applied sciences

MDPI

Article

Impact of Blockchain Technology for Business and Information Systems: Automation of Inter-Company Debt Compensation Case Study

José A. Mateo-Cortés [1], Enrique Arias-Antúnez [2,*] and Diego Cazorla-López [2]

[1] Albacete Research Institute of Informatics, Universidad de Castilla-La Mancha, 02071 Albacete, Spain
[2] Departamento de Sistemas Informáticos, Universidad de Castilla-La Mancha, 02071 Albacete, Spain
* Correspondence: enrique.arias@uclm.es; Tel.: +34-967599200 (ext. 2497)

Abstract: Debt compensation is a process via which various entities (companies, people, etc.) propose the payment of their mutual debts by compensating them according to the credits they may have in their favor. The aim is to establish the financial information of a company so that its debts can somehow be satisfied by its credits, breaking the vicious cycle of debts that cannot be paid. To this end, this financial information is retrieved from the business information system of a company, encompassing how much is owed and to which companies. From the retrieved information, this paper describes the implementation of an automatic system that allows calculating not only the debt cycles involving different companies, but also how these cycles can be effectively solved using blockchain technology thanks to the use of smart contracts. Blockchain technology can have a great impact for companies in this kind of application, guaranteeing the security and traceability of transactions among companies, which are two well-appreciated features from a financial auditing point of view.

Keywords: debt compensation; blockchain; smart contracts; Ethereum; solidity

Citation: Mateo-Cortés, J.A.; Arias-Antúnez, E.; Cazorla-López, D. Impact of Blockchain Technology for Business and Information Systems: Automation of Inter-Company Debt Compensation Case Study. *Appl. Sci.* **2023**, *13*, 4805. https://doi.org/10.3390/app13084805

Academic Editor: Gianluca Lax

Received: 13 February 2023
Revised: 24 March 2023
Accepted: 31 March 2023
Published: 11 April 2023

1. Introduction

Debt compensation is a process via which an affected party proposes the payment of its debt through compensation using the credits it may have in its favor. Debt compensation is used for the benefit of the companies involved, since they can reduce their debts without a direct payment mechanism, thus reducing the need for treasury. Let us note that this process requires the approval of the companies involved. To this end, prior to debt compensation, a determination of the debt cycles among these companies is required, so that a compensation cycle can be established. In the real world, this mechanism is managed by banks or financial entities. In any case, both companies and managerial agencies achieve several advantages. On the one hand, companies can efficiently use their funds, while the risk of insolvency is reduced. On the other hand, banks or financial agencies can attract more clients and make a succulent profit thanks to the fees generated by debt compensation cycles.

It is worthwhile to mention that debt compensation is very important for the finance of companies. Debt cancelation, therefore, resolves a major problem for the economy in general. It is well known that past due invoices are a serious problem for many businesses, especially small and medium-sized enterprises and freelancers, reducing their liquidity and considerably slowing their development. According to [1], 12% of the productive fabric considers its continuity to be at risk due to the impact of non-payments by clients, suggesting that 220,000 companies are at risk of closing in the coming months, which also implies job losses.

Having that in mind, it is very useful not only to "discover" debt compensation cycles (the authors implemented a Dijkstra likewise algorithm [2] to this end), but also to automatize the process of debt compensation among companies, ensuring the result

according to regulations, including auditing regulations and actions. For this purpose, the main contribution of this paper is to present a complete framework using blockchain technology to carry out the debt compensation mechanism semiautomatically.

In this case, semiautomatically indicates that users (business consultancies) only provide the system with input data, whereas they select the cycle(s) to be canceled since this operation cannot be automatically conducted due to Spanish regulations (companies involved in a cycle must legally agree on the terms of the compensation). More detailed information about the workflow is provided in Section 5. As far as the authors know, there is no similar application of blockchain technology and its associated smart contracts.

This paper is structured as follows: Section 1 introduces the problem studied; Section 2 presents some examples of debt compensation and the obtained cycles; Section 3 presents related work about the use of blockchain and smart contracts in this field; Section 4 conceptualizes blockchain and smart contracts, as well as introduces the related technology used in this work; Section 5 presents the proposed solution; Section 6 outlines the conclusions and future work.

