Free market blockchain P2P energy trading

DESIGN DOCUMENT

Team #41

Arun Sondhi: Hardware and API Development

Alec Dorenkamp: Smart Contract and Blockchain Design

Noah Eigenfeld: Smart Contract and Blockchain Design

Brendon Geils: Blockchain and API Development

Jack Myers: Smart Meter Circuit Design

Joe Staudacher: Power Sensing and Hardware

Client: Sodima Solutions

Adviser: Goce Trajcevski

sdmay18-41@iastate.edu

sdmay18-41.sd.ece.iastate.edu

Revised: 10/15/17

Table of Contents

List of figures/tables/symbols/definitions

1 Introduction

1.1 Acknowledgement

To preface our design document for this project, we would like to thank those who have made contributions to the project, outside of the members of our team. First and foremost, we would like to thank our adviser, Dr. Goce Trajcevski, for his technical assistance and comprehensive guidance throughout the project. We would also like to thank Sodima Solutions for providing the funding for the required hardware for the project, as well as providing the overall idea of the project. Lastly, we would like to express our gratitude towards the faculty of Iowa State University for their support in giving us the technical background and knowledge for us to handle a project of this scale. Without the support of these individuals and organizations, our project's success would not have been possible.

1.2 Problem and Project Statement

On a large scale, our project is targeting one of the most significant problems that is affecting the world right now: climate change. Renewable energy usage is seen as one of the key ways to target this problem, but a large portion of worldwide energy does not come from this "clean" energy generation. To change this, the generation of renewable energy needs to be more accessible to individuals and businesses, rather than only those who have the resources and capabilities of a large energy company.

Our solution to this problem is to incentivize renewable energy generation from individuals and small businesses by facilitating peer to peer trading of surplus energy. By creating this free market environment for energy trading, individuals will think more about how they produce and consume energy. and be inclined to generate energy of their own. With this new understanding and market accessibility, energy prices will fluctuate to be at parity with their true value, not just what the utility company dictates. A more detailed description of our implementation of this free market solution is found later in this document.

On top of incentivizing clean renewable energy, another goal that our project aims to reach is aiding in the decentralization of the power generation market. The interconnectedness of grids has already contributed to the reduction of blackouts, as one individual power plant or utility company is not solely responsible for all energy generation. As things currently stand, a Chicago grid can pull from a Toronto plant if they approach their capacity curve. Our system would aid this interconnectedness a degree further, as the power loss and cost to move energy a mile up the road would be less than the power loss and cost to move that energy from Toronto to Chicago. A more decentralized grid would help the overall grid to be robust to fluctuations, as the sources of energy would be widespread and independent.

The connection between decentralizing energy generation and our solution is clear. We hope that an open market will allow individuals to feasibly operate their own renewable energy sources like solar panels or wind turbines. If this goal is realized, energy will not be produced only at large plants. Because of this, consumers will be able to rely on a more diverse array of energy sources than those that are currently available.

Our project has two major components: the development of an inexpensive and user friendly smart power meter and software to facilitate peer to peer trading of surplus energy. When these two stages of the project are complete, individuals using our hardware and software will be able to buy or sell surplus energy at significantly better rates than could be obtained going through a utility company.

A stretch goal for the project would be to develop a marketplace where these peer to peer interactions could be facilitated. We would have to find a way to automate this process so users would simply have to say how much power they need at what times and the system would arrange for that transaction to take place.

1.3 operational Environment

The operating environment for our solution will certainly be a relevant factor in the final implementation. The meaning of "operating environment" is quite different for the hardware and software aspects of our project. For the hardware component the operating environment is taken in a literal sense. Essentially, our smart power meter has to be able to withstand all of the conditions that existing power meters currently withstand. It will have to be able to survive in various weather conditions that naturally come with being a product that lives outside of the home.

On the software side, "operational environment" has a less literal and physical meaning. Instead, the operating environment of our software will be economic and political climate in which our solution is being used. There are many legal factors that could come into play with this kind of trading, like use of the utility company's infrastructure or trading between different cities, states, countries. These are all factors that would need to be explored in detail if this project were to be expanded beyond a simple test environment with two nearby homes or small businesses.

