

# Its Different: Hitchikers Tryst with Energy Consumption Patterns in India

Alice Security  
Department of Computer Science  
University of Southern California  
alice@example.edu

Bob Privacy  
Networked Embedded Systems Group  
Swedish Institute of Computer Science  
bob@example.se

## Abstract

This paper provides a sample of a  $\text{\LaTeX}$  document for ACM Sensys. It complements the document *Author's (Alternate) Guide to Preparing ACM SIG Proceedings Using  $\text{\LaTeX}$ 2 $\epsilon$  and BibTeX*. This source file has been written with the intention of being compiled under  $\text{\LaTeX}$ 2 $\epsilon$  and BibTeX.

To make best use of this sample document, run it through pdf $\text{\LaTeX}$  and bibtex to directly produce a pdf document.

## Categories and Subject Descriptors

H.4 [Information Systems Applications]: Miscellaneous; D.2.8 [Software Engineering]: Metrics—*complexity measures, performance measures*

## General Terms

Delphi theory

## Keywords

ACM proceedings,  $\text{\LaTeX}$ , text tagging

## 1 Introduction+Related Work

- Why buildings must be targeted for energy [10]
- Importance of feedback [7]
- Why we need deployments
- Deployments- Residential, Office [1, 5]
- Previous such residential deployments, some of which were presented in Buildsys itself [13, 4, 9]
- Some applications-NILM[12, 6], Fixture Finder [14]
- Specific learnings from our deployment, some of them complement the ones given earlier [13]
  - Glowing LED in night
  - Deployments should be transparent
  - Noisy server owing to dust (specific to developing countries)

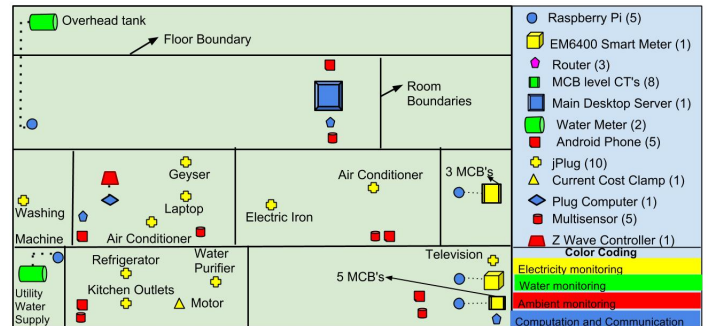


Figure 1: Schematic showing overall home deployment

- Electricity failure- as a consequence all systems should be capable to restart upon resumption of electricity
- Unreliable internet -Forcing to use Sense-Store-Upload paradigm
- Normalization -Voltage fluctuation, different measurement by different instruments

Also deployment was maintained as an open source project. Shows how we faced issues and tackled them. Also contains metadata log provided by the end user.

## 2 Deployment Overview

Over the past year, we have deployed sensors across 22 homes. While 20 of these homes have been instrumented only with smart electricity meters, 2 homes have been extensively instrumented with upto 33 sensors measuring electricity, water and ambient parameters. Figure 1 shows the deployment in a 3 storey home where 33 sensors, 5 single board computers and 3 routers were used. Detailed description of the deployment is given below.

### 2.1 Sensing Infrastructure

For our sensing, we took a “leave no stone unturned” approach, where we chose to monitor as many physical parameters (such as ambient conditions, electricity usage, water usage) and non-physical parameters (such as network strength etc.). However, it must be noted, that we chose to deploy sensors in a way that home users can continue their daily routines without getting affected. We describe the various

sensors used in our deployments below:

**Electricity monitoring:** A typical home electricity setup involves a meter which is installed by utility companies and measures overall electricity usage. Further electric cabling is divided into various Miniature Circuit Breakers (MCB's) which control separate circuits. Typical installations involve putting separate MCB's for heavier loads such as air conditioners and clubbing various lights, fans and other smaller loads into separate MCB. Further each individual appliance is controlled via a switch. There are two types of appliances- i) plug loads like refrigerator and electric iron, which need to be physically "plugged" into the sockets; ii) loads like lights and fans, which do not need to be "plugged" in by the user and can directly be switched on or off. We highlight the above described home electricity distribution in Figure 1. We have 3 different resolutions at which electricity can be monitored:

