Blockchain-based Traceability of Carbon Footprint: A Solidity Smart Contract for Ethereum

António Miguel Rosado da Cruz[1,2,*], Francisco Santos[1], Paulo Mendes[1] and Estrela Ferreira Cruz[1,2,*]

[1]*Instituto Politécnico de Viana do Castelo, 4900-347, Viana do Castelo, Portugal*

[2]*Centro ALGORITMI, Escola de Engenharia, Universidade do Minho, Guimarães, Portugal*

Keywords: Blockchain, Solidity, Smart Contract, Ethereum, Carbon Footprint, Traceability, Monitoring.

Abstract: In recent decades there has been an increasing concern about climate change. Every person is increasingly concerned about global warming and, as a consumer, with their own individual contribute to that issue, wich may be measured by each one's carbon footprint. In this sense, it is only natural that each person wants to consume products with a lower carbon footprint, meaning with a lower environmental impact. For this, however, consumers need to be able to know the carbon footprint of the products they are buying. This is only possible by having every company tracking and sharing their own products carbon footprint. The blockchain is a distributed technology that allows for registering and sharing information between those companies and the final consumers. The blockchain is being used in many areas as a distributed database, and has some strong points like trust, transparency, security, immutability, durability, disintermediation and others. In this paper the blockchain technology is being used to track and trace back the carbon footprint of products and organizations. More exactly, this paper proposes a smart contract-based platform for the traceability of the carbon footprint of products and organizations.

1 INTRODUCTION

Climate change is today one of humanity's greatest challenges. Efforts to fight it tend to focus on reducing known causes, such as the concentration of greenhouse gases in the atmosphere. One of the major causes of climate change is the so-called carbon footprint of mankind. Year 2018 was the year when the average ocean temperature was highest since the beginning of recordings, and atmospheric concentrations of greenhouse gases such as carbon dioxide (CO_2), methane (CH_4) and nitrous oxide (N_2O) were the highest ever (NOAA, 2019; C3S, 2019). The growth of these indicators, along with many other directly or indirectly related indicators, is the reason why more and more environmental awareness is taking a prominent role, which it had not a few years ago.

Carbon footprint may be defined as "the amount of gaseous emissions that are relevant to climate change (gases with greenhouse effect) and associated with human production or consumption activities" (Wiedmann and Minx, 2008). Different greenhouse gases last in the atmosphere for different amounts of time and absorb different amounts of heat. Carbon footprint, measured in carbon dioxide equivalent (CO_2e), allows to describe different greenhouse gases in a common unit. For any quantity and type of greenhouse gas, CO_2e refers to the amount of CO_2 that would have the equivalent global warming impact (C3S, 2019). A carbon footprint considers the seven Kyoto Protocol greenhouse gases: Carbon dioxide (CO_2), Methane (CH_4), Nitrous oxide (N_2O), Hydrofluorocarbons (HFCs), Perfluorocarbons (PFCs), Sulphur hexafluoride (SF_6) and Nitrogen trifluoride ((NF_3)[3]). CO_2e is calculated by multiplying the emissions of each of the six greenhouse gases by its 100-year global warming potential (GWP). The GWP of CO_2 is 1, and the GWP for all other greenhouse gases is the number of times more warming they cause compared to CO_2 (CarbonTrust, 2019; C3S, 2019).

Wiedmann and Minx mainly discuss two methodologies for carbon footprint analysis: process analysis and input-output analysis. For the assessment of individual products, the authors suggest a Hybrid-EIO-LCA approach (EIO- Environmental input-output, LCA- Life Cycle Analysis/Assessment), where lifecycle assessments are combined with input-output analysis. The authors also noted it is important for

*Contact Authors.

Rosado da Cruz, A., Santos, F., Mendes, P. and Cruz, E.
Blockchain-based Traceability of Carbon Footprint: A Solidity Smart Contract for Ethereum.
DOI: 10.5220/0009412602580268
In *Proceedings of the 22nd International Conference on Enterprise Information Systems (ICEIS 2020) - Volume 2*, pages 258-268
ISBN: 978-989-758-423-7

a carbon footprint to include all direct as well as indirect CO_2 emissions, and it is also "important to avoid double-counting along supply chains or life cycles" (Wiedmann and Minx, 2008). So, for assessing a product carbon footprint within a value chain, one needs to measure and store information about carbon emissions on every step of the product life cycle, from conception or production, to transport, storage, etc. For implementing this, we decided to use the blockchain technology.

Blockchain is a distributed registration technology that aims at decentralization as a security measure, and in which all committed transactions are stored in a chain of blocks (Zheng et al., 2018). This chain is constantly growing as new blocks are added to it in a linear and chronological way. It is a decentralized system, where each node of the network gets a copy of the database after joining the network and has the task of validating and passing on transactions.

Some blockchains, like Ethereum, offer a decentralized virtual machine to handle smart contracts, which are like digital contracts that control users' digital assets, formulating the participants' rights and obligations (Lin and Liao, 2017). Ethereum is a decentralized platform capable of executing smart contracts and decentralized applications using blockchain technology. It has a Turing complete decentralized virtual machine, the Ethereum Virtual Machine (EVM), which can encode rules and run scripts for processing transactions (Luu et al., 2016).

The blockchain technology is being used as a distributed and replicated database in many areas (Saberi et al., 2019; Tian, 2017). The technology has some strong points like trust, transparency, immutability, among others (Dujak and Sajter, 2019).

