Building a Mobile Application for a Smart Power Grid

Hieu Nguyen
University of Southern California
EE657 – Spring 2012

***Abstract*—Following the trend towards home automation and mobile technologies, smart grids will be used to intelligently monitor and control all electricity flowing in the power grid system. The smart grid tracks energy usage in real-time and works with the consumer to efficiently meet electrical power demands while minimizing energy usage. However, a major obstacle in further optimizing smart grids involves the communication between the infrastructure and its users. This project implements a mobile application to transform smart grid sensor data into useful knowledge formats that encourage sustainability. Utilizing the Cloud, the mobile app provides scalable operations in an information-rich environment, allowing distributed stream processing for low latency response.**

I. Introduction/Background

Environmental challenges with greenhouse gas emissions and energy inefficiencies have triggered a reevaluation of today's electric power grid. Modern energy solutions, which utilize renewable energy resources like wind and solar, seek to improve the energy efficiency in our homes, businesses, and public institutions while delivering secure, affordable, and low-carbon energy. However, today's electric power grid was not built to handle the large and increasing amounts of renewable energy production. The evolution from a system composed almost entirely of steady state fossil fuels to clean renewable energy sources requires a smarter electricity system that facilitates the integration, transmission, and consumption of energy.

One solution known as the "smart grid", or an advanced metering infrastructure (AMI), has the potential to revolutionize the global energy market while supporting emerging technologies such as plug-in hybrid electric vehicles. The smart grid is an upgrade over the current power management and distribution systems because it aims to deliver electricity in an optimal way from source to consumption. This intelligent system applies sensing, measurement, and control devices to electricity production, transmission, distribution, and consumption subsystems. The smart grid uses these smart meters and grid sensors to collect accurate, real-time information on the actual energy usage factors in order to enable better energy management and minimize power disruptions. Within this infrastructure, buildings such as shopping malls, hospitals, offices, industrial warehouses, and residential homes can take advantage of the smart grid to optimally control alarm systems, indoor climate controls, elevators, lighting, appliances, furniture, and other electronic devices.

For many utility providers, power usage information is only available once a month, and consumers receive monthly bills with static rate slabs. With the smart grid, power usage can be measured on the order of minutes, which helps utilities balance demand, reduces expensive peak power use, and provides a better deal for consumers by allowing them to see and respond to real-time pricing information. Bidirectional communication can then be utilized to develop environments that regulate themselves—balancing occupant comfort with energy use. For example, when power is least expensive, the consumer can allow the smart grid to turn on selected home appliances such as washing machines, refrigerators, or HVAC systems. At peak times or when residents are away, the smart grid can reroute energy usage to reduce demand. By giving consumers more information about their energy footprint and the means to manage their electricity consumption, the smart grid can provide real-time situational awareness and influence more energy-efficient consumer behavior.

One potential drawback of the smart grid is that the dynamic tracking and comparison of device loads are both network and compute intensive. According to some estimates, smart grid networks can collectively eclipse the size of the Internet [4]. In order to handle the vast amount of sensor data, a scalable Cloud infrastructure has been proposed as the software platform to collect, manage, and analyze the information. Clouds provide a flexible model for growing processing and data storage over time, allowing virtual compute resources to be acquired on demand and released when no longer required. Thus, the utility can scale as the number of consumers increases and more complex algorithms are incorporated into the application pool. Clouds also provide redundancy for mission-critical applications,

implicit replication of data storage, and its data centers are equipped to handle millions of open network connections with small data exchanges per connection.

The smart grid will be composed of a combination of public and private Clouds for data privacy, security, and reliability considerations. In other words, a core set of internal, regulated services may be hosted within the utility's privately-hosted Cloud while the public Cloud is used for a different set of services and to off-load applications that exceed the local computational capacity [4]. The ability of the smart grid to continuously learn and rapidly incorporate new information sources and predictors is essential for a sustainable software architecture as the algorithms used for data analysis, mining, and decision making will change over time. As a pervasive computing application area, the smart grid will dynamically mitigate energy use by measuring, inferring, manipulating, and leveraging human behavior and context across various domains and environments.

