A Smart Power Outlet for Electric Devices that can Benefit from Real-Time Pricing

V.P. Ramavarapu*, R. Sowers†, and R.S. Sreenivas‡
* Centennial High School, Champaign, IL 61821 (Summer Intern, RIoT Lab, UIUC)
† Co-Director, *Rapid IoT (RIoT) Laboratory*, ISE & Mathematics
‡ *Senior Member, IEEE,* Co-Director, *Rapid IoT (RIoT) Laboratory*, ISE, CSL & ITI
University of Illinois at Urbana-Champaign, Urbana, IL 61820.

Abstract—**Under *Real-Time Pricing* (RTP) the cost per kWh varies with time as a consequence of the changing supply and demand conditions in the market. Customers that are able to shift their electricity consumption patterns can reduce their costs by using their devices when price is low. This requires a smart power outlet that can turn itself on/off based on price, and customer preferences.**

This paper is about a smart power outlet that uses a *Belkin WeMo Mini Smart Plug* (WeMo) [1], which can be autonomously turned on/off using an *Intel Edison* (Edison) [2] through a WiFi connection. The JavaScript code, which runs on the Intel Edison, currently turns on (resp. off) the electric device that is connected to the WeMo if the current price is negative (resp. positive). This code can be modified to incorporate other criteria like customer preferences if necessary. Our ongoing work involves the design and validation of a suite of switching-algorithms for this platform.

I. Introduction

Tesla's *Charge-at-Home* [3] recommends an hour of charge every 29 miles of range. It is estimated that a Tesla user would require about four hours of charge daily, which is usually done at night at the customer's home. It is important to note that the charging-times do not have to be contiguous. The charging cost to the user would be determined by the price offered by the electricity provider during the charging-periods.

The electricity provider uses the supply and demand of the market, along with other considerations, to set the price. Traditionally, this used to be a fixed/flat price. Nowadays, providers offer a dynamic price to the consumer, who can make informed energy choices that are tailored to the individual. This promotes smarter energy use, which can improve efficiency, lower costs, and has a positive impact on the environment.

Commonwealth Edison (*ComEd*) is the largest electric utility in Illinois, serving the Chicago and Northern Illinois area. ComEd offers a 5-minute price (cf. https://hourlypricing.comed.com/live-prices/five-minute-prices/), and an hourly-price (cf. https://hourlypricing.comed.com/live-prices/), which is the real-time average of the current hour's twelve 5-minute prices. Figure 1 shows ComEd's *Real-Time Hourly Price* (RTHP) on May 14th, 2017. On this date, there was a period of 6.5 hours where the RTHP was negative. A Tesla user, whose charging adapter was connected to a smart outlet that turned itself only during periods when the price was negative, would have charged his vehicle for minimal cost on this day. There might be other days where a full-complement of six hours of negative price might not be realized. Under this circumstance, the user would need a charging-strategy that ensures a full-charge at minimal cost. Our current research is focused on the design and back testing of these strategies. Before any of these strategies can be implemented in practice, it is critical to have a smart electric outlet that can be remotely turned on/off based as per the direction of the implemented strategy. Stated differently, an end-user device that decides the appropriate times to turn itself "on" or "off" based on price is the mandatory first-step before any high-granularity, smart-energy management methods (eg. [4], [5]) can be implemented in practice.

This is essentially a device in a customer's premise that can make intelligent decisions on when to use power. There

This paper is about the construction of such a smart outlet using an *Intel Edison* (Edison) [2],

