Fostering Consumers' Energy Market through Smart Contracts

Ioannis Kounelis, Gary Steri, Raimondo Giuliani, Dimitrios Geneiatakis, Ricardo Neisse, Igor Nai-Fovino
European Commission, Joint Research Centre (JRC)
Cyber and Digital Citizens' Security Unit
Via Enrico Fermi 2749, 21027 Ispra, Italy
Email: {firstname.surname}@ec.europa.eu

***Abstract*—Micro-generation promises to greatly contribute to the energy balance of the energy grid; however, so far, its market penetration is going slow due to the few, or not-existing, direct economic benefits end-users would enjoy by deploying an in-house micro-generation system. In this paper, taking advantage of the potentialities of blockchain technologies, we propose a solar energy production and distribution architecture using smart contracts, a particular distributed ledger paradigm, to support automatic energy exchanges and auctions, potentially enabling a new, open and more fruitful, under an end-user perspective, energy micro-generation market. We present the conceptual design of the approach, as well the energy grid prototype and the control layer, running on the Ethereum platform. The proposed architecture has been implemented and validated through an in-house developed test-bed.**

I. Introduction

Micro-generation is the capacity for consumers to produce electrical energy in-house or in a local community. The concept of "market" indicates the possibility of trading the electricity that has been micro-generated among producers and consumers, where a user acting both as a producer and consumer is called a "prosumer". Traditionally, this market has been served by pre-defined bilateral agreements between prosumers and retail energy suppliers. This means that until now, electricity-generating prosumers have not had real access to the energy market, which remains a privileged playing field for the institutionalised energy suppliers. This fact has, so far, heavily impacted on the the real diffusion at large scale of micro-generation due to the limited economic advantages this energy generation approach would bring to the prosumers.

Indeed, the main options considered so far by the technical literature, were completely centralised and their viability (under a prosumer perspective) was in general challenged as they introduce additional management fees and costs and assume the intervention of a trusted third party reducing once again the potential gains of end-users. New approaches should be developed enabling end-users to have free access to the energy market. In this context the advent of distributed ledgers, i.e., blockchains, can be considered beneficial.

In particular, the use of a blockchain for energy representation and exchange provides several advantages. First of all, it gives the possibility to have a trusted and decentralised direct exchange between two parties. No intermediaries or third parties are needed in order to fulfill transactions. The data on the blockchain are public, easily verifiable by interested parties, consistent, and always available. Even if the data are available, the users remain pseudonymous, as for the transactions blockchain addresses and not personal data are used. Moreover, due to their decentralised nature and therefore lack of a central point of failure, blockchains are very resistant to denial of service attacks. Finally data on the blockchain are immutable, meaning that once inserted in the blockchain it cannot be altered, providing therefore a reliable point of reference. By having these features, blockchain provides a trusted technology that can be used as an Information and Communication Technology (ICT) backbone for an open energy market.

According to our approach, self-generated electricity could normally be either consumed within the house, accumulated in next-generation batteries for later use, or simply given back to the grid, where, thanks to the distributed and pervasive nature of the blockchain, the produced energy could be redeemed elsewhere. For example when charging an electric vehicle abroad, or sold through the blockchain to the best buyer, according to a mechanism similar to that of a stock-exchange market.

Exploiting the potentialities of blockchains and distributed ledgers, in this paper we propose a solar energy production and distribution architecture that uses smart contracts to support automatic and distributed energy exchange, thus allowing the development of an energy micro-generation market more open and fruitful, from an end-user perspective. More in detail we introduce a platform named Helios that facilitates micro-generators to exchange energy freely in a limited geographical area. In this setup a custom made Internet of Things (IoT) smart meter is used to account and register the micro-generated energy in the blockchain, while the smart contract supports the monitoring and accounting of energy exchange in terms of a financial transaction. The model has been implemented and validated through an in-house developed test-bed composed by a real physical energy infrastructure and the related control and ICT layers[1]. To the best of our knowledge, Helios is among the first solutions built on off the shelf devices and open source technologies, enabling prosumers to access the energy market.