II. Approach

The consumer is a key element in the functioning of the smart grid, as it must respond to human behavior and user preferences to regulate power consumption. While the smart grid aggregates diverse sources of information such as weather, traffic, event schedules and even social network data along with the energy usage data from its smart sensors, energy consumers need guidance on using this information meaningfully to make informed decisions. Consumers will find it easier if the information is interpreted for them and they are provided with specific and appropriate actions to take [10]. A combination of visual and textual cues help consumers understand their current bill, assist with demand response optimization, and reduce their monthly power usage and costs. This project explores interacting with customers through a mobile application for smart phones that works with Cloud-hosted services to encourage sustainability and energy conservation at the University of Southern California (USC).

A. USC Smart Grid

As part of the economic stimulus, the U.S. Department of Energy (DoE) has invested in improving energy conservation in buildings, identifying new energy sources, and making the electric grid smarter. The DoE has awarded the Los Angeles Department of Water and Power (LA DWP) $120 million to test and evaluate innovative smart grid technology, and demonstrate it on a regional electric grid. The Los Angeles Smart Grid project is the result of this funding and a collaborative effort between the LA DWP, USC, UCLA, NASA Jet Propulsion Laboratory, and the USC Information Sciences Institute.

In order to experiment and evaluate smart grid technologies, the USC campus is serving as a micro-grid test-bed for the Los Angeles Smart Grid project. The USC campus contains over 33,000 students and 13,000 faculty members, who collectively consumed 148GWh in 2010—making the university the largest private customer for the LA DWP [4]. The campus is comprised of over 100 major buildings—class rooms, residence halls, administrative offices, restaurants, laboratories, etc.—each with varying electrical, heating, and cooling facilities. Currently, each building has the ability to measure its total energy usage at 1 minute intervals, while newer buildings have room-level or equipment-level energy usage measurement granularities. Because of these characteristics, the USC campus forms a "city within a city" and is an ideal testing ground for running smart grid experiments. Upon completion of the research, the successful campus-scale models developed for the USC Smart Grid can be scaled up to serve the Los Angeles Smart Grid.

The energy data collected across all campus buildings is currently being aggregated using a proprietary software system accessible through a Meronymy SPARQL database server, which is a high performance cross-platform Resource Description Framework (RDF) NoSQL database management system. By querying energy usage measurements from the database using the SPARQL query language, a central control center can override heavy energy sinks such as the HVAC equipment that occasionally consume up to 50% of the total campus power [4]. The current electrical control system is only utilized under severe overheating or budgeting conditions and still lacks an intelligent system that can perform automated demand response optimization and interact with consumers for energy reduction.

B. USC Smart Grid App

With the advent of mobile devices, information on the Internet is readily available at any time. Mobile applications provide an interactive and educational experience that has been found to be more engaging and effective than traditional web-based applications [3]. Utilizing the benefits of mobile applications, the USC Smart Grid App is an effective feedback method that persuades and motivates energy consumers towards sustainable behavior. The mobile app serves as a management console for a user's utilities, allowing users to manually control powered devices from their phones, view energy usage statistics, and participate in communal energy-saving initiatives.

The USC Smart Grid App promotes voluntary energy reduction by providing personalized incentives to conserve. These natural language-based incentives include positive encouragement for energy saved, reminders to turn off appliances, and comparisons with peers. Overall, the app makes it easier for consumers to comprehend and interpret their energy usage and configure their home or building area networks to be controlled by the smart grid on their behalf.

While mobile applications are useful for communicating with people by acting as thin clients for display and interaction, the heavy computing and data management of the smart grid is usually performed in the Cloud. Similarly, the USC Smart Grid App relies on Cloud-hosted backend services to manipulate smart grid sensor data, provide reliable information sharing, and facilitate the scalability of the mobile application. This project focuses on the interaction between the Cloud and mobile devices, as well as technical issues in the Cloud such as data sharing, compute-data co-locality, synchronous and asynchronous communication, latency and responsiveness, and scalability.

III. Implementation

The "USC Smart Grid App," shown in Fig. 1, was developed for mobile devices running Android, a Linux-based operating system that is free and open-source to developers. This mobile app utilizes the location-aware technology of GPS and the interactivity of high-density graphics and touchscreen displays to create an engaging, interactive front-end experience for users. In the backend, the mobile app relies on Cloud-hosted services to interface with the USC Smart Grid. These cloud computing, database, and storage services are provided by Amazon Web Services (AWS) in this proof-of-concept implementation. Realistically, the Cloud-hosted services would be provided by interfacing with a private Cloud running an open-source infrastructure-as-a-service such as Eucalyptus. However, at the time of this project, the Cloud infrastructure for the USC Smart Grid was under construction, so AWS was used as a placeholder.

