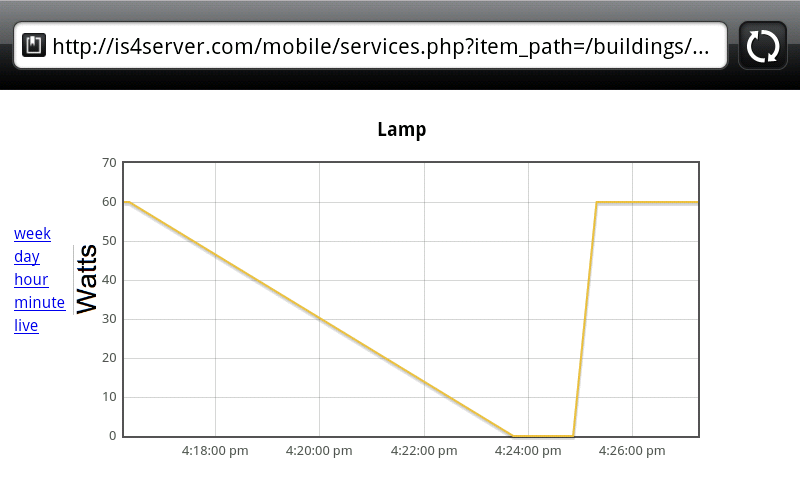
Figure 4. This figure shows how StreamFS is used to manage the deployment in a common namespace. We take items from the building, including the QR codes, and build up the namespace and inter-relationships using the mobile application.

To access the acme device in room 410 on the fourth floor of Soda Hall we fetch the data from /soda/4/410/acme123. This naming structure suggests that the name encodes the location, but we may also wish to access the name information according to the panel it is attached to or who the owner of the device is. For this access pattern we simple create a new hierarchy and use the symbolic linking mechanism.

StreamFS supports symbolic linking, similar to a Unix filesystem. This allows us to capture the entity-relationships by symbolically linking nodes from multiple sub-namespaces. In the example above, we can create a new tree rooted at /soda called /soda/electrical and access the meter from /soda/electrical/rootpanel/panel5/load83/acme123, where this is a symbolic link to /soda/4/410/acme123. Our application used three namespaces: /dev, /qrc, /spaces. The /dev namespace was used to group all device folder and their associated streams, the /qrc folder held all QR code ids (encoded in the QR codes we generated), and the /spaces namespace grouped the floors, room, and spaces throughout the building. Symbolic paths were resolved as a lookup mechanism. QR code file were linked to the device and the location folder linked to the device. Devices linked to meters. To fetch device info, the QR code device link is resolved, to get associated data, the device’s meter link is resolved.

**Mobile application**

The mobile phone application is mainly used to input data about the deployment but can also be used to fetch and display. The primary operations supported are 1) registration, 2) bind/attach, and 3) scan. Figure 5 shows a few screen shots of the menu in the mobile application.