In the following sections, we first provide some background information on blockchain technologies in Section II, and then after having presented a general overview of the approach in Section III, a description of the electrical grid physical

[1]The communication between all system elements is achieved through local or remote Internet Protocol (IP) connections.

978-1-5386-2066-3/17/$31.00 ©2017 IEEE

architecture is provided in Section IV. The system logic, together with the description and the code of the energy controller and smart contract is provided in Sections V and VI respectively. Finally, Section VII provides the related work and Section VIII presents the work conclusions and the future planned research activities.

II. Preliminaries

In this section we briefly describe the main characteristics of blockchains and smart contracts technologies our proposed solution relies on.

Blockchains are the backbones of cryptocurrencies, such as Bitcoin[2]. They are the technology on which cryptocurrencies are built on, and on which transactions can succeed without the need of having a trusted third party. In particular, a blockchain is a tamper-proof and shared data structure composed of a list of blocks of transactions. The blocks are distributed to all nodes of the network and contain all the transactions that took place from the creation of the cryptocurrency. New transactions are inserted in the end of the chain and are linked to the previous block of transactions, as each block references the previous block's hash.

The intrinsic nature of blockchains presents some interesting advantages:

- *Disintermediation and Trustless model*: exchanges (or transactions) do not require intermediaries or trusted third parties; moreover, the parties have full guarantee that the transactions will be executed as expected
- *User Empowerment*: transactions and data are in control by the users community
- *Resilience*: due to their decentralised nature, blockchains do not have a central point of failure
- *Transparency and immutability*: every modification in public blockchains is visible to everybody, moreover, the transactions stored in a blockchain cannot be altered or deleted as it is not computational feasible to do so
- *Low transaction costs*: being completely unsupervised, the intermediaries costs are eliminated

For these reasons, blockchain can be used to implement other services apart from currency transactions. One of the most promising is smart contracts. A smart contract is a computer program that is capable of executing or enforcing a predefined agreement using a blockchain, when and if specific conditions are met. Its main goal is to enable two parties to perform a trusted transaction without having the need of intermediaries. Moreover, smart contracts inherit the characteristics of blockchains and thus have no downtime, censorship or third party interference.

In the model presented in this paper we build up on the open-source Ethereum[3] blockchain-based distributed computing platform, namely the Ethereum Virtual Machine (EVM). The main goal of the EVM is to keep a distributed record of transactions performed using the Ethereum digital currency, Ether (ETH). A blockchain-based platform such as the EVM can be seen as a distributed database that can be accessed/managed by many people that do not necessarily trust each other and do not share a common trusted 3rd party. In contrast to other blockchain-based platforms that target mainly mining of the digital currency and transaction management, Ethereum also provides smart contracts as a core functionality. In Ethereum, an obligatory payment fee, named *gas*, is required in order to finalise the transactions.

[2]https://bitcoin.org/

[3]https://www.ethereum.org/

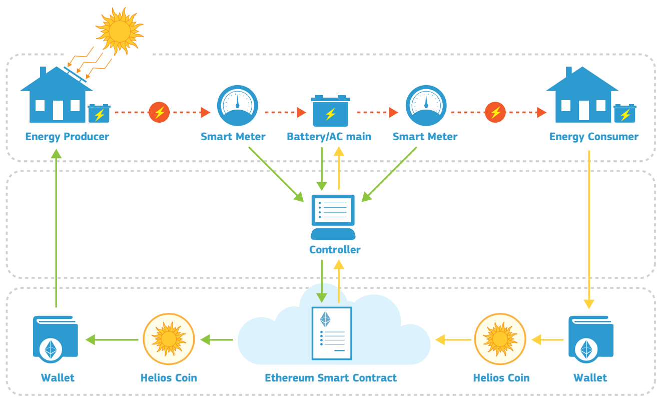

Fig. 1. Helios model high level architecture

The functionalities that the Ethereum smart contracts provide along with its wide and well documented use make it ideal for the development of our prototype. Moreover, it provides a user friendly Javascript Application Programming Interface (API) for accessing the smart contract's functionalities from a third application. We have used this feature for developing our middleware controller, as described in Section V.

III. Helios Model Overview

This section provides an overview of the general principles adopted by the *Helios* model proposed in this paper.