As the project continues (potentially with future senior design groups), more focus can be put into refining the robustness of the hardware and making sure our software implementation integrates effectively with the economy and politics of the location where our solution is being used.

1.4 Intended Users and uses

The users can be split into two groups, which we'll call producers and consumers. The producers are the users who will be supplying excess energy that they produce into the system. Producers are looking to maximize the profit that they can create from producing energy, and our tradable energy market will enable them to do just that. In order to best serve these users, we are minimizing the transaction costs and maximizing the ease with which they can find buyers for their energy. The consumers are the users who will be consuming the excess energy that producers create. We can best serve them by minimizing the transaction costs and making it as easy as possible for them to find producers whose energy they can consume.

1.5 Assumptions and Limitations

– Two separate lists, with a short justification as needed.

– Extremely important, as it can be one of the primary places where the client can go to determine if the end product will meet their needs.

– Examples of assumptions: The maximum number of simultaneous users/customers will be ten; Blue is the best background color and will be used; The end product will not be used outside the United States.

– Example of limitations: The end product shall be no larger than 5"x8"x3" (client requirement); The cost to produce the end product shall not exceed one hundred dollars (a market survey result); The system must operate at 120 or 220 volts and 50 or 60 Hertz (the most common household voltages worldwide).

– For limitations, include tests not performed, classes of users not included, budget/schedule limitations, geographical constraints, etc.

Assumptions:

- Enacting the distribution of power after our transaction is completed is outside of the scope of our project, including the new power equipment that could be required for this distribution
- For a full implementation of our project, an agreement will have to be completed with the utility company owning the power infrastructure so they will allow these transactions to take place
- The level of testing that we will complete will be within an individual municipality, so interstate/international trading laws will not be applicable

Limitations:

- The cost of the IoT smart power meter must not exceed that of the average power meter used in Ames, IA (the area of testing)

1.6 Expected End Product and Deliverables

Blockchain Implementation

Power transactions will be made and recorded using a blockchain implemented on Ethereum's platform. This blockchain will be the backbone of our project, as it will keep a ledger of all power transactions that occur in a network of users. The decentralized-ledger approach of a blockchain will ensure all transactions are trustworthy, even though users may remain completely anonymous.

The blockchain will consist of a set of smart contracts in Ethereum defining our own cryptocurrency for power transactions, and logic to initiate and accept transactions.

IOT Smart Meter

An "internet of things" smart meter will need to be installed at a user's property to read the flow of energy into their home/building. This smart meter will be connected to the internet, and will interact with the blockchain to physically enact transactions. The smart meter will be able to verify transactions of power over set periods of time, as determined by the agreement between the buyer and seller in the transaction. The goal for the end of this semester is to have functioning basic communication between the smart meter and the software side of our project. By the end of the year, we will develop a working prototype that is able to verify the completion of a transaction, connect to the blockchain network and web application, and display vital information to the user via a user interface directly on the physical meter.

Web Application

Users will manage their power transactions through a simple web application, which will interact with our blockchain. Since the focus of our project is on the blockchain and smart meter implementations, this application will provide only the basic functionality needed for users to track and trade their power. Such functions include login/account creation capability, viewing transaction history, and initiating/accepting transactions. In the future, this web application may be expanded to better meet the needs of the customers.

2. Specifications and Analysis

We have researched various ways to go about solving our problem, the most significant of which are detailed below.

Software

1. Ethereum Approach

Using the open-source ethereum (blockchain) platform to develop our smart contract for buying and selling assets (energy). Using ethereum we can utilize their stable Solidity language to implement our smart contract. The advantage here is there are many projects built with this stack allowing for more resources and support.

2. Hyperledger Approach

Using the open-source hyperledger (blockchain) platform which is newer compared to the tried and true ethereum approach. Hyperledger is supported by larger organizations such as IBM. We believe this will allow the technology to stabilize long term with the backing of a large company compared to the burn-out many open-source projects that lack an organization have seen.