1. **Meter level:** We use Schneider Electric EM6400<sup>1</sup> smart meter to instrument the main power supply. While cheaper variants from the same company were available, we chose to use EM6400 as it also provides reactive power. This additional information has been known to be useful for NILM applications[12]. Figure 4a shows EM6400 smart meter deployed in the electricity panel. We had to put 30:5 ratio CT on the power mains coming from the grid to ensure that the load current was transformed to be within EM6400's permissible limits (5 A). EM6400 allows reading various registers to measure more than 40 parameters including: voltage, current, phase, power (apparent, real, reactive) and various other parameters, using Modbus over RS485-serial link.
2. **Circuit level:** We used our indigenously developed Current transformer (CT) monitor having split core CT's, which can be clamped to individual MCB's for monitoring circuit current. This device consists of ADC connected to a low cost microcontroller for sensing max. 8 analog channels simultaneously and pushing computed numerical values of current from each CT over serial. CT deployment is shown in Figure 4b.
3. **Appliance level:** We used jPlug<sup>2</sup> to measure individual appliance power consumption for 9 plug-load type appliances. We also used Current Cost based CT to measure the power consumption for electric motor (used to pump water), which is not a plug-load, yet has significant power consumption (approx. 700 Watts). jPlug and Current Cost CT deployment are shown in Figure 4c and Figure 4d respectively. jPlug makes a HTTP POST connection to log data. Current Cost exposes apparent power data over the serial port.

More details regarding sensors used for electricity monitoring are provided in Table 1.

**Water monitoring:** There are few differences in home water distribution in India when compared to developed countries. In India, there are separate water lines for drinkable and non-drinkable water. Also, overhead water tanks (typically 1000

liters capacity) are used to store water. Electric motors are used to push water against gravity to be stored in the tank. Thus, the flow can be summarized as follows: 1) Water from utility comes to the home; 2) Electric motor is used to aid in pumping the water up to the tank; 3) Water flows downward from the tank whenever water is consumed. Thus, we put a water meter at the inlet (coming from the utility) and the outlet from the water tank (flowing downwards) to capture the incoming (from utility) and outgoing (from tank) water consumption.

Digital water meters are very expensive and are not easily available in India. Thus, we chose to use Zenner Aquamter's multijet<sup>3</sup>, which are pulse output based water meters, which measure water volume flowing through them. These water meters have a 4 ma current loop **Manoj: Add details** and send a pulse every few liters. The precision is based on the quality of the sensor and the diameter of the water pipe. The water meter we used for overhead tank gives a pulse every 10 liters, whereas the one used for inlet supply from utility gives a pulse every 1 liter. These pulses can be measured using the circuit diagram for 4 ma loop shown in ... Figure 4e shows the water meter deployed inline at the overhead tank.

**Ambient conditions monitoring:** We used HomeSeer HSM100<sup>4</sup> based multisensors for monitoring motion, light and temperature. While motion is reported in event-driven fashion (i.e. whenever there is change in motion status, reading is reported), temperature and light are polled at 1 Hz. This multisensor uses the well known ZWave<sup>5</sup> protocol for communication. We also place Android phones at fixed locations and ran FunF journal<sup>6</sup> to log ambient parameters such as light and sound level.

**Miscellaneous:** Android phones, in addition to measuring ambient conditions, were also logging nearby bluetooth and wireless devices and strength of cell-tower signal. All home occupants were requested to keep their phone's bluetooth on during the duration of the experiment. We also logged outside weather conditions such as temperature, humidity and wind speed using publicly available API's from weather monitoring stations. We believe that this data can be used for many applications including: energy-apportionment (i.e. assigning energy usage to different occupants within the home) and localization.

Sensing infrastructure is summarized in Table 1.

