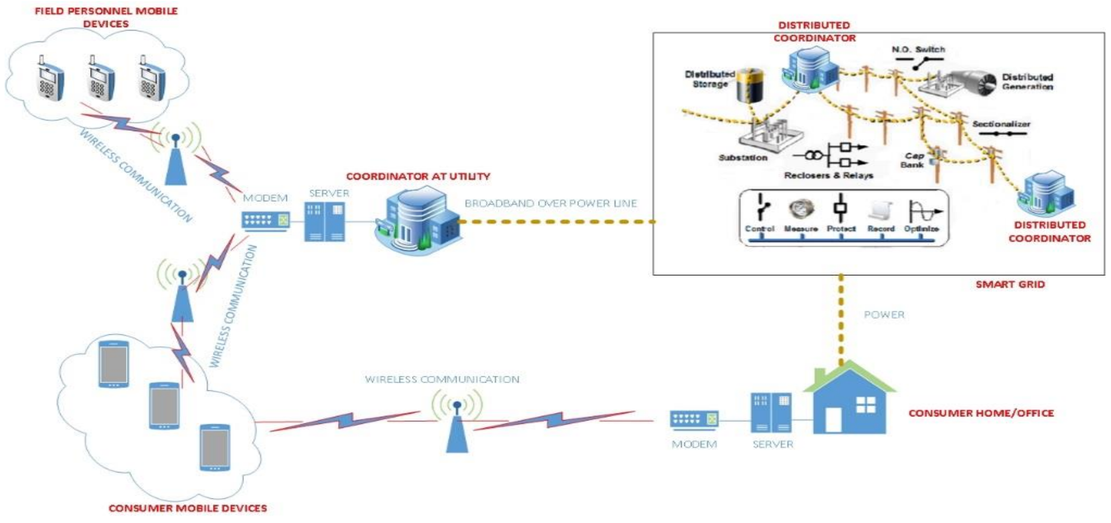
Final Project

Mobile Devices and Big Data (Fall 2014)

1. Overview:



The Requirements fulfilled in the project are as follows:

- Developed a mobile application which is user friendly and simple to use
- Established a connection between the Home Server and the Mobile App
- Established a connection between Mobile App and Utility server
- Home server uses simulated annealing to schedule the appliances
- Utility server calculates the power profile and ensures the peak power is not exceeded
- A RPS system is assumed to exist
- The Utility server also uses simulated annealing to re-schedule

2. Components of the Project:

The project consists of 3 main parts:

- Mobile App
- Home Server
- RPS
- Utility Server

Mobile App:

The App has the following features:

- It has a Login Screen which allows existing users to login as well as new users to sign up. This is the main activity of the android app.
- We have used Parse.com to maintain a database for each user. The login authentication has also been implemented use the Parse.com API
- On successful login the user is taken to the Home Activity
- The Home Activity allows the user to Add new appliances , View existing Appliances, Schedule the Appliances as well as Delete and Edit the Appliances
- The Manage Appliances takes the input from the user i.e Appliance Name, Start Time, Stop Time, Run Time, Constraints , etc
- The data entered by the user is saved on the database as well as posted to the php script.
- The schedule provided by the Home Server is displayed on the app and the app sends the schedule to the Utility server.
- The feedback from the Utility server and the decision to either commit or reschedule is also provided by the App
- The mobile App serves as the interconnection between the Home Server and the Utility server. All communications are bi-directional and always through the Mobile App.

Home Server:

- The Home server receives all the information entered by the user.
- It performs simulated annealing to schedule all the appliances
- Appliances with soft constraint are scheduled in a manner most profitable to the user.
- The Home server posts the schedule to the mobile app , thus the user will be able to see the schedule provided by the Home server.
- The App then sends the schedule received to the Utility server
- The home server keeps a track of the user, his list of appliances as well as the schedule.

RPS:

- The RPS is assumed to be available all day long. The power availability is different for every hour. The hourly value was generated randomly using rand() function.
- For every hour , the power requirement is obtained using the formula:
  Power required = Power to run the appliances – Power from RPS
- If during any hour there are no appliances running and the renewable power source is available then this power is sold to the Duke server.

Utility Server:

- The utility server receives the schedule as provided by the Home Server
- It runs Hadoop to check the power profiling
- It uses a map reduce function to calculate the power profile
- It runs a test to check the power profiling and if at any hour the consumption is greater the peak value then it calculates the excess cost that the user will have to incur.
- After running the test, the Utility server posts to the mobile app to indicate if the schedule was successfully accepted or not.
- If the schedule was not accepted due to violation of the peak profile then the Utility server posts to the mobile app the excess cost the user will have to incur if he wants to stick to the schedule provided by his home server.
- It also gives the user the option to reschedule.
- If the user agrees to reschedule then the Utility server will schedule the appliances and send the user the updated schedule.
- If the user decides to retain the schedule provided by his home server then the user COMMITS to the Utility server indicating that he has agreed to pay the additional expenses.

Thus in this way the end to end communication takes place between the Mobile App , Home Server, RPS and Utility Server

3. Algorithm Used:

The algorithm uses a flavor of simulated annealing where the temperature is kept fixed. A list of all tasks is iterated through and provided with an initial schedule. The total power and price for this schedule is calculated and added to a list which maintains the configuration history. The total price is compared with a minimum price value, and If it is lesser, then the minimum price is updated to hold the total price. This loop continues until the system's stability is maintained. The stability is computed by assigning components with 'Soft Constraints' a particular value. This value is used to see how an appliance with SC can be allowed to continue after the end time for it.

After it breaks out of the loop, the history list is used to find out which configuration produces near to minimum price. And that schedule is sent to the user.

4. Results and Analysis:

The entire architecture was set-up and demonstrated successfully to the Instructor and TA.

5. Scope for Future Work:

- The home server can be made to run Hadoop as well
- Rescheduling on the Utility server can be made to run Hadoop
- The system can be tested for 1000 users or more
- Commit requests from more than one user at a time can be processed using a FIFO queue concept

6. Group Members:

Akshay Pandian – apandia2@uncc.edu
Swathi B Ayas - sayas@uncc.edu
Nikunj Tonthanahal – ntonthan@uncc.edu