2. Debt Compensation: An Example

Debt cancelation algorithms have as data input the debts (mainly invoices) among companies; that is, companies must provide, clearly and precisely, information on how much is owed and to which companies. Thus, the algorithm is able to construct a graph representing the debts among these companies, allowing an evaluation of various debt cancelation cycles. Cancelation is applied for the minimum amount common across the components of such a cycle. Next, we show a graph highlighting the relationships among different companies and a small example of a debt cancelation cycle. Obviously, in the real economy, such transactions can include many more companies, making the problem nontrivial.

Suppose there are three companies, as shown in Figure 1. In this case, company A has a debt (e.g., an invoice) of 1500 EUR with company B, which in the graph is represented by an arrow directed from company A to company B, where the weight is the amount owed (1500 EUR). Likewise, company B has a debt with company C of 4500 EUR and a debt of 500 EUR with company A. As can be seen, this creates a debt cycle where multiple companies owe each other; therefore, a cancelation worth 500 EUR can be established. As mentioned earlier, cancelation cycles are always applied for the lower amount. After compensation, we obtain the new graph shown in Figure 2.

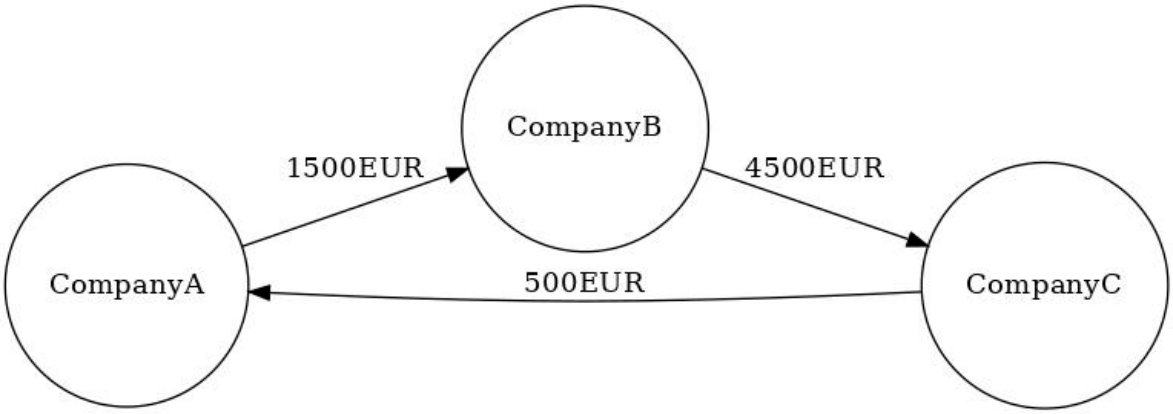

Figure 1. Debt relation among three companies.

Therefore, we started with an example where the initial total debt was 6500 EUR, i.e., the sum of all the debts of the participants. After the cancelation, we can clear 1500 EUR; therefore, the new total debt is 5000 EUR.

As can be seen in the example, circular debt compensation can reduce the liquidity needs of companies, help to solve the problem of non-payment, and improve the balance sheet by facilitating access to credit for companies. In the example, an economic activity

of 6500 EUR is resolved with only 5000 EUR of liquidity. The remainder can be resolved through a multilateral agreement for the exchange of services and/or goods. Note that, for a higher minimum debt between two companies, cancelation becomes more beneficial; therefore, the system contains lower liquidity requirements.