There are three main types of blockchain: permissionless public blockchain; permissioned public blockchain (hybrid blockchain) and permissioned private blockchain (closed network) (Pedersen et al., 2019). Blockchain technologies can be used in many areas like digital currency (e.g. Ethereum, Bitcoin), Smart Contract (e.g. Ethereum, Hyperledger), protection of Intellectual property, Traceability in supply chain, Identify certification, International payments, Financial services, Risk management, Internet of things (IoT), etc. (Lin and Liao, 2017).

This paper's main contribution is to present an approach to trace and (dynamically) calculate the carbon footprint of products and organizations, by using a permissionless public blockchain, namely Ethereum. The paper also presents a distributed application providing to consumers information about the carbon footprint of a product or organization stored in the blockchain.

The structure of the presentation is as follows: In the next section, related work is presented, concerning the value chain traceability of carbon footprint and blockchain-based approaches for traceability of other product's properties. Then, in section 3, the conceptualization of the proposed traceability platform of products' and organizations' carbon footprint is described. Section 4 reports on the development and validation of the platform's smart contract, and section 5 addresses the architecture and development of the blockchain-based distributed application. Section 6 concludes the paper and raises ideas for future work.

2 RELATED WORK

2.1 Tracing Carbon Footprint

It is not easy to find articles alluding to the carbon footprint traceability of a product in the literature. Some articles present methods for calculating carbon emissions, as is the case of (Li et al., 2013). Li *et al.* present a method for tracing the carbon flow to determine carbon emissions obligation from electricity consumption. In (Cordero, 2013), the author briefly surveys carbon footprint estimation approaches within supply chains.

Fu *et al.* propose a framework to expose the FAMI (fashion apparel manufacturing industry) carbon emissions to the general public and define a set of guidelines for reducing carbon emissions at all stages of manufacturing (Fu et al., 2018). The study starts by reviewing existing blockchain applications to improve sustainability and then the authors propose a blockchain based solution to encourage lower carbon emissions (Fu et al., 2018).

In (Pan et al., 2018) a blockchain-based application to carbon trading is proposed.

To our knowledge, there is not in the literature a proposal presenting a platform for tracing a product's carbon footprint using blockchain technology. Despite this, the next subsection presents some works that use the blockchain technology for value chain operational support or location traceability.

2.2 Blockchain-based Treacealility Platforms

Blockchain technology embraces concepts like decentralization, transparency, open source, autonomy, anonymity and immutability (Lin and Liao, 2017). It also has some disadvantages, though. Compared to traditional centralized databases, Blockchains have limited efficiency and require larger storage capacity, leading to poor performance and much higher energy costs. The blockchain technology can be a great solution to be used in some situations but it certainly is not the best solution to be used in all of them. Pedersen *et al.* propose a set of ten steps to help in deciding if the blockchain technology is the best option to a specific problem or not, and if so, what kind of blockchain is a better fit (Pedersen et al., 2019).

In (Deloitte, 2019), Deloitte performed a survey about integrating blockchain into their business operations. The study involved around 1,000 big companies from 12 countries from 4 continents. This survey, from 2019, concluded that 34% already had a blockchain system in production and 41% expected to implement a blockchain application within the next 12 months. The survey concludes that “53% of respondents say that blockchain technology has become a critical priority for their organizations”. In comparison to the survey previously performed in 2018, the authors notice a 10-point increase in 2019 over 2018.

Value chain integration and digitization is becoming increasingly important, not only for practical reasons, but also for reasons of transparency and security (Cruz and da Cruz, 2019). Korpela *et al.* research the supply chain integration and the role played by blockchain technology in digital supply chain transformations. According to the authors, value chain integration has advantages for everyone involved: suppliers, customers, intermediaries, etc. The authors present several advantages such as: increase value chain efficiency; ease of the communication between chain operators; possibility of tracing the product since the beginning of the chain; decreased need for human intervention (automatic data collection) reducing human error, etc.(Korpela et al., 2017).

In (Saberi et al., 2019), the authors identify some strengths of the use of blockchain for the integration of the value chain. Blockchain technology can support data collection, storage, and management, thus enabling support for most of the value/supply chains information. Saberi *et al.* focus their attention mainly on the use of blockchain in sustainable supply chains. According to the authors, blockchain can be used to: ensure that green products are environmentally friendly; trace the carbon footprint of a product; improve recycling; improve emission trading schemes efficacy; etc. (Saberi et al., 2019).

Several authors are using blockchain for traceability in food supply chains, as is the case of (Biswas et al., 2017), (Tan et al., 2019), (Tian, 2017), (Galvez et al., 2018), (Caro et al., 2018) and others.

Biswas *et al.* propose a blockchain-based wine supply chain to record detailed information in order to trace the origin, production and purchase history of the wine. The authors designed and implemented a blockchain with five entities (one entity for each wine chain operator) (Biswas et al., 2017). Three of the entities were designed as the miners and all entities in the chain generate their own blocks containing transactions. These transactions are verified by the miners before being added in the chain (Biswas et al., 2017).

Feng Tian proposes a decentralized system based on IoT and blockchain technology to support traceability in food supply chain (Tian, 2017). The author identifies a set of advantages and disadvantages of blockchain databases and proposed a new concept, named as BigChainDB, by combining the benefits of distributed databases and blockchain. This way the author deals with blockchain problems like low throughput, high latency and scalability (Tian, 2017).