## 2.2 Communication and Computation Infrastructure

In our experience we found that using a desktop computer for collecting data from sensors would be an overkill in terms of cost, processing power and physical space used. Although microcontrollers are cheap and occupy little space, they do not expose enough abstractions for high level programming and are often difficult to debug and are inherently hard to multi task. We thus decided to use Single Board Computers (SBC's) for sensor data collection. We used Raspberry

<sup>1</sup>[www.goo.gl/01edPS](http://www.goo.gl/01edPS)

<sup>2</sup>A variant of nPlug[11]

<sup>3</sup>[www.aquametwatermeters.com/multijet.html](http://www.aquametwatermeters.com/multijet.html)

<sup>4</sup>[www.homeseer.com/pdfs/guides/HSM100\\_Release\\_Notes.pdf](http://www.homeseer.com/pdfs/guides/HSM100_Release_Notes.pdf)

<sup>5</sup><http://en.wikipedia.org/wiki/Z-Wave>

<sup>6</sup><http://www.funf.org/journal.html>

Table 1: Sensing Infrastructure

Sensor name	Sensor type	Sampling frequency (Hz)	Resolution	Quantity	Communication	Observed parameters
EM6400	Electric Meter	1	Home	1	RS 485 Serial	Voltage, Current, Frequency, Phase, Power(Active, Reactive and Apparent), Energy
Aquamet multijet	Water Meter	5	Main supply and tank	2	Direct wire connection to GPIO	10 liter events for tank and 1 liter events for main supply
Homeseer HSM-100	Ambient multisensors	Light, temperature: 1 PIR: event based	Room	6	ZWave	Light, temperature and motion
Android phones	Ambient multisensors and network	Audio, light: 0.05 Nearby WiFi, cellular, bluetooth: 0.001	Room	5	Manually transfer files to PC	Audio features, light, nearby bluetooth, cell-tower, WiFi
Prototype CT	Electricity meter	20	MCB	8	Serial	Current
jPlug	Electricity meter	1	Appliance	10	WiFi	Voltage, Current, Frequency, Power (Active and Apparent), Energy, Phase
Current Cost	Electricity meter	0.1	Appliance	1	Serial	Apparent power

Pi<sup>7</sup> (RPI) and Ionics Stratus<sup>8</sup> as our SBC's. These SBC's are available for about the same cost (25 \$) as microcontrollers. Moreover, they possess 700 MHz ARM processors and support Linux based distributions. Thus, one can use the full Linux stack and code in higher level languages such as Python. These SBC's run from SD cards whose size can be chosen as per requirement.

We used a total of 6 SBC's of which 5 were RPI and 1 was Ionics Stratus plug computer. We used a 2 GHz Desktop PC running Linux as the main server where all the data was stored. The software stack running on the SBC's and how they interacted with the main server and collected data from different sensors is described below:

**EM6400 smart meter data collection:** EM6400 allows data to be read from its registers over RS485 using Modbus protocol. We connected a RPI to EM6400 using RS485-USB converter which exposes serial interface over USB on RPI. We developed a custom Python program based on pyModbus<sup>9</sup> library which would serially read 80 registers, using Modbus protocol, to compute 40 electrical parameters as described in Table 1. It would store these 40 parameters along with UTC timestamp in a CSV file. A new CSV would be created every 15 minutes. A background program would periodically try to upload CSV files older than 15 minutes, to the main server, where the data would be dumped in MySQL database.

**Current Cost data collection:** Although Current Cost has its own cloud based API platforms, we chose to collect data from it locally, in order to avoid data losses occurring due to network failures. Current Cost provides power data in xml format over the serial interface. We thus connected the USB cable-out from the Current Cost receiver to RPI and wrote a simple Python script to read data serially using pySerial<sup>10</sup> and store it in a CSV file alongwith UTC timestamp. CSV

creation and uploading to central server was done in a similar fashion as was done for EM6400.

It must be noted that we used the same model- create new CSV periodically, upload older CSV periodically to main server where they are dumped into MySQL for all sensors. We call this model Sense-Local Store-Upload and describe it in more details in section Section 2.3

**MCB data collection:** Our custom built CT based sensor solution for collecting current data from different MCB's exposes data serially which is read using pySerial on a RPI and processed further using Sense-Local Store-Upload.

**jPlug appliance data collection:** jPlug makes a HTTP post every second with upto 10 electrical parameters. We saved this data directly on the server machine where a web daemon listened to requests from jPlug and dumped them in MySQL.