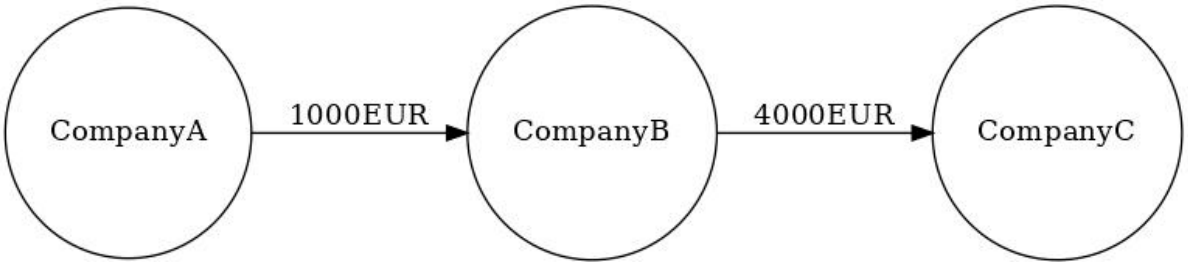

Figure 2. Debt relation among three companies after cancelation.

3. Related Work

To begin with, some related studies regarding the management of debt cancelation in enterprise resource planning systems are presented. First, to the best of our knowledge, Odoo is not endowed with a native system to process debt cycles. By using an extension (an app in Odoo) [3], one can summarize all the debts and credits in just one single transaction, but only for a single company. Moreover, in SAP software, one can use an external system [4] to cope with the problem of debt compensation. Unfortunately, we could not find the information about how this process is conducted and if any state-of-the-art algorithm is used.

On the other hand, blockchain technology is nothing new. In 1991, Haber and Scott Stornetta had the vision of what many people have come to know as blockchain. However, it was in 2008 that the blockchain story began to increase in importance, thanks to the work of Nakamoto [5] related to cryptocurrency, particularly Bitcoin.

Today, nobody disputes the use of blockchain technology in different fields of application [6] such as business intelligence [7], healthcare systems [8], or vehicle-to-everything applications [9], mainly thanks to the inherent good features of blockchain architecture and design, providing qualities such as transparency, robustness, auditability, and security [10,11]. However, blockchain extends beyond cryptocurrency. In fact, one of the main applications is in finance, where blockchain technology is applied to business services, settlement of financial assets, prediction markets, and economic transactions [12]. Most financial applications necessitate the avoidance of mediators in transactions. Some examples can be found in the literature [13–17]. In those papers, blockchain technology was considered from a specific point of view, i.e., research, energy consumption, or partial enterprise solutions.

It is well known that there is also huge interest by the European Commission in the development of blockchain technology. Documents from the European Commission (e.g., [18]) demonstrate the significance and attention placed by Europe on blockchain technology. The European Union Blockchain Observatory and Forum specifically showcases the present capabilities and prospects of blockchain. Furthermore, in order to boost innovation with respect to blockchain technology, the EU is enabling both industry and citizens to benefit from blockchain applications. Thus, the European Commission, after a prospective process, has established a set of applications of blockchain technology in different areas such as finance, trade, and the public sector, which go beyond the current use of blockchain technology. On the other hand, these documents [18] warned of the legal implications of blockchain.

More specifically, blockchain technology has great applicability to the financial sector beyond cryptocurrencies. In fact, blockchain technology is becoming an essential part of what is known as FinTech, due to the following technical features of blockchain: decentralized, open, and untampered information, high security in transactions, and intelligent

trading contracts [19]. There are several applications of blockchain technology in finance. Among them, the following stand out [20]: payment and settlement systems, security issuance, clearing and settlement processes, derivative trading, trade repositories, credit reporting agencies, and corporate governance.

Normally, in mainstream finance, permissioned blockchains are the preferred choice for most blockchain applications. This is because such blockchains only allow participants to access the data stored in them. Furthermore, not all participants are authorized to add new transactions to the chain, and their identities are often verified. Since the voters are identifiable, it is relatively simple to implement consensus mechanisms based on majority votes in these chains. Typically, a majority or super-majority of privileged participants must approve each new transaction [21].