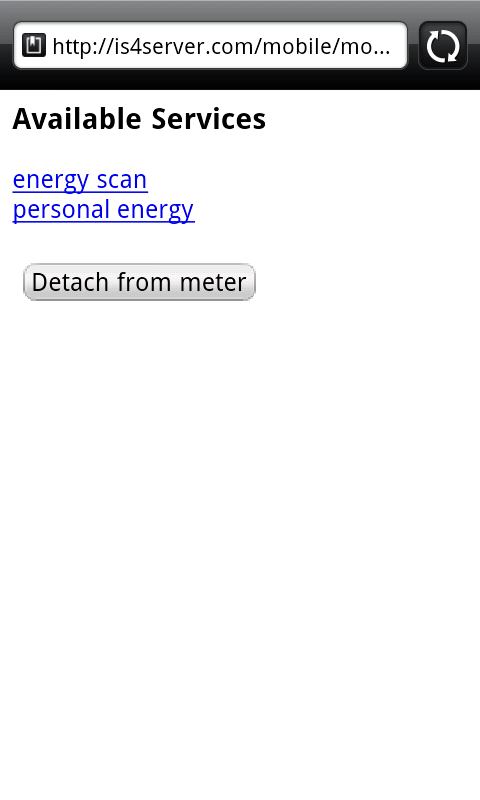
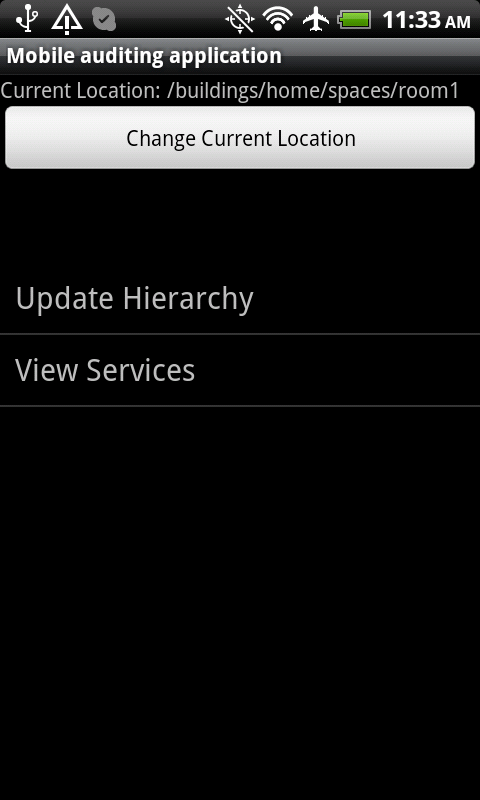
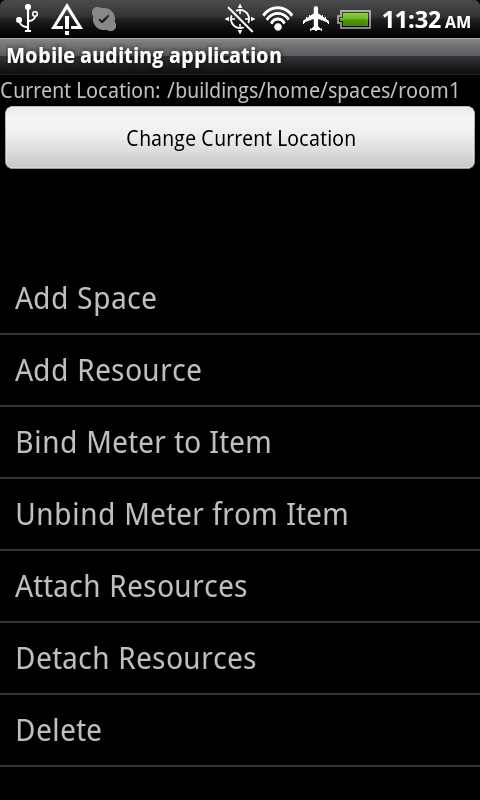


Figure 5. Screen shots from the Berkeley mobile auditing application. The figure on the left shows the menu items that the user chooses from. The figure on the right shows the results of a scan on a Lamp.

Before using the application, a user must generate a QR code. The QR code contains a URL that is used by the application look up the item that the QR code is attached to. Encoded in the URL is the unique identifier for the QR Code. This identifier is used as the name of the folder placed in the /qrc namespace, as described in the previous section.

**Registration**

To register an item, the user generates, prints, and attaches a QR code to the item and scans it. If the QR code is either not in the /qrc folder yet or not associated with an item (symbolically linked to an item in the /devices folder), the registration screen comes up. The registration screen lets the user to enter information about the item and permanently associates the QR code with the item --creates a folder for the item, places symbolic link form the /qrc/[id] folder to the /devices/[device] folder and updates the metadata tags attached to /devices/[device]. Any future QR code scans will display information about the item that was entered during the registration phase. The mobile application also allows the user to edit information by scanning and choosing the “edit information” option on the screen. Every item in the inventory, including devices and meters must be explicitly registered.

**Location management**

We require that all rooms be registered as well. Room registration provides a course-grained mechanism for locating items spread throughout the building. We also try to simplify the registration process by implicitly associating the “current location” of the user with the current item being registered. The user’s current location is set when they scan a QR code that’s attached to a location location in the building. Registering a room is similar to registering an item, except no metadata needs to be recorded for a room.

**Categorization**

As previously mentioned, part of the inventory collection processes involves categorizing the item. We used the taxonomy of miscellaneous and lower-power product [tax]. On the registration screen, the user enters the category and the application does a closest match to the text the user is entering.

**Bind/Attach**

Registration performs a binding between the QR code and the item it is attached to. We implemented two explicit associations that are important, *binding* a meter to a device and *attaching* two devices together. The bind operation involves a pair of swipes. 1) Swipe the QR code of the device and 2) swipe the QR code of the meter. This creates an association (symbolic link) between the meter and the item so that user can scan the item later and retrieve meter measurements associated with the item. It also creates an explicit association for detailed accounting after the fact. ‘Attach’ is very similar to bind, except that you attach two devices together. This is important to capture since there are situations where a meter is actually attached to a power strip which had multiple items attached to it. The association allows you to follow the link path between a meter and an item that has other items attached it.