In Galvez *et al.* the authors main goal is to study the use of blockchain to store data from chemical analysis to ensure transparency in food supply chain, avoiding food falsification or adulteration. The authors study the potential of blockchain technology for assuring traceability and authenticity in the food supply chain (Galvez et al., 2018). The authors conclude that the use of blockchain has many advantages in traceability, such as allowing all stakeholders to check the entire history and current location of a product; assures geographic biological origin by collecting information from sensors and store it directly into the blockchain(Galvez et al., 2018)..

Caro *et al.* propose a decentralized solution for agri-Food supply chain management and traceability based on blockchain. The presented solution is able to integrate data produced and consumed by IoT devices. The authors used Ethereum and Hyperledger Sawtooth blockchains (Caro et al., 2018).

Luu *et al.* research the security problems of executing smart contracts on the Ethereum platform. The authors identify several vulnerabilities and propose solutions to overcome them and make contracts more secure (Luu et al., 2016). They also propose a tool to identify and alert for potential security flaws when coding smart contracts (Luu et al., 2016).

3 THE CARBON FOOTPRINT TRACEABILITY PLATFORM CONCEPTUALIZATION

Informed people tend to make better, or more conscious, choices when presented with various alternatives as long as there is no personal detriment. The best way to encourage consumers to choose products with the least impact on the environment is to provide a platform where each consumer may follow the production chain of a given product, and verify its provenance, as well as access the quantification of the product's impact on the environment.

We believe the trend is for consumers to choose the most environmentally conscious products, and, over time, this will lead to a paradigm shift with a big impact on how we look for and purchase the products we need. Similarly, organizations would also benefit from this model, since if their production is the most environmentally efficient, in addition to contributing to the well-being of the planet, they may be also contributing to their sales and their image to consumers.

For computing the environmental impact of producing a given product, and since not all environmental costs can be accounted for by the same metric, it is necessary to use equivalence tables, which convert to CO_2e the greenhouse gases generated by the consumption of certain materials. The CO_2e is calculated based on the global warming potential for the mixture and amount of greenhouse gases produced by processing a particular material. This metric gives us a standard and comparable way of accounting for the impact of all production activities, regardless of the materials used, and eventually reach an absolute value that can be used for benchmarking. This is called the carbon footprint of the product, and it is a value that fluctuates from month to month, from production activity to production activity, being recalculated and indexed to each specific batch/lot of articles produced.

To calculate the carbon footprint of a product, first we need to know the activities involved in its creation (production), the products used in its production and their corresponding carbon footprint. For example, suppose you want to make a yogurt cake. This cake contains yogurt, eggs, flour, sugar, etc.. In turn, yogurt needs milk, strawberry, etc.. Each involved product has its own carbon footprint. This may be represented as a directed graph, or tree, and to calculate the CO_2e it is necessary to know the CO_2e of each preceding node that leads to a given node. After production, it is necessary to also account for the carbon footprint that will still be added to it in its supply chain (transport, storage, etc.).

The proposed platform allows the consumer to trace the production chain of a particular product, verify the product's origin, and verify how its carbon footprint increments at each node of its value chain.

The next subsection presents the domain model for the Smart Contract created to trace the carbon footprint. A Smart Contract is "a digital contract that controls user's digital assets, formulating the participant's rights and obligations" (Lin and Liao, 2017). Smart contracts are programs executed in the blockchain to manage, secure and transfer digital assets (Bragagnolo et al., 2018). Smart contracts define a data structure and the operations used to interact with these data, much like classes in object-oriented paradigm (Watanabe et al., 2016).

3.1 Smart Contract's Domain Model

A Product carbon footprint is the CO_2e value of producing a given output product from several input products through a given Monthly Activity or sequence of monthly activities (see Figure 1) .

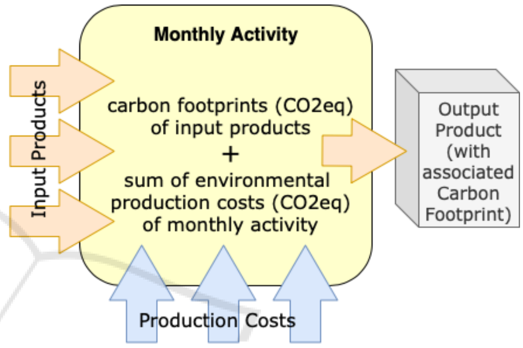

Figure 1: An output product's monthly activity.

The domain model for tracing the environmental impact, in the form of CO_2e value emissions, of products and organizations is depicted in Figure 2. The reading of the domain model goes like this. An Organization has multiple users and produces multiple products. A product from an organization has carbon footprints, which are calculated monthly based on the activities that produced it. A monthly activity uses raw materials and other products in different quantities and produces a certain quantity of one (final or intermediate) output product. Intermediate products are not targeted for the final consumer, as is. Instead, they will serve as input to a monthly activity in the same or other organization, which will produce another final or intermediate output product.