Hardware

1. Arduino + Wifi Shield Approach

Using GPIO to continuously calculate available energy and control the power management system. The data will then be used for buying and selling. A wifi enabled shield will provide IoT capabilities to support transactions.

The arduino will be linked to a two-way meter and work as a smart meter. Current and power usage will be tracked and data will be transmitted to the marketplace.

2. Microcontroller/custom PCB Approach

Functionally identical to the arduino approach, only the built from a different set of components. Both will record and send data for buying and selling.

Requirements/standards:

Project development will follow a structured model for Git version control. We will use pull requests that'll require at least one other member's approval. Each task will have it's own branch and those branches will be appropriately named based on their task. All commits will have will have sufficiently descriptive comments. All code will include descriptive objects and class names. Each method and class will be sufficiently commented except obvious getters and setters.

2.1 Proposed Design

Functional requirements:

1. A blockchain software implementation
2. An IoT Smart Meter device
3. Web app for management of transactions
4. API for communication between the smart meter and the blockchain

The requirements will grow and change as progress is made but these requirements are the core of the project and will dictate our goals over the coming months.

Non-functional requirements:

The project must meet certain standards that we as a group agree on. We want the platform for the blockchain to be robust and have the ability to expand on it if desired. This robustness will lead to trust in our energy trading system, and the scalability will enable growth. Code conciseness goes hand in hand with robustness, and makes testing and expanding our project simpler and quicker. Portability is an important requirement for the smart meter, allowing it to be installed easily and effectively, and portability for the web app is important so that users can access it from multiple different devices.(such as laptops, tablets, and mobile devices) Our goal is to end up with a product that we can be proud of and have it be something that we are eager to present to the client. Each group member will strive to put their best work forward and each contribute their own unique values to the project.

Blockchain Implementation

Our blockchain implementation will consist of a collection of Ethereum smart contracts. We will base our cryptocurrency on the ERC20 token standard, which defines the basic functions that any Ethereum cryptocurrency must implement. The units of our cryptocurrency will be some rate of power multiplied by the amount of time that power will be transferred. Because running transactions through Ethereum which change the state of the blockchain take gas (a unit that determines how much work a block takes to mine), we will break the functionality of our blockchain into several contracts, instead of combining them into one large contract. This will ensure that the overall gas cost is lower, as only the necessary tasks will be run for each transaction.

Web Application

Our web application will interact with our blockchain via an interface of Javascript methods. These methods will trigger functions defined in our blockchain smart contracts to make transactions and read changes from the blockchain. The web application will allow a user to sign in using their Ethereum address and password. It will have a home screen from which the user can view any pending transactions (which they may approve or decline), and links to initiate a new transaction or to view the ledger's transaction history.