The main aim of our model is to enable micro-grid prosumers to produce, consume and trade energy. In particular, they would be able to:

- **Produce** energy and store it in an in-house cache-battery (for local energy consumption)
- **Consume** the stored energy
- **Release** excess energy to the grid and receive virtual coins in return
- **Transfer/Exchange** the virtual coins
- **Redeem** the virtual coins in exchange with energy

In our model, we assume a local grid where energy is produced and consumed in a limited geographical area, such as a local neighbourhood. Energy produced by a prosumer may be saved in the user's local battery for later use or may be immediately injected in the local grid. An additional possibility is to have a common, central to the neighbourhood, battery shared as a temporary energy buffer. As it can be observed by the high-level overview of the Helios model in Figure 1, the

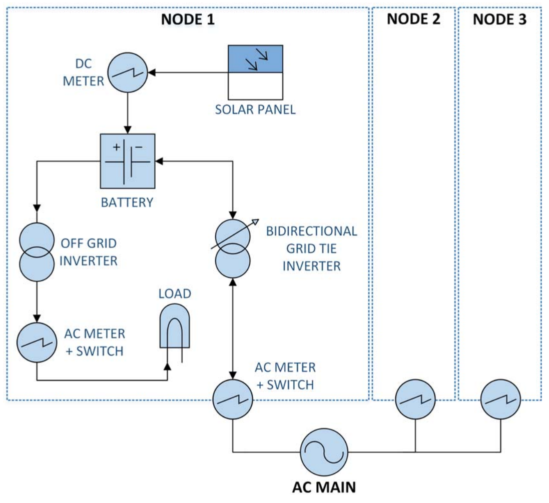

Fig. 2. Energy grid model main components

model is divided in three layers: (a) the energy grid, (b) the middleware controller, and (c) the smart contract.

When energy is injected in the grid a smart meter linked to each producer continuously measures how much energy has been injected in total. These smart meters, along with the software that handles their output, i.e. a middleware controller, are the input source for our smart contracts. After a predefined amount of energy has been injected to the grid, an Helios Coin (HEC) is awarded to the corresponding prosumer.

The middleware controller interconnects the grid with the smart contract since these systems cannot communicate directly with each other. As a result, the controller plays the role of invoking the smart contract on one end, and on the other receiving the readings from the grid, thus facilitating communication between the two entities.

The energy grid is handled by its own smart contract. It is aware of the entities connected to it, it can transfer a specified amount of energy to a connected energy consumer, and it is aware of how much energy for consumption is available at any time. The grid's smart contract takes as input HECs and then releases the energy that corresponds to the amount of HECs received in the payment by the sender.

The way HECs can be circulated in a market depends on their owners interests and strategies. The simplest way would be for each owner to have a smart contract in which he sells HECs in exchange for another asset or coin. It should be noted that the smart meters, and the electrical grid in general, is considered a trusted party; meaning that its measurements and operations are considered reliable and are treated as such.

IV. Smart Energy Grid Model

In this section we present the description of the energy grid (as seen on the top part of Figure 1) of the Helios model as well as the implementation of this model using off the shelf components and IoT devices .

The model is based on the concept of a smart power island; i.e. a local area power distribution system. The high level architecture of our model is illustrated in Figure 2. Specifically, it is composed of an Alternating Current (AC) Main single phase distribution line using a bus topology. The line provides power exchange among nodes, while power is exchanged using a locally generated frequency at 220/110 Volt nominal. Within the island there are at least two nodes that are capable of generating, consuming and storing power. The basic rules regulating the nodes are:

1) They must conform to the frequency provided by the AC Main line
2) They must comply to a fixed capped maximum power that can be exchanged with the line
3) The cap can be different for power injection or withdrawal
4) Power and energy metering must be provided by the nodes in the point where it exchanges power with the local grid

Any node complying with the above rules can be added to the network as a producer and/or a consumer. Any node can access the local line provided it respects the above mentioned rules. This model can represent both the on grid or off grid operation of the network. Note that for our custom made system, each node includes an AC and a Direct Current (DC) subsystem. The DC subsystem is made of a 12 V battery (300 AH), a solar panel (130W) and a power controller for charging the battery. Two inverters and an AC to DC battery charger are found in the AC subsystem.