Carbon Footprints of a given product refer to a given product's footprints over a period of different months. Note that a product per se does not have an associated CO_2e value, the value used to account for the environmental impact of the product. This value is calculated monthly on the entity ProductFootprint. This way, a product can improve or worsen its carbon footprint from one month to the next. In reality, the CO_2e values associated to a product (its ProductFootprint associated entity instances) are as many as the monthly activities registered to produce that product each month. This allows for a company to actively

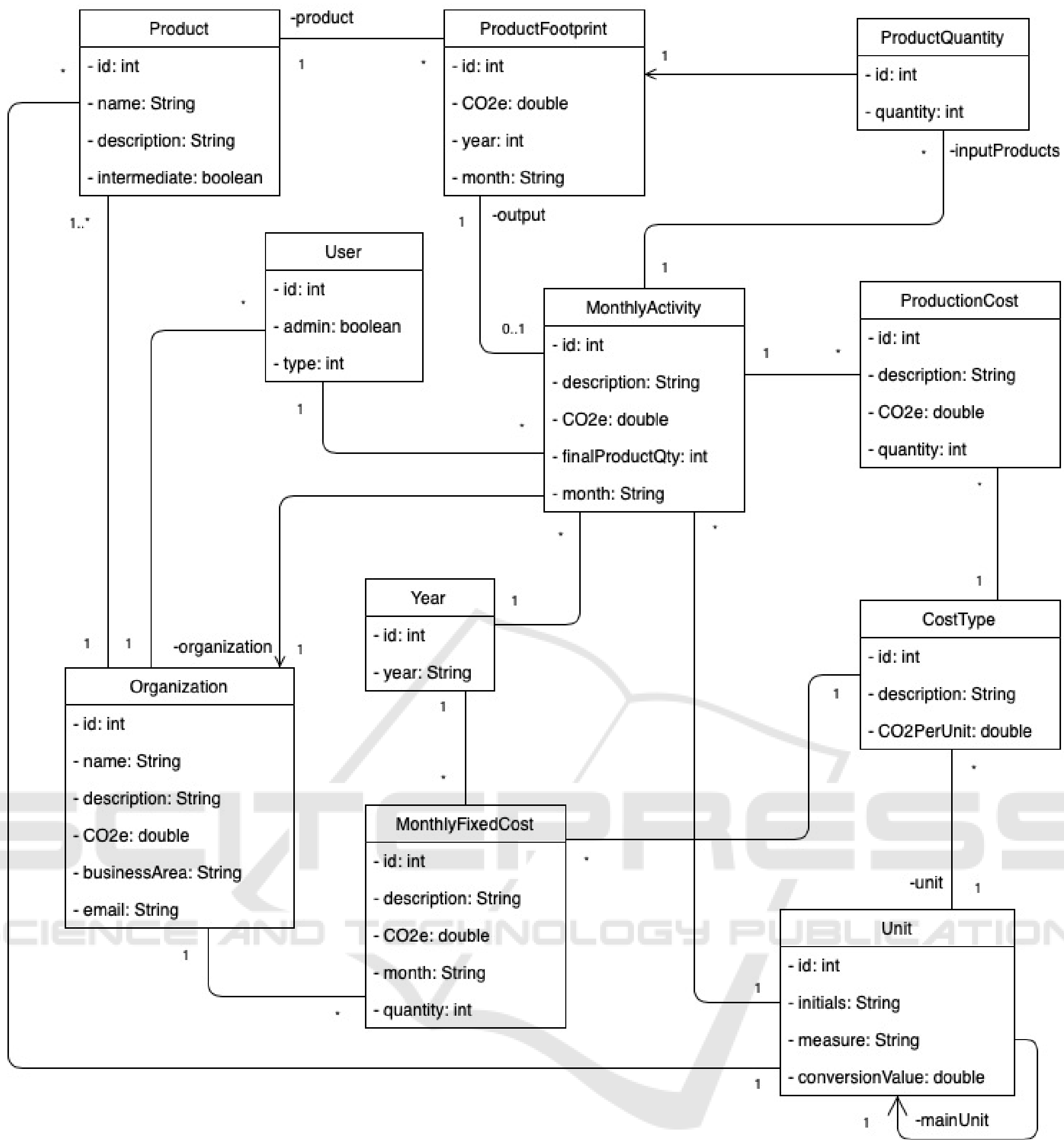

Figure 2: Contract domain Model.

seek to improve its products' carbon footprints, and thus its own company-wide carbon footprint, by optimizing processes for trying to reduce the carbon footprint associated to its monthly fixed costs (e.g. energy, heating, water) or the carbon footprint of the monthly activities to produce the chain of intermediate products that leads to the production of an end product.

MonthlyActivity (Figure 1) and MonthlyFixCost are, then, the entities that register CO_2e values relating, respectively, to the monthly activities that allow producing a product, each month, and to the company's fixed costs, not directly related to a product's production process, but that are company-wide environmental costs indirectly associated to the company's production activities, and thus must be divided by all its produced products.

4 SMART CONTRACT IMPLEMENTATION

4.1 Solidity Smart Contract

Solidity is a high-level programming language used for implementing smart contracts on several blockchain platforms, including in Ethereum (Bragagnolo et al., 2018). It is an object-oriented programming language whose syntax is similar to JavaScript.

Solidity has the possibility of creating data structures that go beyond basic data types, including the possibility of value mappings and the inheritance between contracts. A contract's state variables are permanently stored in contract storage, hence in the Ethereum blockchain.

Solidity contracts will, after their publication, run on the Ethereum Virtual Machine (EVM) and can be invoked via their blockchain addresses.