**Water data collection:** Based on circuitry explained in Section 2.1, we used GPIO header on RPI and wrote an interrupt driven program in Python to detect 10 liter and 1 liter events for the tank and supply water meters respectively. We found that noise introduced in the circuit due to long cable lengths led to a lot of false events. Thus, we modified our program and polled at a frequency of 5 Hz to obtain GPIO status and further processed this data using Sense-Local Store-Upload.

**Homeseer HSM100 data collection:** We wrote custom wrappers around OpenZWave<sup>11</sup> program to collect temperature and light information on a per second basis, and motion information based on events. ZWave controller which controls all the ZWave based sensors was connected to the plug computer over USB. Data was processed further using Sense-Local Store-Upload. Figure 4g shows plug computer collecting ambient sensor data from ZWave controller.

**Android data collection:** We used FunF journal which would store data inside phone's SD card. We would take a dump once every 15 days and empty the SD card for further data collection.

<sup>7</sup>[www.raspberrypi.org](http://www.raspberrypi.org)

<sup>8</sup>[www.ionics-ems.com/plugtop/stratus.html](http://www.ionics-ems.com/plugtop/stratus.html)

<sup>9</sup>[www.github.com/bashwork/pymodbus](https://www.github.com/bashwork/pymodbus)

<sup>10</sup>[www.pyserial.sourceforge.net](http://www.pyserial.sourceforge.net)

<sup>11</sup>[www.code.google.com/p/open-zwave](http://www.code.google.com/p/open-zwave)

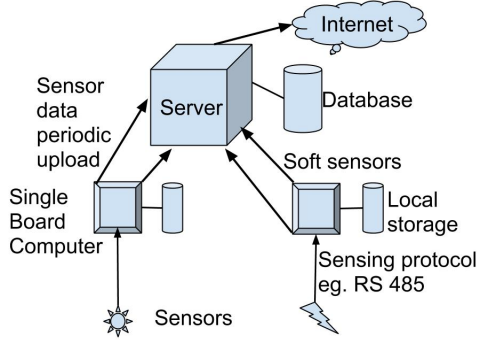


Figure 2: Sense-Local Store-Upload

**Weather data collection:** Electricity failure and reliable Internet are well known problems in India and are highlighted in Section 3. Owing to these problems, we chose to collect weather data from 3 different weather stations, in our collaborator’s institute in the USA.

In our deployment a lot of issues pertaining to SBC’s were found. For instance, the OpenZWave program that we used would create log files for its own diagnostics. This eventually ate up the 512 MB flash drive space on the plug computer. We fixed this by deleting older logs. This encouraged us to develop soft-sensor streams whereby we collected hard disk space, ping success, CPU utilized, RAM left, temperature of processor for all the computing devices including the server at regular intervals. These soft-sensor streams can be used for alerting mechanisms and can also be used for fault diagnosis. Similar soft-sensor streams have also been used in previous work[13] where they used them for alerts.

We found that one WiFi router will not suffice for a 3 storey building. This has also been reported in previous work on residential deployments[13]. We thus used 3 routers, where the router on the first floor acted as the host and routers on ground and second floor were bridged to it. More details regarding the need of multiple routers and network availability in homes is described in Section 3 and Section 4.

### 2.3 System Architecture

Various middleware systems such as sMAP[8], Building Depot[2] and SensorAct[3] have been proposed in the past for sensor data collection. However, we found that they are not fine tuned to our needs arising from faulty internet, power failures, etc. Moreover, these systems provide a lot more features than we intend to use. Thus, based on our experience and previous work[13], where importance of simplifying the architecture are proposed, we propose Sense-Local Store-Upload. Sense-Local Store-Upload involves local storage and periodic data upload. As explained above, we used SBC’s to collect data from sensors. This data was **locally stored** in form of comma separated value files (CSV) and was **periodically uploaded** to the main desktop server. In case the upload failed, it was retried the next time again. Each SBC had sufficient flash based local storage to accommodate sensor data for a few days.