Focusing on settlement, it is well known that the main advantage of a blockchain is faster and more flexible settlement. In fact, it is estimated that "for the US corporate debt market yield, the net gains from blockchain are in the range of 1–4 bps" [22]. The work presented here has a strong relationship with settlement, more precisely, with debt compensation among companies. In fact, late payments in commercial transactions continue to be a pressing problem for small and medium-sized enterprises [23]. In the European Union, they increase financial costs, contribute to illiquidity, and can even lead to bankruptcy. In [23], the authors combined the properties of distributed ledgers with smart contracts leading to harmonization of redress procedures. Such a system improves the standard of proof required to issue a European Order of Payment by providing formal control of the evidence.

Although blockchain has potential in finance, there are several challenges that need to be addressed, such as the following [24,25]:

- Technical performance: the current blockchain systems on the market may not be able to meet the demands of actual businesses with issues in data processing throughput and verification speed, data storage, and node synchronization.
- Security risks: information security and encryption algorithm security are not addressed here. Relevant articles in this area can be found in the literature [7–9].
- Financial regulation: this topic is also outside of the scope of the work presented here. A good example of work in this area can be found in [26].

As a blockchain infrastructure specially built for blockchain in finance, DeFi (decentralized finance) can be highlighted [27]. Decentralized finance is a term used to describe a different financial system constructed on blockchain. The system utilizes smart contracts to design protocols that duplicate existing financial services in a more transparent, interoperable, and accessible way.

On the basis of previous state-of-the-art articles, in this paper, we transfer all the knowledge in blockchain acquired through our research in the enterprise world dealing with a very common problem: debt compensation. Moreover, blockchain technology has been naturally incorporated into company information systems, firstly retrieving debt information from all companies involved in the process to calculate debt cycles, and then enabling automatic debt cancelation thanks to the use of smart contracts, according to Spanish regulations. The problem of energy consumption is not addressed in this work since the energy cost of a private permissioned blockchain is much lower than that of a public blockchain. As far as the authors know, there is no similar application in the literature.

4. Conceptualizing Blockchain Technology

In this section, we reiterate some important concepts for a better understanding of this work.

4.1. Blockchain Technologies

The recent years have seen a lot of interest in blockchain technology across various industries [9,18,28] and sectors [3,29–31]. Blockchain is a decentralized, secure, and tamper-

proof ledger that records transactions and tracks assets through P2P connections. It operates on the basis of a consensus protocol that ensures transaction validity and security, and its immutability is one of its key features [32].

The term blockchain refers to the structure of the ledger. It consists of groups of transactions, organized and stored in blocks, which are chronologically ordered and linked by a block number (hash) and a proposer or validator. The participants in a blockchain network depend on the type of blockchain: public or permissionless blockchains allow anyone to participate, and all participants have equal roles, while private or permissioned blockchains are controlled by a limited number of entities that manage the chain, grant permissions, propose transactions, and validate blocks. There are also hybrid blockchain models. In private and hybrid blockchains, the roles and interactions of participants are defined by smart contracts, which are also stored on the blockchain and known to all participants.

The rise of Bitcoin has led to the creation of various blockchain environments. Currently, Ethereum [33–35], Cardano [36], and Hyperledger [37,38] are the dominant technologies in the market for decentralized application development based on smart contracts. Ethereum is a global, open-source, permissionless platform for decentralized applications that was launched in 2015, becoming the world's first blockchain for DApp development. Hyperledger, hosted by the Linux Foundation, is a growing open-source community that aims to advance blockchain technology for cross-industry use in business. With over 250 members worldwide, it is the fastest-growing project in Linux Foundation history. IBM has adopted Hyperledger as its blockchain technology, offering Hyperledger Fabric [37], which provides a framework for developing blockchain solutions with a modular architecture, pluggable implementations, and container technology. This allows for the development of applications as components that build a final product, while also offering confidentiality, scalability, and security for enterprise environments. This is important for companies who do not want their transactions to be made public.

4.2. Smart Contracts

In this section, the concept of smart contracts is introduced. We consider when the procedure of a debt compensation cycle should trigger a smart contract to make this compensation effective.