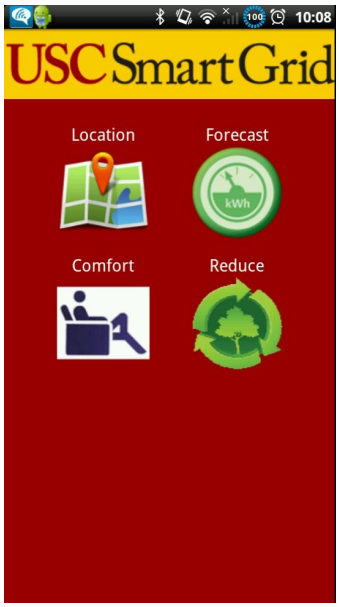


Fig. 1. The USC Smart Grid Mobile App.

AWS is a Cloud computing platform that offers a flexible pay-as-you-go pricing model. A central part of AWS, the *Elastic Compute Cloud* (EC2) provides resizable compute capacity in the cloud and allows scalable deployment of applications by virtualizing compute resources. For the USC Smart Grid App, a free-tier micro instance running a 64-bit Ubuntu Oneiric Server AMI was used. The micro instance is a single-core, 613MB virtual machine that is configured as an Apache webserver to run PHP scripts. The EC2 security group opens TCP ports 22 and 80 for SSH and HTTP access. The AWS *SimpleDB* is a distributed NoSQL database service that allows storage, transfer, and throughput over the Internet. In this implementation, SimpleDB is used to store user data and preferences for use with the smart grid. Amazon's S*imple Storage Service* (S3) provides a fully redundant data storage infrastructure for storing and retrieving any amount of data through web service interfaces. Amazon S3 is used to store the users' server-generated energy usage graphs.

IV. Application Features

The USC Smart Grid App has four modes of operation called "Location", "Forecast", "Comfort", and "Reduce." These four features provide the user with a basic bidirectional interface to the USC Smart Grid's information. Once the app is installed, it generates a random universally unique identifier (UUID) to identify the particular installation of the application. The UUID is a 128-bit number, meaning the chance of generating a duplicate UUID is 4×10^{-16}

[7]. The 64-bit android_id is generated when the device first boots and should remain constant for the lifetime of the device, allowing the smart grid to identify a unique physical device.

A. Location Activity

As people migrate or relocate within the USC campus, the smart grid will need to adapt to better serve its customers. The location activity on the USC Smart Grid App is responsible for keeping track of the user's location, allowing a more personalized carbon footprint monitoring. By knowing the room or building a user is located, the smart grid can access the associated usage statistics and present the user with location-specific energy conservation information.

Using GPS, the user can share his/her geolocation with the smart grid through a cloud database. When 'Location Sharing' is enabled, the mobile device calls its GPS to read the current location, returning the timestamp of the request and the latitude and longitude coordinates. As shown in Fig. 2, the phone sends this GPS data along with the user ID in an HTTP POST request to the EC2 server. A server script then runs a point-in-polygon algorithm to determine if the users geolocation coordinates are within a particular building on campus. For each building, an array holding the coordinates of the building's polygon vertices is defined. The point-in-polygon algorithm programmatically checks if the point is within the polygon by count how many times a line drawn from the point (in any direction) intersects with the polygon boundary. If they intersect an even number of times, then the point is outside. This algorithm determines if the user is on campus, and if so, which particular building. The results of the point-in-polygon algorithm are put into SimpleDB as attributes for an item defined by the user ID. This database entry will hold all user data and preferences.

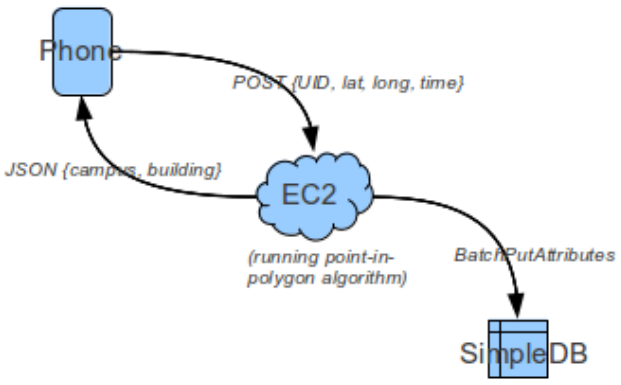


Fig. 2. Data flow for the "Post Location" operation. The user's geolocation information is stored in a SimpleDB domain.