Web applications running on the server allowed a home

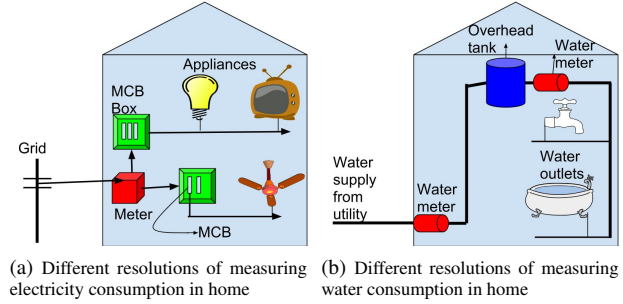


Figure 3: Different resolutions of measuring electricity and water consumption in home

user to locally visualize his data from multiple streams. Further, data from the server can be copied to cloud based servers such as Dropbox where applications such as analytics, alerts can be done, allowing outside users and specifically researchers to keep an eye on the deployment sitting in their labs. Figure 2 explains Sense-Local Store-Upload architecture. The salient features of our architecture are as follows:

- **Decoupled sensing and data uploading:** This ensures that an error in one does not prevent the other action from occurring correctly and thus allows easier debugging. Moreover, this design choice of decoupling comes from the well established principles of software engineering.
- **Minimal internet requirement:** Internet is required **only** when outside researchers wish to view and analyze collected data in realtime. Internet failure does not have any impact on sensor data collection.
- **Reduced load on server:** Since data is uploaded periodically in larger chunks rather than sending data for each sensor packet, computation and bandwidth requirements are greatly reduced.

### 3 How is this deployment different?

In this section we discuss unique aspects brought forward from our deployment. Some of the key aspects are as follows:

**Unreliable grid is commonplace:** Load shedding or rolling blackout are common in developing countries, owing to several reasons such as improper infrastructure management, high demand and low electricity production. Power cuts are more common in summers when load is higher due to usage of air conditioners. As a result, voltage fluctuations or brownouts are also common. Figure 6b shows power cut in number of hours per day during a two month period in summer of 2013. Electricity outage for upto 12 hours has been reported on some days. Figure 6b shows the duration in minutes of power outages. 107 power outages were found in 61 day period from May to July. While the average power outage lasted about 1 hours, there were outages lasting upto 9 hours. Figure 6c shows the power outage distribution by hour of the day. It can be seen that there are maximum outages around 10 AM in the morning and around midnight.