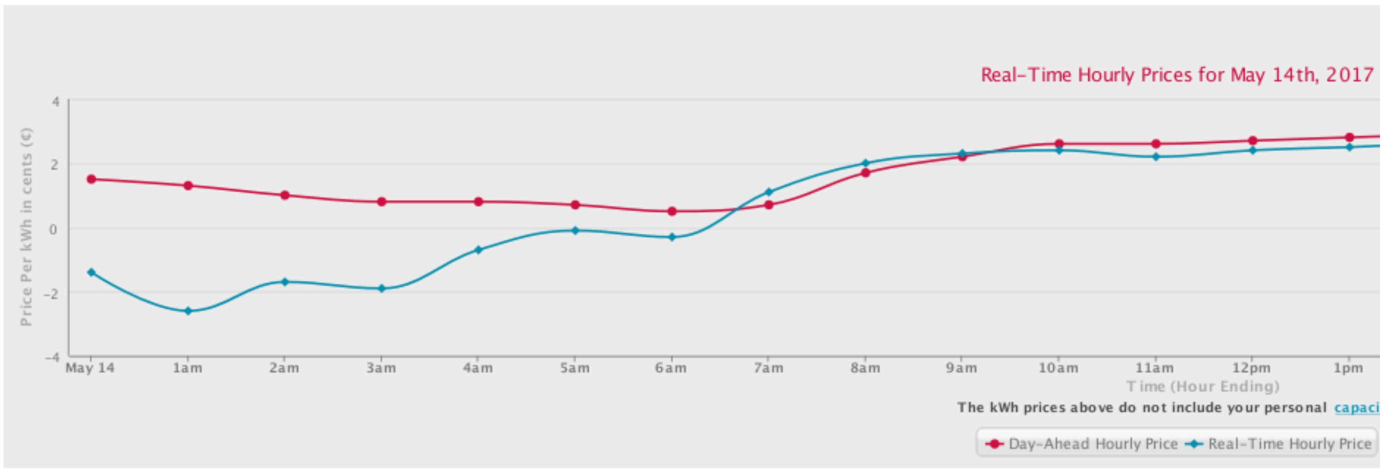


Fig. 1. ComEd's *Real-Time Hourly Prices* (RTHP) for May 14th, 2017 (cf. https://hourlypricing.comed.com/live-prices/). The RTHP (shown in blue) was negative from Midnight to 6:30AM. This plot has been clipped to highlight the RTHP-trajectory during the early hours of May 14th, 2017. The entire RTHP-trajectory for this date can be found here – https://hourlypricing.comed.com/live-prices/?date=20170514.

Belkin WeMo Mini Smart Plug (WeMo) [1], and some (easily extended) JavaScript programming. The example of charging an electric vehicle was used for illustrative purposes. This smart electric outlet can be used in conjunction with any electric device that benefit from the real-time price offered by a provider.

The rest of the paper is organized as follows. In the next section we present a brief review of real-time pricing and its influence in the power-market. Section III describes the smart power outlet, and presents a verification of its functionality. The paper concludes with a review of our ongoing work.

II. A Review of Real-Time Pricing in the Power Market

The *U.S. Energy Information Administration* (USEIA) reports that in 2016, the annual average price of electricity in the United States was 10.28c per kilowatt-hour (kWh). Residential customers paid 12.55c per kWh; commercial customers paid 10.37c per kWh; Industrial customers paid 6.75c per kWh; and Transportation customers paid 9.48c per kWh, respectively [6].

Dynamic pricing offers a price that depends on when electricity is used. The customer can consume energy when it is cheaper, and chose to conserve energy when the price is expensive. This results in better utilization of generated-power, lower-costs to the customer, and is said to have a positive impact on the environment. It was thought that while dynamic pricing will promote the judicious shifting of energy use, it would not reduce the total energy use in any way. However, the data shown in Table 3 of reference [7] shows that dynamic pricing has resulted in a 4% annual reduction in kWh usage per household.

Section 3.4 of reference [7] notes that in 2010 ComEd's *Residential Real-Time Pricing* had generated bill savings of $3,954,882, and the average 2010 bill savings among customers who were in the RRTP program for all of 2010 was $177.

The above mentioned reference also notes that the RRTP program had some unintended benefits as well, in that the RRTP program reduced SO_2, NO_x and CO_2 emissions, which is estimated to be about $185,000 per year. That is, dynamic pricing is environmentally friendly, as well.