Within the island there is also a special gateway node with the following characteristics:

1) It must provide AC synchronization frequency to the line
2) It must provide energy absorbing capabilities up to the sum of the maximum production rating of all nodes
3) It must provide energy injection capabilities up to the sum of the maximum consumption rating of all nodes
4) Power metering of total incoming and outgoing power must be provided.

Therefore, the distribution line can be connected via a gateway node to an ideal generator/load, to a main grid with the same characteristics, or to both. The gateway node is representative of a main power grid connection or a local power generating/storing device such as a neighbourhood battery.

To control and monitor the power flow, we rely on custom made smart meters that enable power management through an IP connection. To do so, we make use of Internet of Things devices supported with Wi-Fi and Ethernet interfaces. Specifically, the metering and control subsystem are made of an Arduino Yun and an Emon Shield board correspondingly. The metering subsystem is capable of measuring AC current, DC current and AC/DC voltage. Listings 1 and 2 show a portion of code for AC and DC readings respectively. The control subsystem is used to route power within the node among the loads, the grid, and the battery storage. Furthermore, to enable the power-management over an IP we expose the services of metering and power control through an Hypertext Transfer Protocol (HTTP) interface that supports the following commands:

- Open/Close relay N [write command]

Listing 1. AC reading.
```
if (CT1) emon1.calcVI(50,2000);

if (CT1){
        pfPow1[meas_count%mem_d]=emon1.realPower;
        accum1=0;
        for (int ii=0;ii<mem_d;ii++)
        accum1+=pfPow1[ii];
        accum1=accum1/mem_d; }
```

Listing 2. DC reading.
```
sensorValue = analogRead(analogInput0);
outputValue = ((long)sensorValue - 500 ) * 1000 / 133;
amps = (float) outputValue / 1000;
value = analogRead(analogInput1);
vout = (value * 5.0) / 1024.0;
vin =calibB*vout / (R2/(R1+R2));
if (vin<0.09)
   vin=0.0;
```

- Read AC Meter (N, param) [read command]
- Read DC Meter (N, param) [read command]

where *N* is the sensor number and *param* can be voltage, current, instant power, average power. All read commands will return a timestamp from the sensor's local clock.

V. MIDDLEWARE CONTROLLER

In order for the smart contract to communicate with the grid, to get the measurements of the smart meters and issue commands to release the requested energy towards a client, a middleware application is needed to facilitate the communication between the two parties (as seen in the middle part of Figure 1). This application, is the node controller application. The controller interacts with the physical model, as described in Section IV, with the Arduino Yun board. As a result, the controller has in real time the data transmitted by the smart meter and can immediately issue HECs. The controller is a single application that handles simultaneously all the smart meters of the local grid. It is also the entity that is the owner of our smart contract. On the other end, the controller communicates with the smart contract using the Web3 Javascript Đapp API[4].

The controller reads from the smart meter every hour. It then calculates the difference with the previous measurement and is thus in position to determine how much energy has been put in the grid from the corresponding user. Once the energy committed to the network has been calculated, new coins will be issued and assigned to the energy creator. The controller will have to call the mint function, i.e. the function that issues new coins, of the smart contract and pass the two necessary parameters: the address of the creator and the value of energy created in Wh (watt-hour). To do this, however, it will first need to unlock the account as for this transaction a fee for the gas consumed has to be paid to the network (in Ether).

As the creation of coins assumes a cost for the controller, and thus for the smart contract owner, the transaction frequency should be well estimated. Even if the cost is small, it is still an extra burden that will in the end affect the energy producer. As a result, the frequency of the control of the smart meter and thus the coin creation should be set up taking into account the average amount of energy that is produced by the user and his preference on receiving immediately coins for the energy that has been committed into the network. These parameters can be set up during the registration of the user on the grid and then fetched when needed by the controller.

[4]https://github.com/ethereum/wiki/wiki/JavaScript-API

For our demo, we assume this factor not relevant and have it accumulated with the estimated loss of energy during the energy transactions. Moreover, as our scope is to test the proposed solution and monitor its behaviour, we have set the coin generation frequency to a low limit of one Wh. For a market ready application the limit will need to be adapted to higher values, always depending on the solar panels output.