Solidity implementation of the domain model presented before, needs that each entity is defined as a structure (*struct*). The main structures are listed next:

```
pragma solidity >=0.4.2;

contract CarbonFootPrint {
//-- Structures - Entities Implementation --
struct Product{
   uint32 id;
   string name;
   string description;
   bool intermediate;
   uint32 idOrganization; //ref to Org.
   uint32 idUnit;  //reference to Unit
   uint32[] productFootPrints;
     //refs to ProductFootprint
}

struct ProductFootprint{
   uint32 id;
   uint32 co2eq; // base
   uint16 exp;   // exponent
   uint32 idProduct;
   uint32 year;
   string month;
   uint32 idMA; //ref to MonthlyActivity
}

struct MonthlyActivity{
   uint32 id;
   string description;
   uint32 co2e;
   uint16 exp;
   string month;
   uint32[] productQuantities;
     //refs to input prod quantities
   uint32 output; //ref to output Prod FootPrint
   uint32 finalProductQty;
   uint32 idOrganization;
   uint32 idUnit;
   uint32 idYear;
   address idUser;
   unit32[] productionCosts;
}

struct MonthlyFixedCost{
   uint32 id;
   uint32 co2e;
   uint16 exp;
   string description;
   uint32 quantity;
   string month;
   uint32 idCostType;
   uint32 idOrganization;
   uint32 idYear;
}
```

Unsigned integers (*uint*16, *uint*32) have been used for every numeric type, because Solidity currently does not support signed integers (*int*). It also does not support floating point numbers, such as *float* or *double*. This way, every decimal value is implemented through a base integer value and an exponent. See, for instance, the CO2e attribute in the *Product Footprint Entity*, which contains a product's carbon footprint CO_2e value, and is implemented through `uint32 co2eq; uint16 exp;` in the corresponding contract *struct*.

The contract's state variables are defined through mappings, which allow the recording of data collections in the blockchain.

```
mapping(uint32 => Product) public products;
uint32 public productsCount;

mapping(uint32 => MonthlyActivity)
    public mactivities;
uint32 public mactivitiesCount;

mapping(uint32 => Organization)
    public organizations;
uint32 public organizationsCount;

mapping(uint32 => MonthlyFixCost)
    public mfixcosts;
uint32 public mfixcostsCount;

mapping(uint32 => Unit) public units;
uint32 public unitsCount;

mapping(uint32 => ProductCost)
    public productCosts;
uint32 public productCostsCount;

mapping(uint32 => CostType)
    public costsTypes;
uint32 public costsTypesCount;

mapping(uint32 => ProductFootprint)
    public productFootPrints;
uint32 public pfootPrintCount;

mapping(uint32 => ProductQuantity)
    public productsQuantities;
uint32 public productsQuantitiesCount;
```

An *Event* may be defined in the smart contract for being emitted, or fired, by some function. Fired events are stored in logs by Ethereum and may be captured by the distributed application code interfacing with the smart contract for triggering some action at the user interface level.

```
event registUserEvent (
    address indexed _candidateAddress
);
```

When the contract is issued, it is created through a constructor, much like objects and classes. We defined the following constructor for the *CarbonFootPrint* contract, so that the contract creating user is registered as the first user (admin), and a set of units is created on the units mapping (corresponding to the entity *Unit*), and year 2019 is stored as the first year for monthly activities and fixed costs registration:

```
//- CONSTRUCTOR AND DEFAULT VALUES SETTING
constructor () public{
   users[msg.sender] = User(msg.sender, 0, true);
   arrayUsers.push(msg.sender);

   // -- Initalize units
   addUnit("tonne", "t", 10, 0, 1, false);
   addUnit("kilogram", "kg", 10, 3, 1, true);
   addUnit("gram", "g", 10, 6, 1, true);
   addUnit("milligram", "mg", 10, 9, 1, true);
}
```

A particularity of this type of contracts is that their deployment always has costs associated with being implemented in Ethereum. Whenever a query operation is performed, a certain amount of gas/ether is deducted from the requesting user's account. There are, however, some mechanisms that must be implemented so that, if an error occurs, that user is not discounted the full amount for something that may not be effective in the blockchain.

The contract has functions for managing allowed users, such as verifying if a user is registered for using the contract and creating a new user (*addUser*):

```
function addUser (address _userResp,
          address _user,  uint16 _tipo,
          uint32 _organization) public {

    require(
        users[_user].user_add == address(0),
        "User already registered"
    );

    if(_tipo == 0 || _tipo == 1){
        require(users[_userResp].tipo == 0,
                "You need admin permissions");
    }else if(_tipo == 2){
        require(users[_userResp].tipo == 1 ,
         "You need organization admin
          permissions");
    }

    users[_user] = User(_user, _tipo, true);
    arrayUsers.push(_user);
    userOrganizations[_user].
          push(_organization);
    emit registUserEvent(_user);
}
```

Function *require*() serves for validation and, when failing, returns an *opcode* and effectively reverses the transaction and returns to the user the gas (the cost of the operation) that has not yet been spent [1].

Other functions for managing users include blocking and unblocking a user account.