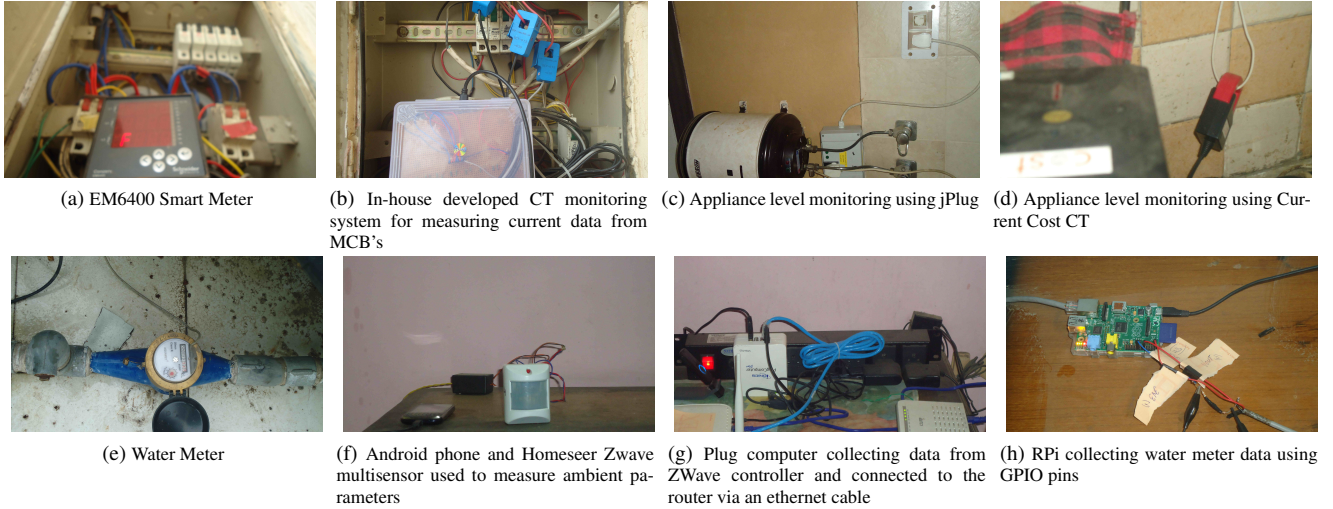


Figure 4: Sensing, computation and communication equipment used for deployment

These times correspond to office time and night time when air conditioners in home are on. We believe that this indicates heavy load on the grid. Figure 6d shows the voltage fluctuations in a typical day, where it can be seen that voltage is fluctuating from 180 to 250 Volts. We can observe that voltage drops during peak load hours. This is in consonance with previous work[11], which hypothesized that frequency and voltage indicate load on the grid. Figure 6e shows fluctuation in frequency in a typical day.

Due to unreliable nature of grid, we ensured that all our systems were capable of automatically starting after a power outage. We put our scripts for data collection and upload in Linux startup script. This provided us another advantage. If we observed that a system was down, we could just ask the home occupant to repower the system again. This ensured that there was minimal data loss till the time our researchers could visit the site and repair the fault.

**Unreliable internet connectivity:** During our prototype deployment, where we used cloud based Homes have poor connectivity. This forced us to develop a different paradigm which we call Sense-Store-Transfer. This is shown in Figure 6f

**Importance of meta data collection:** We collected meta-data associated with electrical appliances throughout our deployment. We collected parameters such as appliance name, age, mode of usage (for instance air conditioner set temperature). We believe this detailed metadata can enhance NILM and can provide useful insights for conserving electricity. For instance, in one of the home refrigerator was sent for repairing on 2<sup>nd</sup> July. Figure 5a and Figure 5b show the active power consumption before and after repair. We found that after repair the refrigerator was configured in coolest mode (lowest temperature), while before repairing, it was configured in least coolest mode (highest temperature). After repair the refrigerator was found to be consuming 1KWh more. Moreover, its duty cycle pattern was modified significantly. This metadata can be useful to NILM approaches which model appliance usage duration as one of their param-

eters.

**Load specifics:** Electrical load setting in India are different from that in the US and Europe. There are no automated loads. HVAC systems are decentralized. Thus, air conditioning is at room level and water heating is done using geysers at bathroom levels. From our deployments we observed that air conditioners and geysers account for upto 70% and 50% of overall home electricity in summers and winters respectively. Thus, any small improvement in efficiency of these two appliance can significantly lower the home electricity consumption.

There are also special appliances such as water motor and water filter, owing to which there is energy embedded in water. Water motor, earlier described in section Section 2.1, is used to pump water upto water tank on roof. It was observed that without motor it took 8 seconds to fill 1 liter of water into the tank and after switching on the motor it took 4 seconds. Also, water filter

**Lack of quality measurement devices commonly found in US and Europe:** ZWave sensors for Indian frequency (865.2 MHz) are not available. Thus, we imported from European frequency (868.4 MHz) based ZWave multisensors and plug load monitors. Import duty and shipment delays make them an expensive choice. As explained above, ZWave based plug load monitors could not be used in our deployments, since they need to be explicitly turned ON. Further other alternatives like Current Cost based CT's would involve exposing live wire from appliances, an option which was rejected by home occupants. Thus, we had to resort to jPlug, which is a prototype solution.

#### 4 Hitchiker's guide revisited

In this section, we discuss similarities between our deployment and those explained in previous work. We further explain how we tackled these issues. Some prominent similarities are:

**Homes are hazardous environments:** We observed that one of our multisensor would fail after power outage. Initially, we felt that the sensor was faulty, but when other sen-