ComEd's hourly price [8] is the average of twelve 5-minute prices for that hour, and the real-time hourly price is only known after the hour has passed (cf. figure 2). The current hour average price, which is posted on the web, is the real-time average of the current hour's 5-minute prices (cf. figure 3).

It is possible for the price of electricity to be negative for short periods of time. This happens whenever the supply of electricity exceeds the demand. When demand is short of supply, due to physical-considerations, many electricity generators cannot reduce output to match demand. If the price goes negative for a period, the consumer is paid to consume electricity. A nominal delivery charge might apply in some cases. As shown in figure 1, the real-time price was negative for 6.5 hours on May 14, 2017.

ComEd provides an API that provides the Current-Hour 5-Minute Price via an API [9], which can provide its output in JSON- [10], RSS- [11], or text-format. The default output format is JSON.

For the remainder of this paper we will work under the assumption that the strategy of consuming power (for the present 5-minute period) only when the *Current-Hour 5-Minute Price* (cf. figure 3) is negative is desirable. The rest of the paper is about the construction of such a device using an Intel Edison and a WeMo switch.

III. The Smart Power Outlet

The *Intel Edison* (cf. figure 7) is a 22 nm Intel *System on Chip* (SoC) that includes a 500 MHz dual core, dual threaded Intel Atom CPU with 4 GB of flash memory with pre-installed *Yocto Linux* [12]. It also has a 100 MHz 32-bit Intel Quark micro-controller, and a Broadcom BCM43340 chip that provides standard dual-band 2.4 GHz and 5 GHz IEEE 802.11 a/b/h/n connectivity. The WiFi connection can be protected using WPA and WPA2 authentication. The Edison can connect via Bluetooth to other devices like smart-phones. It support 40 *General-Purpose I/O* (GPIO) connections. It is can be programmed using Intel's Integrated Development Environment (IDE) – Intel XDK – which can be used develop JavaScript-based Node.js [13] applications for the Edison.

The *Belkin WeMo Mini Smart Plug* (WeMo) (cf. figure 5) is a switched-outlet that can be controlled via IP networks. It comes with an iOS/Android App that can turn on/off the plug (and any device that is connected to it) remotely.

The wemo-client is a low-level client library that can be used to control WeMo devices within Node.js applications. The details of the wemo-client API can be found in reference [14]. It can be installed on the Edison using the npm install wemo-client command via the SSH Terminal on the Intel XDK (cf. figure 6).

We now present annotative description of the JavaScript code shown in figures 8 and 9, which effectively turns on (resp. off) the WeMo when the Current-hour-5-Minute-Price is negative (resp. positive). Before this code is run on the Edison, it is imperative that its clock is synchronized to local time. This was done by removing the existing /etc/localtime file on the Edison, followed by symbolically linking /etc/localtime to the appropriate timezone file in /usr/share/zoneinfo/America/. For our Edison, this was done by the ln -s /usr/share/zoneinfo/America/Chicago /etc/localtime command. The WeMo is assumed to be in the "off" position initially.

The function periodicActivity() is called every five minutes (starting as close as possible to the hour). This is done by the setTimeout(periodicActivity, 300000); command at the end of the code shown in figure 8.

The http.get() command obtains the JSON-formatted output that would result soon after ComEd's API[1] is executed. The command var priceResp = JSON.parse(body); stores the JSON-formatted response from the API. Consequently, each element of priceResp has two fields – "millisUTC," which is the UNIX-time [15]; and "price," which is the price in cents per kWh at the time indicated in the first-field. The price-field of the first-element of priceResp (i.e. priceResp[0].price) is the latest price. If this is is negative, the WeMo has to be turned on (if it is currently off).