**Initial audit results and applications**

We tagged and registered a total of 351 items and 139 rooms throughout a 141,000 square foot, seven-story building. For each load, we recorded power/current rating placed, so that we can later compare the actual readings with plated readings. Figure 6 gives a summary of the type of different types of items we collected.

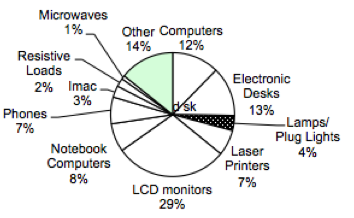
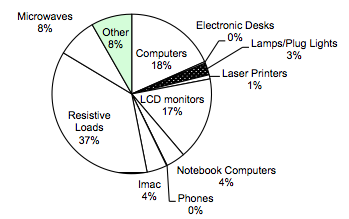


Figure 6. The figure on the left shows the decomposition of various types of devices. The figure on the right shows the plated power-draw of the various types of devices.

The vast majority of items were resistive loads, such as space heaters and coffee machines. The next largest category consists of computing equipment, such as laptops, LCD screen, and desktop machines. These were generated directly from the namespace and symbolic links from the taxonomy sub-tree. The first was a simple traversal of that hierarchy and the second was a search on the tags that contain plated power-draw figures.

The BMAS architecture allows you to incrementally deploy live metering and associate it with devices. In our initial iteration, we deployed a set of Acmes in a room and used the bind/attach operations to associate the meter feeds to the specific items inside that room. Instead of using the collected data to query historical data, we used the real-time feed to drive a usage visualization application. Figure 7 shows two screen shots of this visualization. The basic goal of the application is to inform users of which items are on and which are off. Occupants sometimes leave the lab without turning off all the devices, hence causing lots of energy to be used unnecessarily.

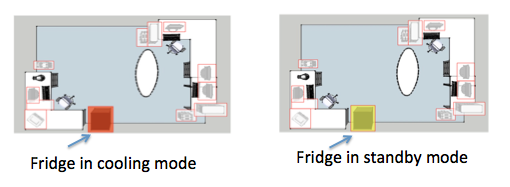


Figure 7. Live data associated with specific device in an office. This is an application built on top of the auditing infrastructure. It shows when appliances in the office are active and when they are inactive.

Finally, we wrote a script that queries the real-time feeds associated with power usage by devices in the building a generated a graph showing times of the day when usage was at its peak. Figure 8 shows this graph. We can pick out the weekdays and weekends. Weekdays are The weekdays see the highest power-draw and the weekends see about half the power draw and weekdays. Furthermore, we can see the peak hours of usage during the day. The times correspond to a typical workday between 10am and 8pm. With this visualization we can observe the work habits of the occupants of this particular building.

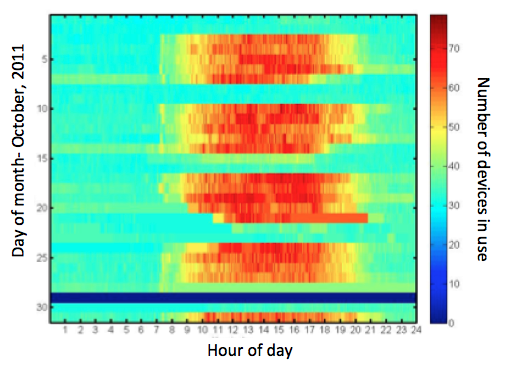


Figure 8. This figure shows the cumulative power draw for plug-load devices a single room in October 2011. We can differentiate between weekdays and weekends. Weekdays are identifiable by the clusters of red made up of 4-5 strips horizontally. Weekends and off-peak hours are is the rest. We can see that the typical weekday work hours are between 10am-8pm.

**Deployment time**

Doing the initial deployment took less than one day with two people. The registration of office items took only a few minutes. Entering information about a device takes a few seconds. The scan time per QR code takes approximately 0.5-2 seconds. A bind/attach operation takes approximately 3-5 seconds, as it involves two scans and a button press. Entering information about an item takes about 30 seconds.

**Continuous auditing, commissioning, iteration**

The applications demonstrate the generalizability of the architecture. We can integrate any kind of sensor feed into a common namespace upon which we can either run multiple studies based on historical data measurement or building applications that display real-time data.

The cost of the system scales well with the size of the deployments and can be done incrementally. Sensors can be added when they are available. The registration process is requires scanning QR code with your phone and inputting a bit of data, QR codes are easily generated and replaceable. They can be printed with any printer and taped on the item of interest. Furthermore, and perhaps to the most important aspect of the architecture, is that it allows the deployment to scale with the number of occupant participants, sensors, and data.