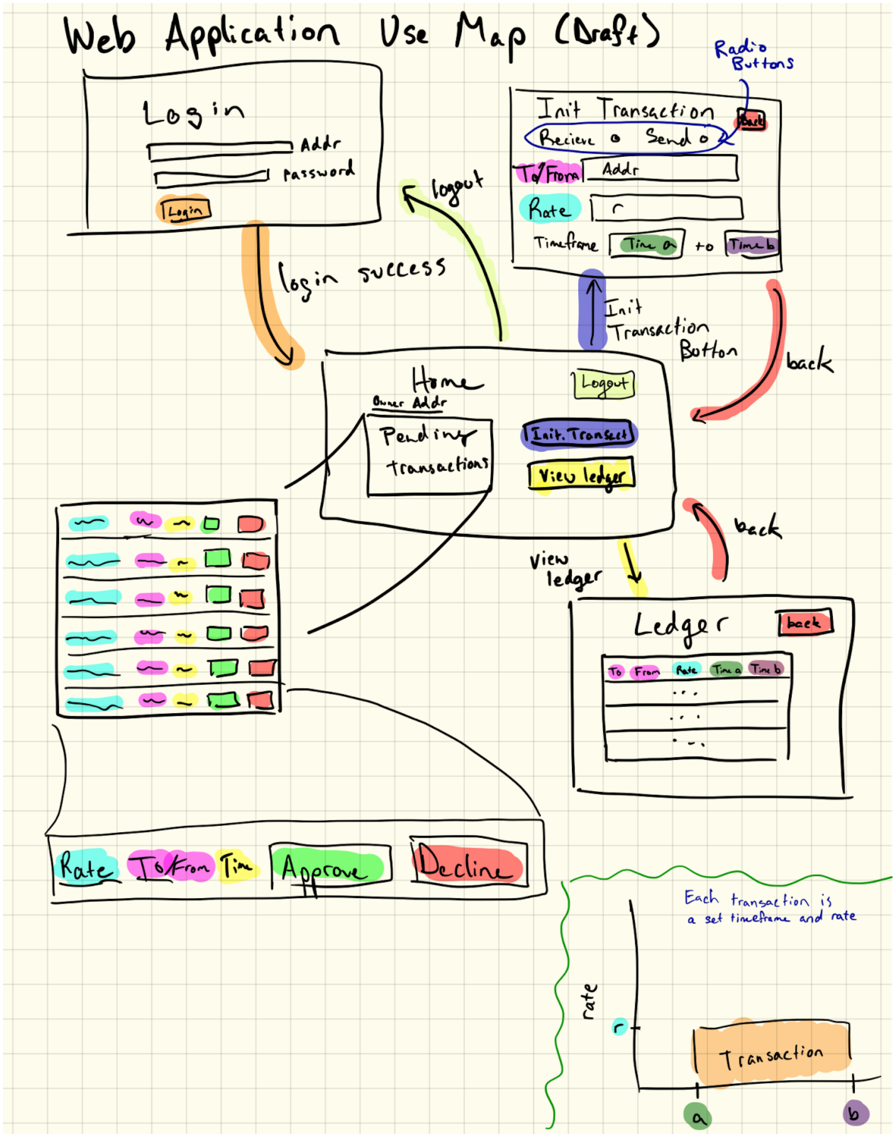

Figure 1: A rough map of the functionality of our web application

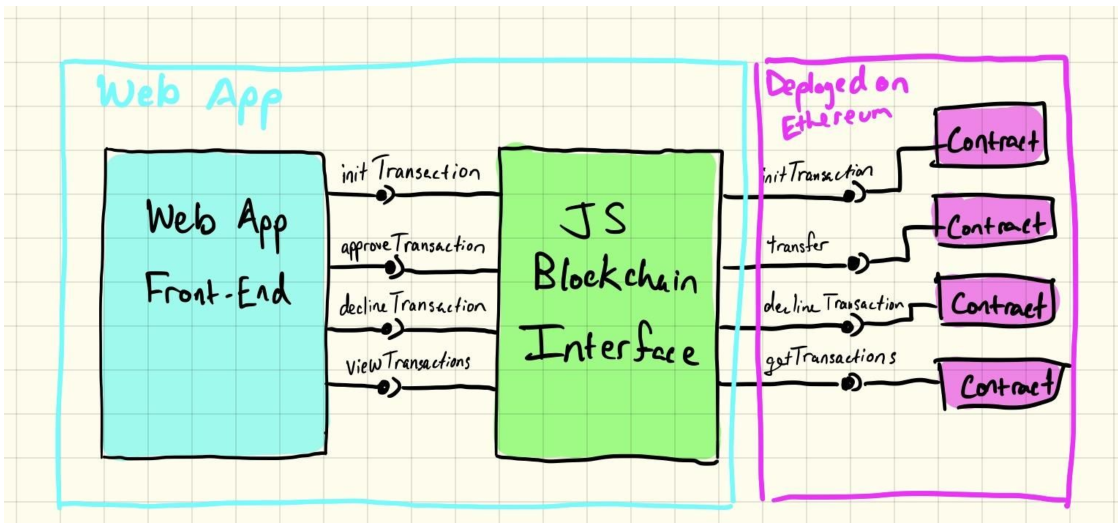

Figure 2: A macro-component diagram of how our web application interacts with the smart contracts in our blockchain