The contract also allows for registering organizations, of which carbon footprints must be traced/monitored. And, of course, the contract has functions for registering new products:

```
// -- Add New Product Function --
function addProduct(string memory _name,
    string memory _description,
    bool _intermediate,
    uint32 _org,
    uint32 _unit,
    uint32[] memory _footPrints) public
    {

    bool exist = false;
    require(users[msg.sender].idOrganization == _org,
        "You need to belong to the organization");
    require(organizations[_org].id != uint32(0),
        "Organization doesn't exist");

    for(uint32 i=1; i <= productsCount; i++){
        string memory name = products[i].name;
        if(keccak256(abi.encodePacked(name)) ==
          keccak256(abi.encodePacked(_name))){
            exist = true;
        }
    }

    require(!exist, "Product already registered");

    productsCount++;
    products[productsCount] =
      Product(productsCount, _name,
        _description, units[_unit].initials,
        _intermediate, _org, _unit, _footPrints);
    organizations[_org].products.push(productsCount);
}
```

Another important function is the one that allow to create information for a product's carbon footprint, an organization's monthly fixed cost, and a new monthly activity for producing a given output product:

```
// -- Add New Product Footprint Function --
function addFootPrintProd(uint32 _co2eq,
    uint16 _exp, uint32 _idProd,
    uint32 _year, string memory _month,
    uint32 _idMa) public {

    require(users[msg.sender].idOrganization
        == products[_idProd].idOrganization,
        "The product doesnt belong
```

[1] https://solidity.readthedocs.io/en/v0.6.0/

```
        to your organization");

    require(products[_idProd].id != uint32(0),
        "Product doesn't exist");

    pfootPrintCount++;
    productFootPrints[pfootPrintCount] =
        ProductFootprint(pfootPrintCount, _co2eq,
            _exp, _idProd, _year, _month, _idMa);
    products[_idProd].productFootPrints.push(
        pfootPrintCount);
}
```

The full contract and the whole project sources can be found at: https://github.com/xicosantos98/CarbonFootPrint.

4.2 Smart Contract Validation

For validating the contract, at development phase, a set of test cases have been defined. Before each test case, a test contract instance is deployed on a metamask test Ethereum blockchain:

```
beforeEach('setup contract for each test',
    async function () {
        cfootprintInstance = await
            CarbonFootPrint.new();
        users_count = await
            cfootprintInstance.getUsersCount();
    })
```

This section presents some example test cases. Among other test cases, we need, for instance, to assure that the contract allows to create a new organization:

```
it("allows to create new organization",
     async function() {
    var name = "Lidia e Costa";
    await cfootprintInstance.addOrganization(
        name, [], [], []);
    var organization =
        await cfootprintInstance.organizations(1);
    assert.equal(name,
        organization.name);
})
```

It is possible to create a new product:

```
it("allows to create new product",
        async function(){
    await cfootprintInstance.addProduct(
        "Fabric",
        "10 kg of fabric",
        true, 0, 2, []);
    assert.equal(1, await
        cfootprintInstance.productsCount());
})
```

It is possible to create a new product carbon footprint:

```
it("allows to create new product footprint",
        async function(){
    await cfootprintInstance.addProduct(
        "Fabric",
        "10 kg of fabric",
        true, 0, 2, []);
    await cfootprintInstance.addFootPrintProd(
        324, 3, 1, 0);
    assert.equal(1, await
        cfootprintInstance.pfootPrintCount());
})
```

And, it is possible to create a new product's monthly activity:

```
it("allows to create new monthly activity",
        async function(){
    await cfootprintInstance.addMonthlyActivity(
        "packing", 353, 5000, 3,
        "April", [], 0, 0, 2, 1);
    assert.equal(1, await
        cfootprintInstance.mactivitiesCount());
})
```

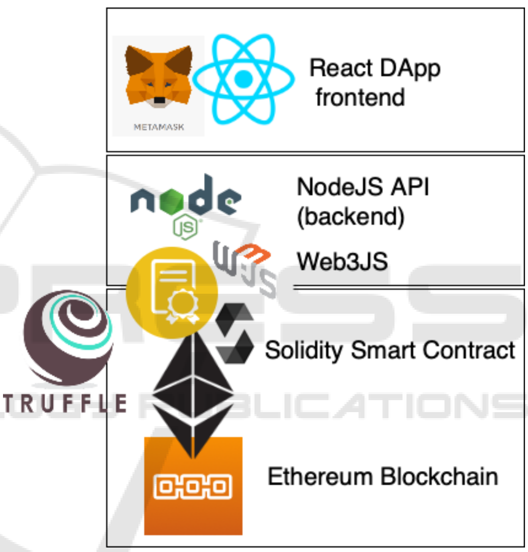

Figure 3: Platform Architecture.

5 DISTRIBUTED APPLICATION

The distributed application (DApp) platform has been implemented in a distributed layered architecture comprised by the layers shown in Figure 3:

- A bottom distributed and decentralized persistence layer, comprised by the developed smart contract on top of the Ethereum Blockchain (the Truffle suite[2] has been used during development, for supplying a local Ethereum development environment and testing framework).
- A NodeJS middleware, which accesses the smart contract through Web3.js libraries[3] and provides