For the initial phase we used a team of undergraduates to run the initial building-wide deployment, and picked an office to have the occupants of that office tag and register their items. In return we offered energy services which kept them engaged over the period of time that we ran the initial study. We are currently working on a web site that the occupants can use to generate QR codes for their own items and to download the mobile application to register their devices with us.

Hosting a cloud-based instance of sMAP and StreamFS is also inexpensive and can be scaled incrementally. We are looking to integrate, not only plug-load data, but also building data that is extracted from the building management system. We are also going to add weather feeds and other information to support a number of applications on the infrastructure. The infrastructure that is in place continues to provide benefits to the occupants and application writers. If we continue to provide interesting applications to building occupants we incentivize them into participate in maintaining deployment consistency. This will allow us to continue to provide energy-related visualization and control services and will help us understand how energy is being used throughout the building better.

**Conclusion**

Miscellaneous electrical loads consume a large portion of the total building energy and it is on the rise. It is important to understand how MELs are used within buildings, so that we can learn how to better manage the ones that are deployed and potentially even design better electrical loads. In order to understand these more deeply, we must measure loads that fall into this category. Various studies have constructed multi-component instruments in order to collect and analyze data on MELs. However, there are various shortcomings in the design of such systems. They are generally expensive to put together, difficult to manage and maintain, are only useful for conducting the particular study, and do not particularly scale well.

This paper explores the use of IT to minimize the cost of doing such deployments, provides an incrementally deployable infrastructure that is useful for learning more about MELs and potentially integrates with all kinds of building data sources. The Berkeley mobile auditing system also makes use of building occupants to provide input and management scalability. We use mobile phones, QR codes, and cloud-based services to collect and manage the deployment data and metadata. QR codes serve as a cheap tagging mechanism that can be read by phones connected to a network to bridge the physical with the virtual, helping to maintain a consistent view of the deployment so that we can correctly interpret the data in our analysis. We were also able to write several useful applications that not only provide a value to auditors for doing specific studies, but for potentially inducing changes through visualization and control.

**References**

[LBNL\_MELS] Steven M. Lanzisera, Stephen Dawson-Haggerty, Xiaofan Jiang, Hoi Ying Cheung, Jay Taneja, Judy Lai, Jorge J. Ortiz, David Culler, Richard Brown. Wireless Electricity Metering of Miscellaneous and Electronic Devices in Buildings. 2011 Future of Instrumentation International Workshop.

[Porter2006] Porter S.F., Moorefield L. and May-Ostendorp P. Final Field Research Report: California Plug-Load Metering Study. Ecos Consulting, 2006.

[Moorefield2008] Moorefield L., Frazer B. and Bendt P. Office Plug Load Field Monitoring Report. Ecos Consulting, 2008.

[smap] Stephen Dawson-Haggerty, Xiaofan Jiang, Gilman Tolle, Jorge Ortiz and David Culler. sMAP – A Simple Measurement and Actuation Profile for Physical Information. 8th ACM Conference on Embedded Networked Sensor Systems (Sensys 2010

[sfs] StreamFS. <http://streamfs.cs.berkeley.edu>

[acme] Jiang X., Ly M.V., Taneja J., Dutta P. and Culler D. Experiences with a High-Fidelity Wireless Building Energy Auditing Network. Proc. of the Seventh ACM Conference on Embedded Networked Sensor Systems (SenSys’09). (Berkeley, CA, Nov. 4-6, 2009). Ed.: Association for Computing Machinery (ACM).

[qrc] ["QR Code Standardization"](http://www.denso-wave.com/qrcode/qrstandard-e.html). *QR Code.com*. Denso-wave.com. Retrieved 23 April 2009.

[doe] US DOE. 2009 Buildings Energy Databook. U.S. Department of Energy, Office of Energy Efficiency and Renewable Energy, 2009. Download at: <http://buildingsdatabook.eere.energy.gov/>

[tax] Nordman B. and Sanchez M. Electronics Come of Age: A Taxonomy for Miscellaneous and Low Power Products. Proc. of the 2006 ACEEE Summer Study on Energy Efficiency in Buildings. (Asilomar, CA, August, 2006). Ed.: American Council for an Energy Efficient Economy, Washington, DC.

[android] <http://www.android.com/>

[cmels] Berkeley National Laboratory, National Renewable Energy Laboratory, Oak Ridge National Laboratory, and Pacific Northwest National Laboratory. Monitoring and characterization of miscellaneous energy and business-process loads: Demonstration of field methods for studying diverse commercial environments. October, 2007. http://endusefiles.lbl.gov/public/CMELs/CMELs\_Joint\_Interim\_Report\_2010\_10\_07.pdf