IoT Smart Meter

Based on research about products that are readily available for a reasonable cost, we determined that is was in our best interest to begin our development of the smart meter on a module that already has basic functionality like an Arduino. We will focus our efforts in this preliminary testing on the network and communication capabilities that will require us to learn the Arduino's general setup and software libraries. Once we are able to get this version of the hardware fully functional, we will assess this form of the solution and determine if it is acceptable. Depending on the results of this assessment, we may move on to trying to create the hardware with a custom PCB if time permits. Having two versions of the hardware implementation will be the most effective way for us to compare these solutions and determine which is the best one to choose for the long term, taking into account the cost and effort that went into creating both.

2.2 Design Analysis

For our software design, we are researching basic smart contract creation. Based on some preliminary research, we decided that the best way to interact with smart contracts once they were deployed was through a Javascript interface. We are still deliberating on the process by which a user would link their address to their smart meter. Going forward, we will continue experimenting with Ethereum smart contracts: specifically, how to create a custom cryptocurrency, and how to best set up the architecture of the blockchain contracts.

So far, for the hardware design, we have decided to use a Raspberry Pi in conjunction with a custom current sensor to create the IoT Smart Meter device. The Raspberry Pi was chosen over a standard arduino because the Raspberry Pi is a more powerful device and the fact that members of our software team won't necessarily need as deep of a knowledge of embedded system programming as they would with an arduino. As work goes on, we may decide that we want to create another prototype with an arduino in order to reduce price and power consumption for the end user. For the moment, we see Raspberry Pi as the best option because it should be easier to debug and troubleshoot while we are in the early stages of prototyping. The Raspberry Pi will also have a built-in Ethernet port which will make the networking that is necessary for our device much easier.

For the current sensor, we plan to connect some low-cost current clamps (rated in the ~120-240V and ~200-400A range) to a signal processing circuit and then feed that information to the Raspberry Pi for networking. We decided on this method of implementation because of its relative simplicity and the fact that it will allow us to compartmentalize the work for different groups of team members. The issues we foresee with this method is that it may not be streamlined enough for a final prototype. Cobbling the various different elements together could result in a clunky prototype but at this stage, we are just focusing and putting out a product that works.

3 Testing and Implementation

3.1 Interface Specifications

The only hardware and software interface that exists in our design is a current sensor connected to the GPIO of a Raspberry Pi. The Raspberry Pi will be keeping track of the energy available. When a new order is ready, the Raspberry Pi will keep track of the energy transfer and notify the block-chain upon completion. In order to test this interface, the sensor's output data will be verified followed by unit and functional testing of the current sensor and Raspberry Pi together.

3.2 Hardware and software

Ideally we will implement several levels of testing including unit and functional testing. At the lowest level we can expect unit and functional testing using Python's unit test modules.

As we develop the components of the system we will write unit and functional tests to verify appropriate use cases are covered. An example of this would be to develop the interface between the current sensor and the Raspberry Pi and to then develop a unit test to determine both individual units are working correctly following by tests based on use cases to verify they are working together correctly.

As a team we will develop use cases to determine the minimal viable product. For example the ability to purchase energy and keeping track of that transaction until it is completed as energy will take time to transfer.

A final step in testing will be integration testing with an actual meter. Those this is not required, we do hope to test this integration as it will be important to the viability of the product's future.. This will include building an enclosure, testing connections and simulating an energy transfer.

The hardware element of the project will be the smart meter. Within the smart meter, it will consist of two main parts: the current sensor and the arduino. Both of these parts will be used in the testing phase because they are critical to the success of the project. The best way to ensure that both of these parts function the way they need to is to test them.

The current sensor is a device that will be attached to a home or business in addition to, or in place of, their utility meter so it can measure the power being used by the building. This can be tested by using the lab equipment within the university for the initial and intermediate stages. For the final stage of testing, a test circuit of a few hundred homes could be used.