To grasp the functioning of a blockchain, it is essential to comprehend the following four crucial concepts:

- Shared ledger: An unalterable record of all network transactions accessible to all network participants. By having a shared ledger, transactions are recorded only once, eliminating the duplicative efforts prevalent in traditional business networks.
- Permissions: As previously mentioned, blockchains can be either permissioned or permissionless. In a permissioned blockchain, each participant holds a distinct identity, which enables the implementation of policies to regulate network participation and access to transaction information.
- Consensus: In a business network where the participants are familiar and trustworthy, transactions can be verified and recorded on the ledger through a consensus mechanism (an agreement).
- A smart contract is a self-executing agreement or set of rules that govern a business transaction, which is stored on the blockchain and automatically executed as part of the transaction.

The concept of smart contracts was first introduced in the 1990s, but it was not until the rise in cryptocurrencies and the implementation of blockchain technology that it became a reality. They play a crucial role in blockchain technology as they dictate and regulate the conduct of transactions. Unlike traditional contracts, smart contracts are integrated into a blockchain network, providing a secure and automated environment for execution. The transparency and self-executing nature of smart contracts is a major advantage; once deployed on the blockchain, they cannot be altered or halted. Furthermore, as mentioned

previously, they are special programs stored on the blockchain, which is a neutral territory outside of any server. The main advantage of this approach is the absence of an intermediary; that is, when the code is deployed in the chain, it is automatically executed where the conditions are met. Thus, they define the agreements among multiple parties without an intermediary. Obviously, the testing phase of this approach is crucial since, for example, the system can incorrectly transfer money to some accounts. In general, these contracts are usually based on the Ethereum network since their integration is mature enough.

Therefore, smart contracts are computer programs that automate the implementation of agreements among parties, eliminating the need for manual coordination or intervention [39,40]. These agreements can be recorded and validated on a blockchain, which can then execute the contract automatically on the basis of predefined "if–then" instructions. For example, "if" a certain condition is met (such as paying for a rental car and short-term insurance), "then" certain actions are triggered (such as the car door unlocking, and payment being transferred). The blockchain's verification and recording process ensures that conflicts or errors are resolved and only one valid transaction is recorded (preventing double entries).

In this way, these smart contracts can be used in a wide variety of scenarios. They can be used to carry out simple transactions with a set of requirements that are verifiable by the network, either within the blockchain itself or outside it using APIs. Today, the main application of this approach is for all kinds of financial products, from funds to call options to many others. For example, one can think of a way to manage things such as copyright for an image bank. Every time an image is used in any media, it will be automatically detected, and the author will be paid. Moreover, micro-insurance can be paid out to farmers depending on certain aspects such as rainfall data collected over a period of time, or micro-payments can be made instantly when the conditions for which they were signed are met (e.g., a worker can be paid according to some performance indicators, or the apartment rent is automatically transferred to the landlord if no complaints are made during the month).

4.3. Related Technologies

Next, we present the blockchain technologies used in this work. We leveraged the potential and benefits of a first-class blockchain, Ethereum. This chain was first designed to be a real alternative to Bitcoin for building decentralized applications. As usual, it comes with advantages and disadvantages, making it more suitable for one scenario than others. The founders (including Vitalik Buterin) advocate for the use of blockchains in a wide range of decentralized applications, not only for money management (cryptocurrencies). These applications can leverage the fast development time, security, and very efficient interoperability of Ethereum as introduced by Vitalik Buterin in his whitepaper. Ethereum achieves this by building the ultimate abstract foundational layer, i.e., an open-source blockchain with a built-in Turing-complete programming language that endows developers with the ability to write smart contracts and decentralized applications, where they provide their own arbitrary rules for ownership, transaction definitions, and functions to change from one state to another. A bare-bones version of Namecoin [40] can be written in two lines of code, and other protocols such as currencies and reputation systems can be built in under 20 lines [36]. Smart contracts can also be built on top of the platform, with vastly more power than offered by Bitcoin scripting because of the added powers of Turing completeness, value awareness, blockchain awareness, and state. In Ethereum, the state of the system is composed of objects called "accounts", with each account having 42 hexadecimal characters, with address and state transitions being a direct relocation of values and information between accounts. Moreover, "Ether" is the main internal crypto fuel of Ethereum and is used to pay transaction fees.