Since WeMo devices can be temporarily lost – we include the WeMo device-discovery process within the periodicActivity() function. There were other WeMo's in the premises where this code was tested, and the specific WeMo we wish to control using the current-price had the name Wemo Mini 4. The wemo-client library provides a setBinaryState(value) function, where value = 1 (resp. value = 0) turns the device on (resp. off). If the WeMo device was previously off, and the price went negative, it is turned on. Likewise, if the device was previously on, and the price went positive, the device is turned off. This device was tested between 2:00PM and 3:30PM on May 21, 2017. Figure 7 shows the RTHP-trajectory for that day. The device turned "on" between 2:10PM and 3:10PM when the price was negative. At 3:10PM the price went positive, and the device turned "off" for 5 minutes. Following this, at 3:15PM the price went negative, and the device turned "on," and stayed on till 3:30PM when the test was concluded.

[1] http://hourlypricing.comed.com/api?type=5minutefeed

IV. Conclusion

This paper describes a smart electric outlet that can be turned on (resp. off) when the current (real-time) price of electricity is negative. The hardware for this device consisted of an *Intel Edison* and a *Belkin WeMo Mini Smart Plug*. The proposed functionality was achieved through the Node.js JavaScript application that runs on the Edison. This software can be extended to include sophisticated switching-strategies that depend on the price-history and user-preference. As an illustration, a Tesla user has to charge his vehicle for six hours each day. This decision is not difficult on a day like May 14, 2017 (cf. figure 1). There will be days when the price does not stay negative for a sufficient amount of time. This calls for the development of alternate strategies that accomplishes the consumption-goal at minimal cost. This is a part of of our ongoing work.

References

[1] http://www.belkin.com/us/Products/home-automation/c/wemo-home-automation/, Last Accessed: May 23, 2017.

[2] https://software.intel.com/en-us/articles/what-is-the-intel-edison-module, Last Accessed: May 23, 2017.

[3] https://www.tesla.com/charge-at-home, Last Accessed: May 23, 2017.

[4] R. Ahmad, A. Gani, S. Hamid, M. Shojafar, A. Ahmed, S. Madani, K. Saleem, and J. Rodrigues, "A survey on energy estimation and power modeling schemes for smartphone applications," *Int J Commun Syst*, vol. 30, 2017, https://doi.org/10.1002/dac.3234.

[5] Z. Pooranian, N. Nikmehr, S. Najafi-Ravadanegh, H. Mahdin, and J. Abawajy, "Economical and environmental operation of smart networked microgrids under uncertainties using nsga-ii," in *2016 24th International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*, Split, Croatia, September 2016, pp. 35–40.

[6] https://www.eia.gov/energyexplained/index.cfm?page=electricity_factors_affecting_prices, Last Accessed: May 23, 2017.

[7] Navigant Consulting Inc., "Evaluation of the residential real time pricing program, 2007-2010," June 2011, prepared for the Commonwealth Edison Company, http://www.icc.illinois.gov/downloads/public/edocket/296671.pdf & http://www.icc.illinois.gov/downloads/public/edocket/296673.pdf.

[8] https://hourlypricing.comed.com/live-prices/, Last Accessed: May 23, 2017.

[9] https://hourlypricing.comed.com/api?type=5minutefeed, Last Accessed: May 23, 2017.

[10] https://www.w3schools.com/js/js_json_intro.asp, Last Accessed: May 23, 2017.

[11] https://validator.w3.org/feed/docs/rss2.html, Last Accessed: May 23, 2017.

[12] https://www.yoctoproject.org/about, Last Accessed: May 23, 2017.

[13] https://en.wikipedia.org/wiki/Node.js, Last Accessed: May 23, 2017.

[14] https://www.npmjs.com/package/wemo-client, Last Accessed: May 23, 2017.

[15] https://en.wikipedia.org/wiki/Unix_time, Last Accessed: May 23, 2017.

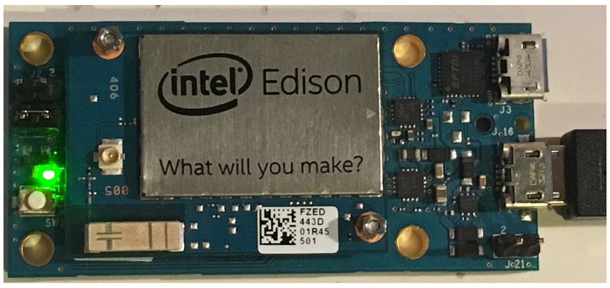


Fig. 4. The Intel Edison [2] with a Mini Breakout Board from Adafruit.com.