[2]www.trufflesuite.com
[3]web3js.readthedocs.io/en/v1.2.1/

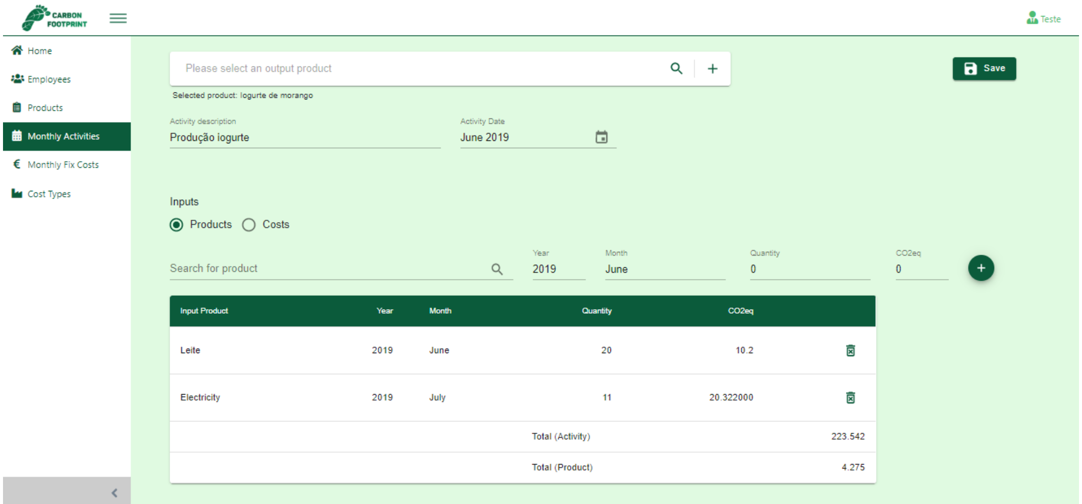

Figure 4: DApp screen for creating a monthly activity.

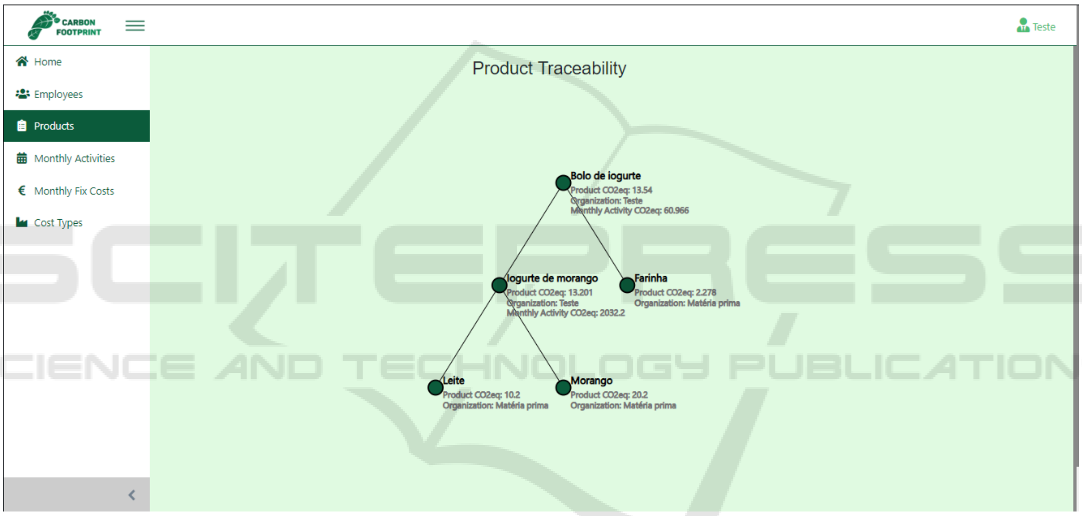

Figure 5: Screen showing the traceability tree for a product's carbon footprint in a given month.

an API that allows to interact with the smart contract on a local or remote Ethereum node.

- A React-based distributed application that runs on Ethereum enabled browsers (e.g.: a browser with a plugin for Ethereum DApps, such as Metamask[4]).

The carbon footprint DApp is based on an Ethereum browser extension for letting an Ethereum user to create and manage their own identities (via private keys, local client wallet and hardware wallets). We used Metamask for this purpose. This way, Carbon Footprint DApp users must authenticate themselves with an Ethereum account, before being able to access the DApp. This ensures a secure interface to access the carbon footprint smart contract on Ethereum.

[4]metamask.io

The DApp allows an organization, which wants to monitor its products' and monthly activities' carbon footprints, to register itself on the platform. A platform *admin* user must confirm each organization's registration. Then, the organization admin may create other organization's user accounts. Each user account is indeed the registration of an Ethereum user on the platform, being given to it a specific user profile on the carbon footprint DApp.

An organization may create products on the platform and, for a given product (output product), monthly activities may be created (Figure 4), which may "consume" input products. A product's carbon footprint is calculated based on the carbon footprint of the monthly activity that creates it in a given month, the carbon footprints of the input products of the monthly activity, and the indirect organizational monthly fixed carbon footprints' in the same month. Any DApp user may access a dashboard with a selection of products' carbon footprint values and, for each product, a traceability graph/tree may be consulted (Figure 5).

6 CONCLUSIONS AND FUTURE WORK

This paper presented the conceptualization, development and validation of a distributed platform application based on the Ethereum blockchain. The platform allows a consumer to follow the chain of production of a product, verify its origin, and have access to the quantification of the carbon footprint that the product has caused in the environment. This way when consumers need to decide which product to buy (or consume) they can decide more consciously, based on valid information (and can choose the most environmentally friendly product). The organizations that produce the traced products also benefit from this platform by improving their reputation for actively improving their effectiveness at the environmental level and for contributing to the well-being of the planet. This will improve their image to consumers and consequently improve their sales.

Some difficulties have been overcome, related to the Solidity data structure limitations, namely the solidity version being used does not support signed integers nor floating point numbers, among other issues.