The arduino will be used as a gateway between the current sensor and the blockchain software, sending and receiving the necessary information for each element. It will be tested in conjunction with the current sensor by feeding known values to and from current sensor and checking to verify if the information provided by the Raspberry Pi is correct.

There are three software components that need to be tested. These include the blockchain implementation, user web portal, and full integration between these parts and the smart meter. The Ethereum implementation will be the first major component to test. It is the backbone of the energy trading system that we are establishing, and will need to be rigorously tested with test payloads to ensure that the transactions work as intended. When we can ensure the basic functionality of the blockchain implementation, the development and testing of the web portal can commence.

The exact details of what the web portal will contain are still being deliberated, but like all web pages, it's functionality, usability, and performance will be tested to ensure that everything works as intended. During development, each feature will be carefully unit tested to ensure that there aren't any simple errors preventing the web page for functioning properly. It's ease of use can be tested by many users, ensuring that it's easy to use for all potential users, not just blockchain experts.

The last thing to test will be the communication between the smart device, blockchain, and web portal. While integrating these three components, basic testing will be done, but the most extensive testing will be done by creating some basic end-to-end test cases that test the performance of the project as a whole. Only once these test cases are successfully completed can we be confident in the software performance of our project.

3.3 Process

Since we have yet to complete testing on each component in our system, we will focus on the tests we plan on using once we reach that milestone. Our initial tests will consist of white and black box testing of each component. We will first unit test the current sensor followed by unit testing the Raspberry Pi related methods. We will then conduct integration tests to ensure the current sensor and Raspberry Pi are communicating correctly.

The second stage in testing will be to unit test the methods written to interact with Ethereum. This will be followed by unit and functional testing the Web App. At this point integration testing for the entire system into a meter can be conducted. In general this testing will be very similar to the previous process however due to the nature of dealing with a live environment, we'll simulate some aspects. For example we'll simulate several users with smart meters and test the ability to purchase, transfer and verify the transaction.

3.4 Results

The only tests we have done at this moment are preliminary tests on the Ethereum platform to study feasibility, which have shown that we can create the tradable energy cryptocurrency that we have planned to create. We have yet to develop the majority of this project, and as a result have not tested much of anything. We will update this section as testing progresses.

4 Closing Material

4.1 Conclusion

Our first goal in taking steps towards the completion of this project was to solidify exactly what problem we were trying to solve. Essentially, our problem can be boiled down into the idea that money is the primary element for many decisions made, and as of right now, there is not enough of an economic incentive to prioritize the generation of renewable energy. We hope to produce this economic benefit by providing a way to make or save money from renewable energy via a blockchain implementation of peer to peer energy transactions.

After establishing the problem stated above, we determined that our solution would require three main components:

1. *IoT smart meter:* for reading energy usage and verifying transactions
2. *Blockchain implementation:* for facilitating secure transactions
3. *Web applications:* for allowing users to interface with our system

We will begin our implementation by keeping things as simple as possible to get a working prototype. Early on in the project, we hope to have a basic smart meter created using a Raspberry Pi and an external current sensing module. Along with this, we want to complete basic functionality on the web application and blockchain part of the software in order to test our system on a small scale. As time progresses, we will improve our system by learning what works and what doesn't, refining smaller aspects after building a solid functional foundation.

While there are groups like Grid+, LO3, and ConsenSys that have already made strides towards a similar solution, we feel that our team is starting our project at just the right time. We are able to learn from the mistakes from our predecessors by taking what they would have done differently and actually doing it differently. While we are not the first to work on this type of project, we are early enough that we are not fighting against any other groups that are dominating or monopolizing the market. With the structural background that we have established in this document, we hope to be able to develop a solution that meets the needs of the users at hand and takes strides towards increasing the worldwide consumption and generation of renewable energy.

4.2 References

ERC20 Token Standard: https://theethereum.wiki/w/index.php/ERC20_Token_Standard

https://spectrum.ieee.org/computing/networks/blockchains-will-allow-rooftop-solar-energy-trading-for-fun-and-profit

4.3 Appendices