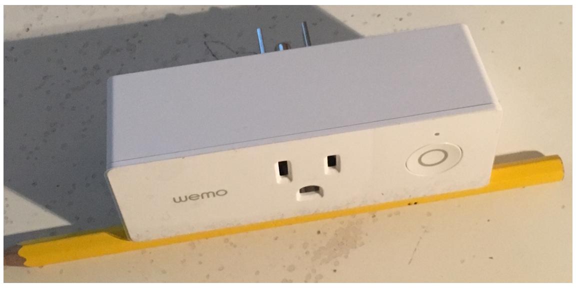
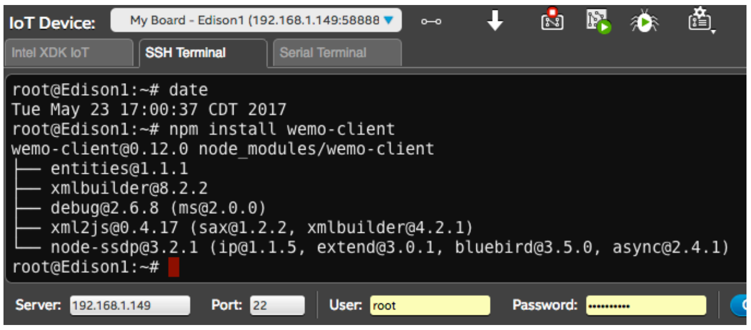


Fig. 5. The *Belkin WeMo Mini Smart Plug* (WeMo).



```
root@Edison1:~# date
Tue May 23 17:00:37 CDT 2017
root@Edison1:~# npm install wemo-client
wemo-client@0.12.0 node_modules/wemo-client
├── entities@1.1.1
├── xmlbuilder@8.2.2
├── debug@2.6.8 (ms@2.0.0)
├── xml2js@0.4.17 (sax@1.2.2, xmlbuilder@4.2.1)
└── node-ssdp@3.2.1 (ip@1.1.5, extend@3.0.1, bluebird@3.5.0, async@2.4.1)
root@Edison1:~#
```

Server: 192.168.1.149 Port: 22 User: root Password: ••••••••••

Fig. 6. Installing the wemo-client on the Edison via the SSH Terminal on the Intel XDK.

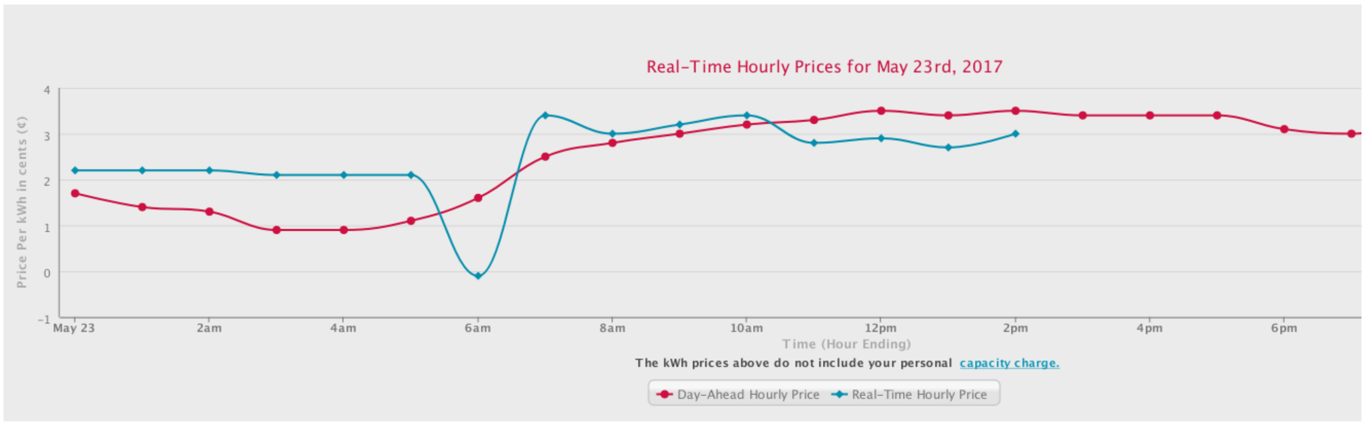


Fig. 2. ComEd's *Real-Time Hourly Prices* (RTHP) for May 23, 2017. The price at 2:00PM was 3c per kWh.