After the information is finished writing to the cloud database, the server responds to the app with a JSON object containing the user's determined location information. The result is then displayed in italics in yellow font, as shown in Fig. 3a, where the geolocation coordinates correspond to the user being on campus in the Vivian Hall of Engineering (VHE). As long as location sharing is enabled, the GPS module in the device will continue to send updates to the smart grid periodically at 15 minute intervals if the user has moved more than 3 meters. This allows the smart grid to keep track of a user's location dynamically as he/she migrates from building to building. Alternatively, as shown in Fig. 3b, the user can also specify their current building location. This option is particularly useful when accurate GPS readings are difficult to obtain while underground or indoors. The selected building is updated in the user's SimpleDB entry to maintain location coherency.

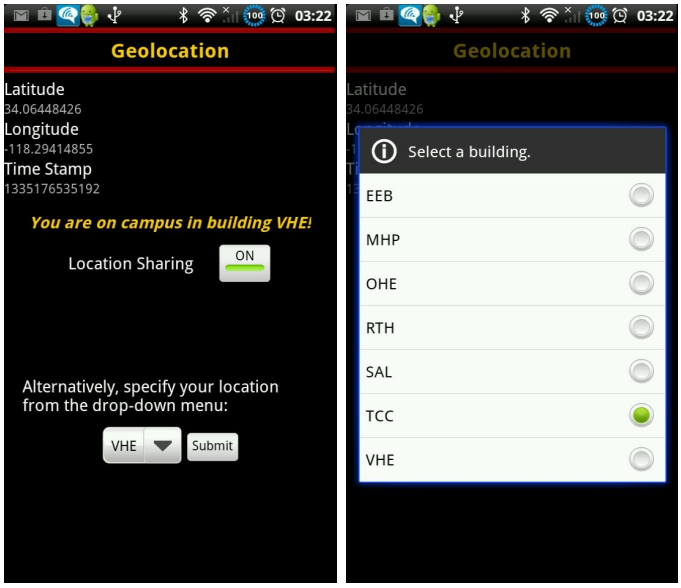


Fig. 3. Location Activity UI. a) Location Sharing results using GPS. b) Choosing the Tutor Campus Center (TCC) as the current location.

B. Forecast Activity

The forecast activity enables users to view energy usage statistics with interactive and eye-catching plots, charts, and forecasts. This feature interfaces with the sensor-data-logging database to obtain the energy usage kWh-readings for a particular building. Using this information, the smart grid can apply demand forecasting to accurately predict the occurrence of a peak load—a situation where the demand for power approaches or goes beyond the current power generation capacity of the utility [4]. Long- and short-term demand forecasts on the order of

days to minutes can be used to initiate load curtailment response.

Users can also use the forecast activity to view their energy usage history. In Fig. 4, the bar graph shows the usage history in the past 2 hours at 15-minute intervals for the Tutor Campus Center (TCC). Based on this plot, the user can determine that the building's energy usage has been increasing in the past couple of hours, and can then take steps to minimize energy consumption. The plots provide customers with a visual interpretation of data to showcase trends and patterns over time.

As illustrated in Fig. 5, when a user clicks "Get Forecast", the mobile device sends an HTTP GET request to the EC2 server. This request includes the date and time of the request, as well as a user ID and building. If a building was not specified, the server retrieves the user's previous location data from their entry in SimpleDB. Then using the building, time, and date, the server queries another database that holds the live kWh measurements. A plot is generated with the query response and that image is uploaded to Amazon S3 cloud storage. When the S3 upload is complete, the server returns the URL to S3 object to the USC Smart Grid App, which can then download and display the image.

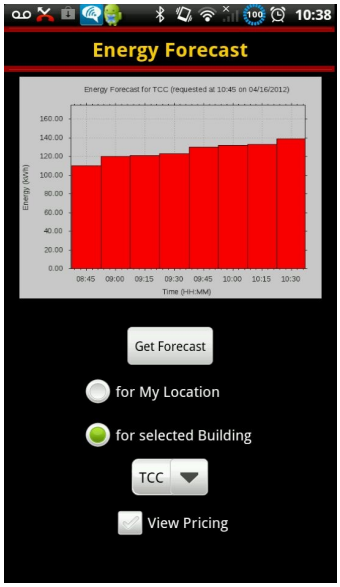


Fig. 4. Energy history plot for TCC in the Forecast Activity.