When it comes to releasing energy towards a user, when coins have been returned to the smart contract owner, the controller gets informed by incoming transactions by following the smart contract's event that announces coin transfers on the network. When a new transaction towards the contract owner's address arrives, the controller communicates with the Arduino board and it issues a command to release an amount of energy that corresponds to the coins received towards the sender of the transaction. The coins then remain at the contract owner's address and are considered spent. The controller's functions that create coins and monitor the transfers in the network in order to release energy can be seen in Listing 3.

Listing 3. Creating and utilizing Helios Coins from the controller
```
// creation of contract object
var myContract = web3.eth.contract(abiHEC);
var myContractInstance = myContract.at(addressHEC);
var events = myContractInstance.Transfer();
events.watch(function(error, result){
if (!error) {
 if(result.args.to == addressOwner){
  console.log('Releasing ' + valueTransfer + ' Wh to ' +
       fromTransfer);
  releaseEnergy(fromTransfer, valueTransfer);} } });

//Coin creation
function createHEC(addressProducer) {
var Wh = smartMeterReader() - lastSmartMeterReading;
//unlocking account
web3.personal.unlockAccount(allAccounts[0], password,
     function(error, result){
 if(!error){console.log(result);} });
//Mint tokens
myContractInstance.mintToken(addressProducer, Wh, function(
     error, result){
 if(!error)
  console.log(Wh + 'Wh sent to ' + addressProducer); }); }
```

VI. SMART CONTRACT IMPLEMENTATION

The smart contract we have deployed (as seen in the lower part of Figure 1) has the role of the record keeper, with the corresponding reward mechanism. The smart contract is written in Solidity, Ethereum's native programming language, and is deployed on Ethereum, both on Ethereum's official testnet (Ropsten) and on our own private network. Every time a user commits energy in the grid, the smart contract will issue coins that correspond to the energy produced and automatically send them to the energy producer.

More specifically, the smart contract has a mint token function that takes as input two parameters: the address of

Listing 4. The basic functions of the smart contract

```
event Transfer(address indexed from, address indexed to,
     uint256 value);

//Constructor
function MyHeliosToken(string tokenName, string tokenSymbol)
      onlyOwner(){
balanceOf[msg.sender] = initialSupply;
symbol = tokenSymbol;
name = tokenName; }

//Create coins
function mintToken(address target, uint256 mintedAmount)
     onlyOwner {
balanceOf[target] += mintedAmount;
totalSupply += mintedAmount;
Transfer(this, target, mintedAmount); }

// Send coins
function transfer(address _to, uint256 _value) {
if (balanceOf[msg.sender] < _value) throw;
if (balanceOf[_to] + _value < balanceOf[_to]) throw;
balanceOf[msg.sender] -= _value;
balanceOf[_to] += _value;
Transfer(msg.sender, _to, _value); }
```

the entity that has produced the coins and an unsigned integer that represents the number of HECs to be issued. What this function does is to issue new coins and assign them directly to the indicated address. The energy producers will need to follow the token in order to detect incoming transactions and thus see their newly created coins. They can easily do so from the graphical interface of the Ethereum wallet; the only parameter needed is the smart contract's address. Once they are in possession of the coins they can circulate them freely according to their desires; they can choose to sell, use or exchange them.

From the smart contract's address, one can verify at any time the balance of any address, view the total supply in coins, view the smart contract's name and symbol, view the smart contract's owner, and follow live the transactions related to the smart contract that occur in the blockchain. The functions of the contract are viewable for everyone, but apart from the transfer function, i.e. the function to transfer coins from one user to another, are only executable by the contract's owner who in this case is the node controller. In order to watch the contract and see its functions, apart from the contract's address the metadata description of its interface is needed. Some of the basic functions of the smart contract can be seen in Listing 4.

According to the description done previously, the deployment of the contract has to be done on the controller node, which is commonly used by all the users of the electric system. In this optic, in order to guarantee transparency and the application of the same rules for everyone, we envisage that the deployment of the contract and the management of the controller node is done by a consortium composed of all the neighbours connected to the electric subsystem.