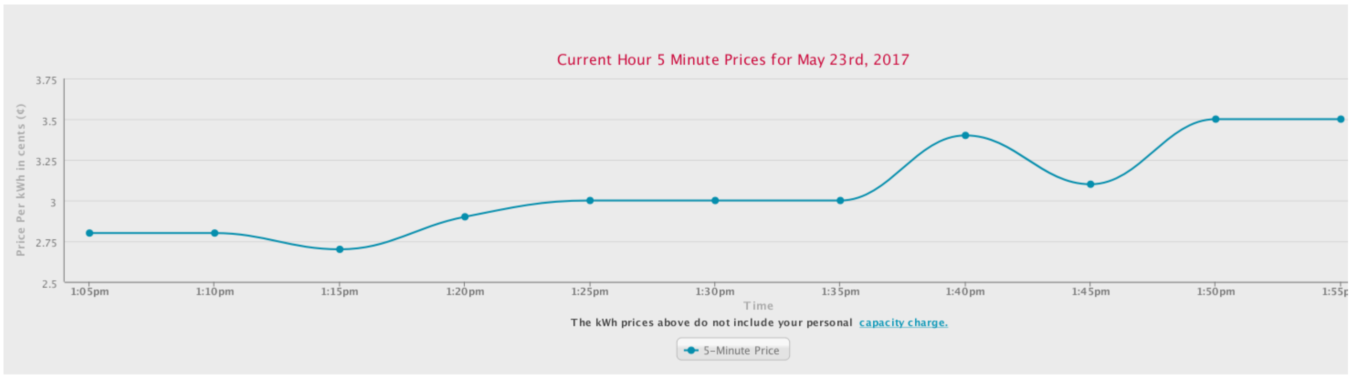


Fig. 3. ComEd's *Current-Hour 5-Minute Price* from 1:05PM to 1:55PM on May 23, 2107. The average of these eleven values was 3.06c per kWh.