As smart grid data becomes more pervasive, the forecast activity can display more interesting figures. For example, the kWh-readings can be correlated with the dynamic utility rates to determine the fluctuating utility costs based on usage. For greater accuracy, the forecast activity could include environmental observations such as cloud cover and wind speed, real-time traffic patterns, schedules of large events and conventions, and equipment duty cycles to provide users with additional levels of detail to understand the relationships between these external variables and energy consumption. Using demand forecasting to predict peak loads, the smart grid can select an effective response strategy to curtail energy usage at the right time.

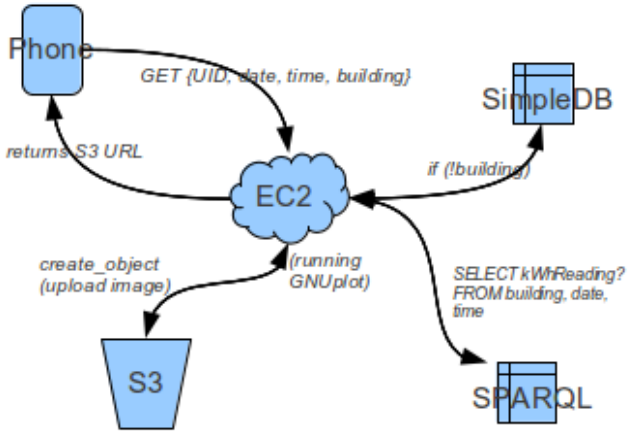


Fig. 5. Data flow for the "Get Forecast" operation.

C. Comfort Activity

The comfort activity presents an environment where users can inform the smart grid of their personal preferences. For example, a customer can use the app to tell the Smart Grid that he/she is feeling cold. This preference is consequently updated in SimpleDB, which keeps track of all of the user's information in the database. The smart grid can then evaluate the user's preferences and location to develop a strategy for optimizing user comfort with energy usage—perhaps automatically raising the regulated temperature in the room to reduce HVAC usage and warm up the customer.

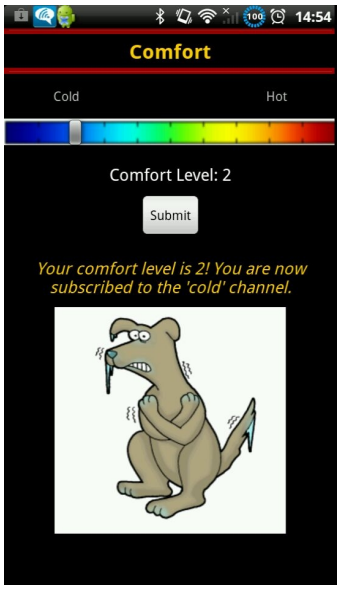


Fig. 6. Selecting a "cold" comfort level in the Comfort Activity.

When a user's preferences or location changes, the USC Smart Grid App subscribes to the corresponding push notification channel. Push notifications allow the app to notify the user of new messages or events even when the user is not actively using the application. With push notification channels, the smart grid can broadcast a message to all users, or just notify a subset of users to give personalized information to the users in a certain building or to those who are feeling a certain comfort level.

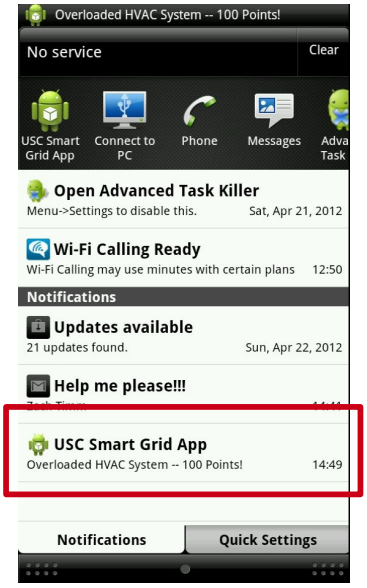
The USC Smart Grid App utilizes a third-party mobile backend-as-a-service called Parse to facilitate the push notification channels and handle the message passing. The Parse library provides push notifications by running a background service that keeps an Internet connection to the Parse push servers. Once subscribed to a channel, the subscription information is cached on the device's disk if the network is inaccessible and transmitted to the Parse servers as soon as the network is usable. The subscribed user can then receive notifications on that channel from the smart grid. Parse has an auto-scaling platform, flexible schema, and REST API for sending push notifications from a web console. The Parse Basic Plan is free and comes with 1 million API requests and 1 million pushes per month. Additional requests and pushes can be purchased if necessary.