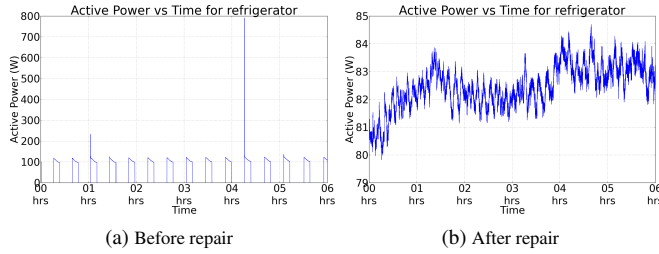


Figure 5: Refrigerator power consumption



Figure 7: Common problems in residential deployments

sors showed the same behaviour in the same location. We figured that this behaviour was due to the fact that this multisensor was put on inverter point (battery backup) and would not go down even on power outage. Thus, when power resumed, ZWave controller was not able to add this multisensor to its network. We resolved this by putting the multisensor at a non-inverter point. We also observed that one of the multisensor always indicated motion. We were able to solve this problem primarily by replacing its power supply. However, within a week this problem resurfaced. We believe it is due to faulty power supply in that room. The home occupants did not allow a thorough investigation of that electrical socket and we had to do away with that data.

Although we took caution of using zip-ties to prevent hanging wires, we observed data loss in one of our multisensor and Android phones, which went out of power due to wire snag. This is shown in Figure 7b. Further, even after 1 month rigorous testing in lab, we found that 60 new issues (service complaints) were registered when we moved the deployment to the first home. This is only a testimony that homes present a unique scenario which can not be emulated well in laboratory settings.

**Aesthetics matter:** As stated in previous work, sensor LED's can be bothersome to home occupants, particularly in the night. Our 33 sensor deployment had introduced more than 50 LED's in the home. Figure 7a shows our sensor LED's blinking in the night. For our current deployment, choosing appropriate sensor location sufficed. However, for future such deployments, we intend to build better casing. Also, the home occupants complained of buzz like sound coming from the desktop server that we had put. We figured that this noise was due to the fact that its CPU was clogged due to dust. This is another unique aspect introduced in the Indian setting. Thus, any long lived deployment should have routine maintenance.

**Homes are not designed for sensing:** We observed that the data collected from our ground floor MCB's (5 in number) was noisy. This was due to the fact that the MCB's are very close (shown in Figure 7c) to each other and this causes interference in our CT monitoring circuit. However, on the 3 MCB's on the first floor, there was adequate gap amongst MCB's and data was of good quality. We thus believe that homes are not designed for sensing and one should be prepared to lose some data quality due to that.

**Redundancy-Accounting for sensor failure:** During our deployment 3 jPlugs and 1 multisensor stopped functioning properly. Fortunately, we had accounted for such failure and had additional sensors ready. We believe that one should beforehand account for sensor failure in residential deployments.

**Homes have poor connectivity:**

## 5 Conclusions and Future Work

## 6 References

- [1] Y. Agarwal, B. Balaji, S. Dutta, R. K. Gupta, and T. Weng. Duty-cycling buildings aggressively: The next frontier in hvac control. In *Information Processing in Sensor Networks (IPSN), 2011 10th International Conference on*, pages 246–257. IEEE, 2011.
- [2] Y. Agarwal, R. Gupta, D. Komaki, and T. Weng. Buildingdepot: an extensible and distributed architecture for building data storage, access and sharing. In *Proceedings of the Fourth ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Buildings*, pages 64–71. ACM, 2012.
- [3] P. Arjunan, N. Batra, H. Choi, A. Singh, P. Singh, and M. B. Srivastava. SensorAct: A Privacy and Security Aware Federated Middleware for Building Management. In *Fourth ACM Workshop On Embedded Sensing Systems For Energy-Efficiency In Buildings*, BuildSys 2012, 2012.
- [4] 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*, pages 43–56. ACM, 2008.
- [5] N. Batra, P. Arjunan, A. Singh, and P. Singh. Experiences with occupancy based building management systems. In *Intelligent Sensors, Sensor Networks and Information Processing, 2013 IEEE Eighth International Conference on*, pages 153–158, 2013.
- [6] K. Carrie Armel, A. Gupta, G. Shrimali, and A. Albert. Is disaggregation the holy grail of energy efficiency? the case of electricity. *Energy Policy*, 2012.
- [7] S. Darby. The effectiveness of feedback on energy consumption. *A Review for DEFRA of the Literature on Metering, Billing and direct Displays*, 486, 2006.
- [8] S. Dawson-Haggerty, X. Jiang, G. Tolle, J. Ortiz, and D. Culler. smap: a simple measurement and actuation profile for physical information. In *Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems*, pages 197–210. ACM, 2010.
- [9] S. Dawson-Haggerty, S. Lanzisera, J. Taneja, R. Brown, and D. Culler. @ scale: Insights from a large, long-lived appliance energy wsn. In *Proceedings of the 11th international conference on Information Processing in Sensor Networks*, pages 37–48. ACM, 2012.
- [10] M. Evans, B. Shui, and S. Somasundaram. Country report on building energy codes in india. *PNNL*, 177925, 2009.
- [11] T. Ganu, D. P. Seetharam, V. Arya, R. Kunnath, J. Hazra, S. A. Husain, L. C. De Silva, and S. Kalyanaraman. nplug: a smart plug for alleviating peak loads. In *Proceedings of the 3rd International Conference on Future Energy Systems: Where Energy, Computing and Communication Meet*, page 30. ACM, 2012.
- [12] G. W. Hart. Nonintrusive appliance load monitoring. *Proceedings of the IEEE*, 80(12):1870–1891, 1992.
- [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*