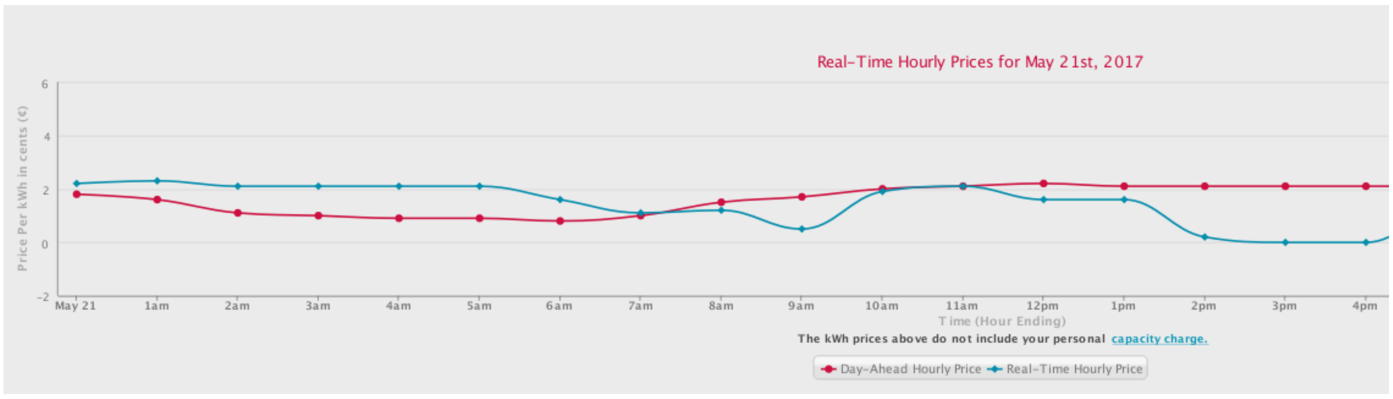


Fig. 7. ComEd's RTHP for May 21, 2017, which is close to the vicinity of 0c per kWh between 2:00 and 4:00PM.

```
// V.P. Ramavarapu, R. Sowers and R.S. Sreenivas, ”A Smart Power
// Outlet for Electric Devices that can Benefit from Real−Time
// Pricing,” In Proceedings of the 3rd International Conference
// on Control, Electronics, Renewable Energy, and Communications
// (ICCEREC 2017), September 2017, Yogyakarta, Indonesia.
//
// Intel−Edison’s JavaScript Code that turns−on (resp. turns−off)
// the electric device connected to the WeMo when the current
// real−time price is negative (resp. positive)
//
var http = require(’http’);
var Wemo = require(’wemo−client’);
// https://www.npmjs.com/package/wemo−client
// Global Variables about switch state & Price
var priceIsNegative = false;
var switchState = false;

// Call the periodicActivity function
periodicActivity();

// This function is called forever (due to the setTimeout() function)
function periodicActivity() {
var getReq = ”http://hourlypricing.comed.com/api?type=5minutefeed”;
// Make the request
var request = http.get(getReq, function(response) {
// Where we store the response text
var body = ’’;

//Read the data
response.on(’data’, function(chunk) {body += chunk;});
// Print out the data once we have received all of it
response.on(’end’, function() {
    if (response.statusCode === 200) {
        try {
            // Parse the JSON to get the pieces we need
            var priceResp = JSON.parse(body);
            console.log(’Spot Price is = %d’, priceResp[0].price);
            priceIsNegative = (priceResp[0].price <= 0) ? true : false;
        } catch(error) {
            // Report problem with parsing the JSON
            console.log(”Parsing error: ” + error);
        }
    } else {
        // Report problem with the response
        console.log(”Response error: ” +
                    http.STATUS_CODES[response.statusCode]);
    }
})
// To be continued in Fig. 9..
```

Fig. 8. First-part of the Node.js application that controls the WeMo using the Edison. This code is continued in figure 9.

```
// Continuing JavaScript Code listing from Fig. 8
var wemo = new Wemo();
    wemo.discover(function(deviceInfo) {
        if (deviceInfo.friendlyName == 'Wemo Mini 4')
        {
            console.log('Wemo Device Found: %j',deviceInfo.friendlyName);
            // Get the client for the found device
            var client = wemo.client(deviceInfo);
            // We want to listen to error events (e.g. device offline),
            // Node will throw them as an exception if left unhandled
            client.on('error', function(err) {
                console.log('Error: %s', err.code);
            });
            // Handle BinaryState events
            client.on('binaryState', function(value) {
                if (deviceInfo.friendlyName == 'Wemo Mini 4')
                    switchState = (value === '1');
                console.log('Device %j turned %s',deviceInfo.friendlyName,
                            value === '1' ? 'on' : 'off')
            });
            if (deviceInfo.friendlyName == 'Wemo Mini 4')
            {
                console.log('Price is negative?', priceIsNegative);
                console.log('Switch State?', switchState);
                if ((priceIsNegative) && (!switchState))
                {
                    client.setBinaryState(1);
                    switchState = true;
                }
                if ((!priceIsNegative) && (switchState))
                {
                    client.setBinaryState(0);
                    switchState = false;
                }
            }
        }
    });
});

    // Wait for 5 mins and call this function again
    setTimeout(periodicActivity, 300000);
    var timeNow = new Date;
    console.log('polling for prices at %j', timeNow);
}
```

Fig. 9. Second-part of the Node.js application that controls the WeMo using the Edison. This code is a continuation of figure 8.