To cope with smart contracts, we used Solidity. Solidity is a programming language not designed to create normal programs; it is a language specifically created for programming smart contracts. Its syntax is based on ECMAScript, similar to other languages such as

JavaScript and C, but with the difference of implementing strong typing when declaring the types of variables and arguments. This is to ensure contract rigor. This similarity with other widely used languages reduces the learning curve since its syntax is quite familiar to the developers. This language can compile and link the contracts created in the Ethereum network code. Solidity was created in 2014 by different collaborators of the Ethereum Project. Specifically, it is a language created to run on Ethereum virtual machines (EVMs) that run on the Ethereum blockchain.

Let us remark that we did not publish to the main net of Ethereum due to current high gas fees (see Figure 3, where ~50 EUR was the total gas fee at the time of writing); however, it is part of Ethereum's roadmap to reduce these fees to make them more affordable for everyone. This cost is high if one considers that a small case study was conducted. In contrast, we used a well-known framework to simulate the Ethereum virtual machine (EVM). Truffle [19] is a world-renowned environment to develop, test, and deploy decentralized applications. It acts as a pipeline for code using the Ethereum virtual machine, with the aim of making a developer's life easier. In Truffle, one can find the following:

- Built-in support for smart contract compilation, linking, deployment, and binary management.
- A mature and automatic testing environment.
- Scriptable, extensible deployment, and migration framework.
- Seamless deployment to any number of public and private networks.
- Complete software for package management (EthPM) and NPM, using the ERC190 standard.
- User console for contract communication.
- Configurable channel with support for strong integration.
- External script executor that runs code within a Truffle environment.

Figure 3. Example of gas fees for deploying a set of contracts in the main Ethereum network.

In addition to Truffle, Ganache [41] is a particular blockchain for quick Corda and Ethereum distributed application development. Developers can use Ganache across the entire development lifecycle. It enables the development, deployment, and testing of DApps in a safe and deterministic environment. Moreover, Ganache can be used in two ways: through a user interface (UI) and through the command line interface (CLI). The Ganache UI is devoted to being used as a desktop application that supports both Ethereum and Corda. Furthermore, the command-line tool, called ganache-cli (formerly known as TestRPC), is also available for development purposes.

5. Automation of Debt Compensation Process by Means of Smart Contracts

In this section, the framework for the process of compensating for a set of invoices obtained from the debt graph is presented. Figure 4 depicts the flow of the entire process. Here, the gear icon represents the system, and the user icon represents the operator. Let us remark that fully automatic compensation is not possible in Spain since, in the final step of

the workflow (choosing the cycles to be compensated), it is mandatory that the involved parties sign the legal allowance for the cancelation of the corresponding debts. Thus, the framework automatically obtains the set of possible cycles that can be canceled, but the tool operator (normally, a consultant) must decide the specific cycles. Then, just by clicking a button (see Figure 11), the framework automatically creates and processes the transactions in the blockchain. As can be observed in Figure 4, the process starts when a consultant extracts a set of invoices from the enterprise resource planning software as a csv file. In our case, IPI Conocimiento y Flexibilidad uses Odoo. Then, the file is uploaded to an Azure storage account. The framework automatically downloads the file and anonymizes it, as can be seen in Figure 4. Once processed, the operator can upload it (see Figure 5), and it is automatically sent to a backend process that calculates the possible cycles. When they are ready, the operator can proceed with the compensation (after all companies legally agree to it). Lastly, all the transactions are created and processed in Ethereum so that the companies have an immutable record of their transactions in the system.