With the deployment of the smart contract we have managed to create a fully working smart grid energy trading system. The system is now able to automatically detect energy inputs and through the controller issue coins with the smart contract. The coins are then circulated as virtual tokens on Ethereum and when they are redeemed they are returned to the controller's address, which on its side communicates with the electric grid in order to release the equivalent to the coins energy. Moreover, the coin transfers and addresses' balance are public and can be monitored by any interested party, as usually expected from a blockchain-based approach.

VII. Related Work

In this section we describe existing approaches applying digital currencies in the energy area and how they differ from our proposed approach. Bankymoon [1] is a startup in South Africa proposing to use smart meters connected to the blockchain allowing users to load Bitcoins in order to enable the energy flow. In this approach the cryptocurrency is simply used as a prepaid payment option.

Other solutions also using simply cryptocurrencies as a payment option are Solether and BlockCharge. Solether[5] is an open source proposal of an autonomous node for energy management consisting of a solar panel, a battery, and an Intel Edison board that interfaces with the Ethereum blockchain. The node is associated with an Ethereum address and it is therefore considered an Ethereum entity. Whenever a payment is received the energy flow is enabled through the USB port, which can be used to charge or power a device. The amount of energy flow allowed is automatically accounted according to the amount of the received payment. In the long term the amount of money received in the device account could surpass the cost of the device itself, allowing for the device to actually work as an asset producing money for its owner.

BlockCharge uses the Slock.it technology that proposes a Smart Plug to enable on the go charging of electric cars using a cryptocurrency. Slock.it is a blockchain-based approach to rent or sell anything directly without intermediaries, which in the case of BlockCharge is selling of electrical energy. BlockCharge itself concentrates on the energy market but the Slock.it technology has a more broad coverage of smart contracts for any application domain.

More advanced applications of blockchain technology are the SolarCoin[6] and GrünStromJeton[7] reward programs. SolarCoin is a global rewards program for solar electricity generation created in 2014 by a group of volunteers interested in helping the environment. The idea is to award producers of solar energy globally with a digital currency named SolarCoin where 1 coin represents 1 megawatt-hour (MWh) of solar electricity generation. In their technical implementation the SolarCoin infrastructure is described as a lite version of Bitcoin, using scrypt as a proof-of-work algorithm. The source code is open source and available online[8]. Similarly to the SolarCoin program, the GrünStromJeton is a proposal for awarding customers with tokens that serve as an indicator of sustainability of current use and production, as well as to determine their CO2 footprint.

Some startups also propose to use blockchain technology to enable home energy producers and consumers to exchange energy credits in a distributed and dynamic way. A pioneer work on this domain is the TransActive Grid technology, which

[5]http://solether.mkvd.net
[6]https://solarcoin.org
[7]https://stromdao.de/gruenstromjetons
[8]https://github.com/onsightit/solarcoin

has been applied in the world's first peer-to-peer blockchain energy solution, employed in the Brooklyn Microgrid[9]. Similarly to our proposal, in the TransActive Grid architecture each house in a neighbourhood acts as an energy producer and consumer. When the house generates energy the smart meter detects energy production/injection and energy tokens are generated to the home owner. The home owner can sell the tokens to neighbours that can then use them. Once used, the tokens are destroyed by the energy consumer's smart meter when it detects the inflow of energy. Consumers can purchase renewable energy credits from 3rd party retailers or buy tokens/credit from their neighbours directly enabling a local microgrid energy market. Whenever a house generates energy, the smart meter detects it and it is usually consumed by the nearest loads.

Another startup example is GridSingularity[10], which targets the energy finance market using a blockchain-based platform. The platform is based on Ethereum and the beta version is currently under development.

All these startups including TransActive Grid and GridSingularity, in contrast to our work, do not provide technical details of their solutions for strategic reasons. For example, there are no details on the smart contract implementation, how the smart contracts interact with the smart meters, what the architecture of the system is, etc.