D. Reduce Activity

When a device receives a push notification, the app's icon and message appear in the status bar. And when the user taps the notification, they are sent to the reduce activity. The reduce activity accesses the user preferences and location to generate a personalized sustainability newsfeed unique to the user. Newsfeed content could include predicted information and advice on how consumption could be reduced, friendly reminders with accompanied expected energy savings, positive encouragement to encourage voluntary reduction, explanations of usage graphs, or possible reasons for high bills. The reduce activity is essentially a strategic way to present interpreted information to the customer.

If the smart grid is forecasting a peak load at the user's current location, it can send a push notification to initiate curtailment through a specific suggestion/action. These actions are completely voluntary, but may have associated incentives for participation. Other incentives such as friendly competitions and peer pressure can also be utilized. Some consumers may sign up for direct control of appliances and equipment by the utility, while others may configure their home or building area networks to automatically respond on their behalf. Users can manually turn off appliances when they receive a notification on their mobile device.

By presenting the smart grid information in a natural language, consumers may find it easier to comprehend and interpret than the structured data presented in the form of tables or graphs. Thus, the reduce activity and its sustainability newsfeed allow a unique communication between the smart grid and the consumer, personalizing requests and determining incentives that work best for different demographics and for different consumer categories such as residential, commercial, and industrial. By educating the consumers about load curtailment benefits, sustainability and energy conservation can be optimized based on the user's preferences.



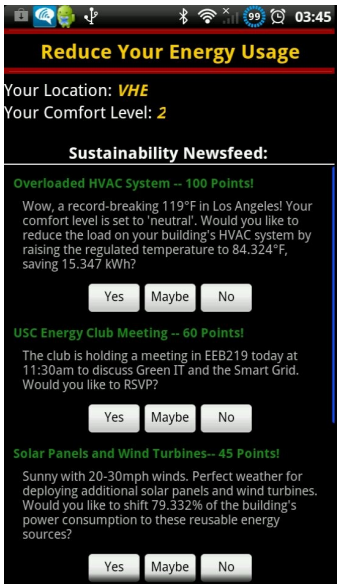


Fig. 7. Clicking a push notification opens the Reduce Activity, which contains the Sustainability Newsfeed.

V. Experimental Analysis

The USC Smart Grid depends on the scalable infrastructure of the Cloud to process and archive the prohibitively large amount of data from smart meters and grid sensors. This infrastructure is essential for timely forecast predictions and triggering responses. The flexible Cloud infrastructure also serves the mobile application requests to utilize smart grid data in an intelligent application. In mobile applications, there is a certain QoS expected for response latency. This latency will increase as the number of server requests increase, from an increase in application users. Thus, as more people use the USC Smart Grid App, the backend Cloud-hosted services will need to scale accordingly to meet performance expectations and minimize request latency.

A preliminary analysis of the request/response latency was performed on the USC Smart Grid App with an EC2 micro instance in the backend. The requests tested were "choose location", "post location", and "get forecast". Please refer to sections 4.1 and 4.2 for the complete data flow of each operation.

The first experiment focused on creating a control set of request latencies for a single user on a micro instance. In each operation, the total latency was calculated by determining the time elapsed from pressing the request button to when the response was displayed. The server latency was calculated by determining the execution time in the server scripts from request set to response echo. Fig. 8 shows the average results after 10 trials on two mobile connection speeds, 4G and WiFi. As expected, the server latency is about the same for 4G and WiFi; they differ in their network latencies, which is a reflection of the connection speed. The slower WiFi connection at an average of 7.94Mbps generated considerably slower request/response latencies than the 4G connection at 10.31Mbps (determined using Ookla's speedtest.net application). Assuming the UI latency is negligible, this means the network latency is the bottleneck.