Future work will address developing a tool for performing data analyzes on the data stored in the smart contract. We hope to be able to answer questions related to which products and types of products have the highest carbon footprint and, within products of the same type, which have the lowest carbon footprint, and whether the footprint is increasing or decreasing over time, etc.

Other future work is related to modifying the smart contract so that, rather than making one extremely long contract, its code could be split across multiple contracts. Solidity offers inheritance to make this possible. Other way of reducing the contract, and the associated transaction costs, is by putting in the contract only the traceability data, moving users, organizations and other data to a database in the cloud.

ACKNOWLEDGEMENTS

The software resulting from this research work has been developed as a curricular project by the second and third authors, and has been partially funded by project ValorMar - Valorização Integral dos Recursos Marinhos - POCI-01-0247-FEDER-024517.

REFERENCES

Biswas, K., Muthukkumarasamy, V., and Tan, W. L. (2017). Blockchain based wine supply chain traceability system. In *Future Technologies Conference (FTC)*.

Bragagnolo, S., Rocha, H., Denker, M., and Ducasse, S. (2018). Smartinspect: Solidity smart contract inspector. In *IWBOSE 2018 - 1st International Workshop on Blockchain Oriented Software Engineering*, Campobasso, Italy.

C3S (2019). Copernicus climate change service (https://climate.copernicus.eu/greenhouse-gases).

CarbonTrust (2019). Climate leadership framework (https://www.carbontrust.com/).

Caro, M. P., Ali, M. S., Vecchio, M., and Giaffreda, R. (2018). Blockchain-based traceability in agri-food supply chain management: A practical implementation. In *2018 IoT Vertical and Topical Summit on Agriculture - Tuscany (IOT Tuscany)*.

Cordero, P. (2013). Carbon footprint estimation for a sustainable improvement of supply chains: State of the art. *Journal of Industrial Engineering and Management*, 6(3):805–813.

Cruz, E. F. and da Cruz, A. M. R. (2019). A food value chain integrated business process and domain models for product traceability and quality monitoring. In *21st International Conference on Enterprise Information Systems (ICEIS)*, pages 285–294, Heraklion, Crete, Greece. INSTICC.

Deloitte (2019). Deloitte's 2019 global blockchain survey - blockchain gets down to business. Technical report, Deloitte.

Dujak, D. and Sajter, D. (2019). *SMART Supply Network*, chapter Blockchain Applications in Supply Chain, pages 21–46. Springer International Publishing AG.

Fu, B., Shu, Z., and Liu, X. (2018). Blockchain enhanced emission trading framework in fashion apparel manufacturing industry. *Sustainability, MDPI, Open Access Journal*, 10(4):1–19.

Galvez, J. F., Mejuto, J., and Simal-Gandara, J. (2018). Future challenges on the use of blockchain for food traceability analysis. *TrAC Trends in Analytical Chemistry*, 107:222 – 232.

Korpela, K., Hallikas, J., and Dahlberg, T. (2017). Digital supply chain transformation toward blockchain integration. In *50th Hawaii International Conference on System Sciences*.

Li, B., Song, Y., and Hu, Z. (2013). Carbon flow tracing method for assessment of demand side carbon emissions obligation. In *TRANSACTIONS ON SUSTAINABLE ENERGY*, volume 4. IEEE Computer Society.

Lin, I.-C. and Liao, T.-C. (2017). A survey of blockchain security issues and challenges. *International Journal of Network Security*, 19(5):P.653–659.

Luu, L., Chu, D.-H., Olickel, H., Saxena, P., and Hobor, A. (2016). Making smart contracts smarter. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, CCS '16, pages 254–269, New York, NY, USA. ACM.

NOAA (2019). National centers for environmental information (https://www.ncdc.noaa.gov/).

Pan, Y., Zhang, X., Wang, Y., Yan, J., Zhou, S., Li, G., and Bao, J. (2018). Application of blockchain in carbon trading. In *10th International Conference on Applied Energy (ICAE2018)*.

Pedersen, A. B., Risius, M., and Beck, R. (2019). A ten-step decision path to determinewhen to use blockchain technologies. *MIS Quarterly Executive*, 18.

Saberi, S., Kouhizadeh, M., Sarkis, J., and Shen, L. (2019). Blockchain technology and its relationships to sustainable supply chain management. *International Journal of Production Research*, 57(7):2117–2135.

Tan, B., Yan, J., Chen, S., and Liu, X. (2019). The impact of blockchain on food supply chain: The case of walmart. In *International Conference on Smart Blockchain*.

Tian, F. (2017). A supply chain traceability system for food safety based on haccp, blockchain amp; internet of things. In *International Conference on Service Systems and Service Management*.

Watanabe, H., Fujimura, S., Nakadaira, A., Miyazaki, Y., Akutsu, A., and Kishigami, J. (2016). Blockchain contract: Securing a blockchain applied to smart contracts. In *2016 IEEE International Conference on Consumer Electronics (ICCE)*, pages 467–468.

Wiedmann, T. and Minx, J. (2008). A definition of 'carbon footprint'. In *C. C. Pertsova, Ecological Economics Research Trends*, chapter 1, pages 1–11. Nova Science Publishers, Hauppauge NY, USA.

Zheng, Z., Xie, S., Dai, H.-N., Chen, X., and Wang, H. (2018). Blockchain challenges and opportunities: a survey. *International Journal of Web and Grid Services*.