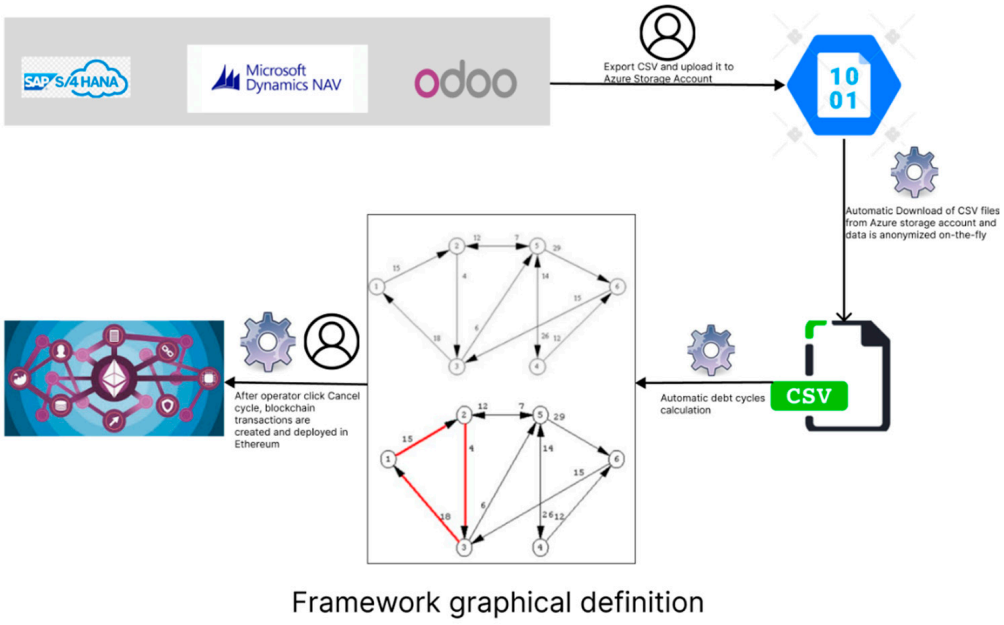

Figure 4. Complete system workflow.

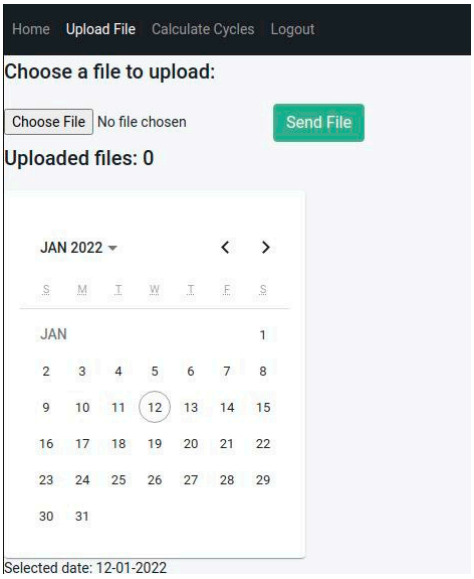

Figure 5. Upload file screen.

Next, the algorithm (see Algorithm 1) for cycle finding and the smart contract for debt cancelation are presented. In this tool, a smart contract is used to mimic the entities involved in the process of debt cancelation (e.g., companies). Listing 1 depicts that a company has a mapping, where a list of company addresses and their corresponding debts is simulated; that is, this data structure simulates a database for each company where its debtors are stored. Moreover, two functions are defined. The first is used to add invoices to the company mapping, i.e., this function is invoked when reading the input csv file that mimics the debt graph so that the debts are updated accordingly. We omit here the screenshot of the graphical user interface implemented in the tool to invoke this method. The second one is invoked to cancel a specific invoice. Let us remark that when the "cancel cycle button" is pressed (see Figure 11), it is run for each row in the screen. In Figure 10, the three transactions automatically created for the cycle of Figure 11 can be observed.

On the other hand, the algorithm is based on Dijkstra's shortest path between two cities (in this case, companies). The main differences are summarized as follows: in this case, the matrices are not symmetric since a directed graph is provided, whereby the debt from company A to B is not normally the same as the debt from company B to A. This slight variation of Dijkstra's algorithm is not a contribution of the article, but an inevitable step in the overall context of debt cancelation.