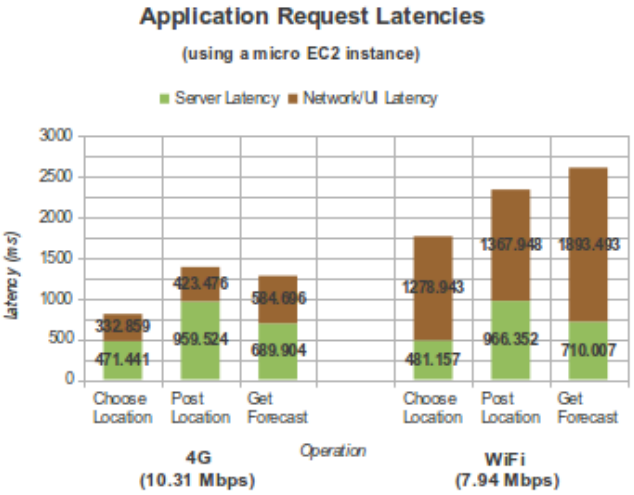


Fig. 8. Request latencies using 4G and WiFi connections.

There are two ways to reduce the total request latency: increase network performance or increase server performance. Network performance is a property of the device and the Internet service provider, so it cannot be actively reduced through the application or Cloud backend. However, the server performance can be improved by utilizing the elasticity of the Cloud—launching additional EC2 instances as users increase to service the increased request throughput.

The EC2 micro instance (t1.micro) provides a small amount of consistent CPU resources, and allows an increase in CPU capacity in short bursts when additional cycles are available. They are well-suited for lower throughput applications that require compute cycles periodically. The detailed specifications are up to 2 EC2 compute units, 613MB memory, and low I/O performance. Amazon offers several other more powerful EC2 instances such as the high-CPU extra large instance (c1.xlarge). This instance provides proportionally high CPU resources with increased network performance and are well-suited for compute-intensive applications and demanding network-bound applications. The detailed specifications are 20 EC2 compute units, 7GB memory, and high I/O

performance. This EC2 instance has 10 times the computational power of the EC2 micro instance.

A second experiment was run to compare the performance of the micro and extra-large EC2 instances in an attempt to minimize server latency by utilizing a more powerful virtual machine. As shown in Fig. 9, the execution time for the three USC Smart Grid App operations are about the same in both instances, varying at most by 8ms. In fact, the extra-large instance was slightly slower in an average of 10 trials. This means that the server execution time for a single request cannot be improved by using more powerful instances. In order to minimize server latency, the PHP code must be optimized.

Although the EC2 extra-large instance performed equally as well as the micro instance, the performance when dealing with multiple requests is yet to be explored. When the server receives multiple requests, it adds additional latency to handle the workload. In this situation, additional micro instances would need to be launched to service multiple requests. The extra-large instance is designed to handle high I/O and would probably outperform the micro instance when subject to multiple requests. The challenge is to find the optimal number of instances based on the number of requests, in order to minimize response time for all users. This number will vary depending on the type of EC2 instance launched. Perhaps a combination of instance types are required. An experiment simulating requests from hundreds to several thousand simultaneous users in would benchmark the system for high traffic. This latency information can then be used to change the behavior of the app. For example, users that are currently on campus may have preference for energy forecasting, and would therefore have priority in the request queue; the latency experienced by a user on campus would be considerably less than a user off campus.

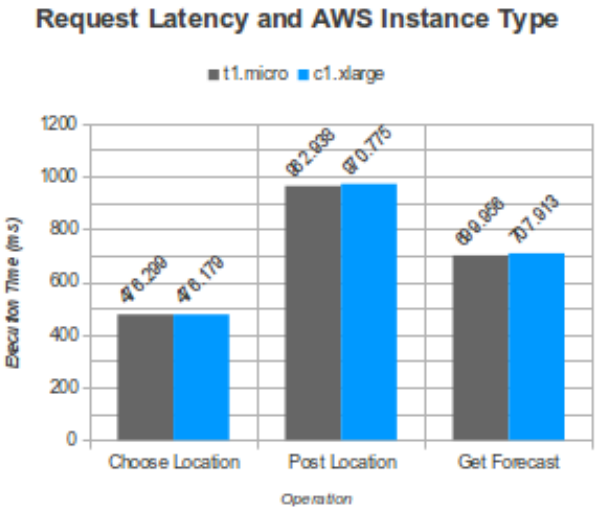


Fig. 9. Request latency based on AWS EC2 instance type.