In the academic literature the concept of energy tokens for trading of energy was discussed in a high-level by Dimitriou & Karame [2], where producers of energy receive tokens directly from the utility provider when energy is injected in the utility grid. In their work also security and privacy issues are highlighted. More recently, Aitzhan & Svetinovic [3] also propose a multi-signature approach to enable security and privacy in a decentralised energy market.

The NRGcoin [4] is currently the only decentralised digital currency proposed in the academic context targeting exchange of energy in smart grids. Together with the NRGcoin currency the authors also propose a novel trading paradigm for buying and selling green energy using a double action process. The main difference with our proposal is that the NRGcoin is a separate coin, built on its own blockchain, while Helios Coin is based on a smart contract. Moreover, there are no technical details on how NRGcoins are created and how the proof of work functions in practice. It seems to be a theoretical proposal, focusing mostly on the NRGcoin trade market.

Close to our approach is the issue of US Renewable Energy Credits (REC) as a cryptocurrency on Ethereum's blockchain, described in [5]. Here the purpose is to represent the RECs as a new cryptocurrency on Ethereum that can be exchanged or traded. However, also this paper does not provide a description of the physical or software implementation.

VIII. Conclusions and Future Work

In this paper we presented Helios, a solar energy distribution system controlled by a smart contract running on an Ethereum blockchain. We have developed and tested the whole infrastructure of the system, from the assembly and configuration of all the devices in the physical layer (solar panels, batteries, smart meters, IoT control devices) to the implementation, and deployment of the smart contract based on Ethereum. In contrast to the other solutions presented in literature, we describe and document all the design and implementation steps of a complete solution that enables users to access the energy market without the need of interacting with intermediaries central entities.

The physical part, presents a certain degree of flexibility, and it can be configured to work in an autonomous island or connected to a main grid. For what concerns the logic, the one adopted in our approach is the first version of the contract. Our goal was not to provide a platform fully enhanced with trading capabilities, but instead to prove feasibility of the proposed design and architecture. We already envisage improvements for our energy model and will gradually pursue them in the next phase of the implementation.

First of all, we plan to study the effectiveness of the system when used in a commercial mode and not only as purely experimental one, as well as comparing various tarification schemes. One factor that could have an impact on this is the energy loss during transfer, which should be considered in the implementation of the system logic and transactions. Moreover, the business model of the system must be well studied in order to implement an end product that can be used in a real environment. One other interesting feature that could be explored is the possibility for our model to interact with an utility company for a bi-directional energy exchange when it is needed.

For what regards the smart contract, we plan to extend it with more complex functions and allow the use of coins from third parties, automatic control of transaction fees from each owner's account, and a market for exchanging HECs. Finally, we would like to implement an independent cryptocurrency that would use the energy creation as proof-of-work for the creation of new coins. We want to compare this method with the one described in this paper and find the advantageous and disadvantageous aspects of each one.

[9]http://brooklynmicrogrid.com

[10]http://gridsingularity.com

References

[1] "Smart meters prepaid: Bankymoon develops bitcoin solution," AMI & Smart Metering, 04 2015, available at: https://www.metering.com/smart-meters-payment-bankymoon-develops-bitcoin-solution/.

[2] T. Dimitriou and G. Karame, "Privacy-friendly tasking and trading of energy in smart grids," in *Proceedings of the 28th Annual ACM Symposium on Applied Computing*, ser. SAC '13. New York, NY, USA: ACM, 2013, pp. 652–659. [Online]. Available: http://doi.acm.org/10.1145/2480362.2480488

[3] N. Z. Aitzhan and D. Svetinovic, "Security and privacy in decentralized energy trading through multi-signatures, blockchain and anonymous messaging streams," *IEEE Transactions on Dependable and Secure Computing*, vol. PP, no. 99, pp. 1–1, 2016.

[4] M. Mihaylov, S. Jurado, N. Avellana, K. V. Moffaert, I. M. de Abril, and A. Now, "Nrgcoin: Virtual currency for trading of renewable energy in smart grids," in *11th International Conference on the European Energy Market (EEM14)*, May 2014, pp. 1–6.

[5] R. D. Leonhard, "Developing renewable energy credits as cryptocurrency on ethereum's blockchain," Broader Perspective, 12 2016, available at: https://ssrn.com/abstract=2885335.