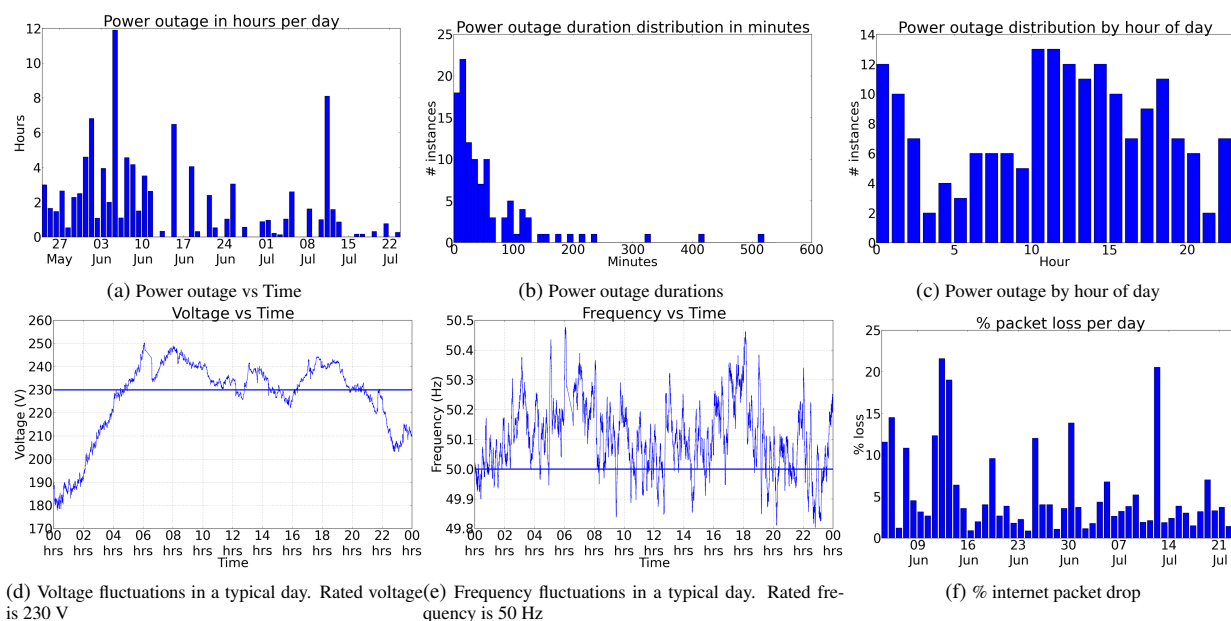


Figure 6: Unreliable internet and grid

*Conference on Embedded Networked Sensor Systems*, pages 232–245. ACM, 2011.

[14] V. Srinivasan, J. Stankovic, and K. Whitehouse. Fixturefinder: discov-

ering the existence of electrical and water fixtures. In *Proceedings of the 12th international conference on Information processing in sensor networks*, pages 115–128. ACM, 2013.