In this experiment, the time required to launch and terminate an instance becomes important, as the requests cannot be executed until they have instances to run on. Working with demand forecasting from the Smart Grid, a demand forecast can be projected to estimate the peak usage of the app. Then, upon expected fluctuations in usage (such as 9:00am when most users start their days), the backend can appropriately launch additional EC2 instances ahead of time to make the cold startup time for new instances transparent. Each additional EC2 instance increases the cost for the backend system. Trade offs between the cost for performing accurate, but computationally costly forecast against the energy gains for the utility need to be considered.

These technical issues of synchronous and asynchronous communication, latency and responsiveness, and data-sharing and privacy need to be further explored to develop a smarter mobile backend. By leveraging the benefits of Cloud-hosted services, the USC Smart Grid App can minimize its request/response latency by launching or terminating EC2 instances according to request load. In a sense, an automated Smart load-balancing system will also need to be implemented to service the backend of the smart grid to minimize cost and maximize performance.

VI. Conclusion

In order to support the highly-efficient renewable energy sources of the future, an intelligent system must be used to regulate and distribute energy. This smart grid will ingest real-time information arriving from millions of smart meters, detect critical anomalies with low latency, annotate smart meter data with other information sources, update the demand forecast using the latest information, and respond to peak load or other events that are detected by interacting with consumers. This personalized interaction with consumers can be achieved through the use of a mobile application. In the preliminary research for a city-scale smart power grid, mobile applications allow for remote access and control of a user's energy consumption and are a convenient and natural solution for the majority of future smart grid consumers.

This project is a first attempt to develop a proof-of-concept application for interacting with the USC Smart Grid's users. It leverages Cloud-hosted services for the mobile backend to enable the computational load to scale with the number of users. Much work remains to be completed for the USC Smart Grid App, including minimizing the request latency, automating the personalized push notifications, and improving the user experience.

REFERENCES

[1] J. Bishop, "Understanding and Facilitating the Development of Social Networks in Online Dating Communities: A Case Study EndModel," http://www.jonathanbishop.com/Library/Documents/EN/docSNCEDS_Ch15.pdf. 2008

[2] Demirbas, et al, "*Crowd-Sourced Sensing and Collaboration Using Twitter,*" IEEE Int'l Symp. World of Wireless Mobile and Multimedia Networks (WoWMoM), Montreal, Canada, 14-17 June 2010. pp. 1- 9.

[3] Markus Weiss, Claire-Michelle Loock, Thorsten Staake, Friedemann Mattern, and Elgar Fleisch. "*Evaluating Mobile Phones as Energy Consumption Feedback Devices,*" Proceedings of the 7th International ICST

[4] Y. Simmhan, S. Aman, B. Cao, M. Giakkoupis, A. Kumbhare, Q. Zhou, D. Paul, C. Fern, A. Sharma and V. K. Prasanna. "*An Informatics Approach to Demand Response Optimization in Smart Grids,*" 2011, Technical Report, University of Southern California.

[5] Joseph Paradiso, Prabal Dutta, Hans, Gellersen and Eve Schooler. "*Guest Editors' Introduction: Smart Energy Systems*," IEEE Pervasive Computing, Vol. 10, No. 1, 2011.

[6] K. Hwang, G. Fox, J. Dongarra. "*Distributed and Cloud Computing From Parallel Processeing to the Internet of Things,*" 2012. Morgan Kaufmann, MA.

[7] "Advanced Universally Unique Identifiers (UUID)." H2. Web. 06 May 2012. <http://www.h2database.com/html/advanced.html>.

[8] "Amazon EC2 Instance Types." Amazon. Web. 06 May 2012. <http://aws.amazon.com/ec2/instance-types/>.

[10] S. Aman, Y. Simmhan, and V. Prasanna, "*Smart communication of energy use and prediction in a smart grid software architecture*," in in IEEE Workshop on Green Energy Production and Adaptive Power Distribution, 2010.

[11] Liu, Yi. "A PHP Interface to GNUPlot." Web. 06 May 2012. <http://php-gnuplot.sourceforge.net/>.

[12] "The Developer's Guide." *Android Developers*. Google. Web. 06 May 2012. <http://developer.android.com/guide/index.html>.

[13] "Point-in-polygon Algorythm." AssemblySys. Web. 06 May 2012. . "Point-in-polygon Algorythm." AssemblySys. Web. 06 May 2012. <http://www.assemblysys.com/dataServices/php_pointinpolygon.php>.

[14] "Push Notifications: Adding Push To Your App." *Android Developer's Guide | Parse*. Parse. Web. 06 May 2012.<https://parse.com/docs/android_guide#push