Moreover, the paths under study are from one company to itself in order to discover cycles involving it. The algorithm has some data structures to store the current minimum debt from one company to another, the path between two companies, and a visited array to avoid visiting a company twice in the same path. Moreover, in the *while* loop, the algorithm studies all the companies in the problem, whereas *for* loops are used to calculate the minimum neighbor company and the minimum debt. Let us remark that we check if one of the neighbors of the company under consideration is the company indeed.

Listing 1. Solidity contract of company entity.

```
contract Company {
    uint16 cif;
    address payable owner;
    //For each company (address), we
      annotate the // debt (invoice)
    mapping (address => uint24) public
      debts;
    function addInvoice (address addr, uint24
      amnts) public returns (uint24) {
        require (amnts > 0, "Amount must
            be greater than 0");
        debts [addr] = debts [addr] + amnts;
        return debts [addr];
    }
    function cancelInvoice (address addr,
         uint24 amnts) public returns (uint24)
        {
        require (amnts > 0, "Amount must
             be greater than 0");
        require ((debts [addr] − amnts) >=
                0, "The amount cannot be
             cancelled since it is greater
             than the current debt.");
        debts [addr] = debts [addr] − amnts;
        return debts [addr];
    }
}
```

It is important to note that the authors implemented many auxiliary functions such as csv processing, anonymizing companies, and back-end services to ship the graph debt to the algorithm, but this material is not included in this works since it is outside of the scope of the paper and their contribution is limited.

Below, the graphical tool the authors developed in order to obtain a set of smart contracts instances starting from an input debt graph is presented. This chart is provided as a set of edges showing the creditor, the debtor, and the amount of the debt. These data are given by an external entity, normally business consultancies, since they have all the invoices of their customers. Figure 5 shows the input screen, where the user must provide the input file in csv format (representing the debt graph), and then choose the date for debt calculation.

This is important since the algorithm must take into consideration the due date of the invoice. Thus, only past due invoices with respect to the date provided by the user are considered. Let us remark that this tool was conceived with business consultancies in mind as they can receive succulent fees for advising debt compensation to their customers. Thus, the tool was developed in collaboration with the company IPI: Conocimiento y Flexibilidad. Once the input data are obtained, normally an Excel or text file, they are anonymized for privacy issues. This task is performed through bash scripts. When this process ends, our algorithm is executed to obtain the debt cycles in the graph (see Figure 6). The output is a text file with the current cycles, as well as the amount of debt that can be compensated. This text file is shown in the tool as HTML tables so that the user can check different cycles and make decisions accordingly (see also Figure 11).

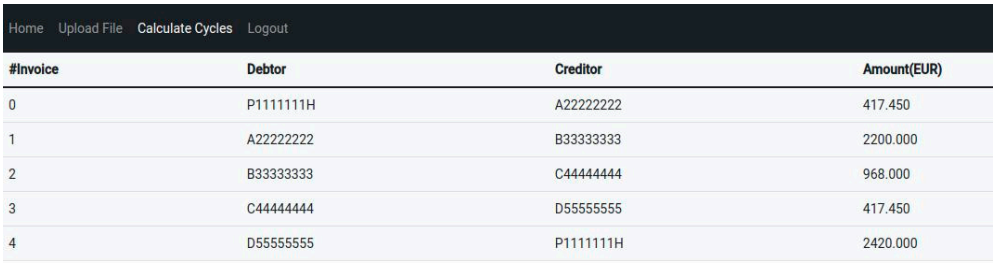

Figure 6. Example of debt cycle.

The required logic to create the stakeholders in the cycle compensation is included in the tool. Thus, companies are created inside the application by means of forms. In future versions, we can consider an automatic process to extract information from external platforms such as Metamask [42], integrating our tool so that companies can attach their ledger in the platform without any human interaction.

In Figure 7, we show the input fields for company creation. As can be seen, participants must provide their name, CIF, and Ethereum address. CIF is the Spanish VAT number of a company, which allows it to be identified for taxation.

Figure 7. Company creation example.

Algorithm 1: Algorithm for finding debt cycles between companies.

```
Data: A set of invoices between companies
Result: A set of cycles that can be compensated
debts ← ∞
path ← −1
visited ← 0
debts [origin] = 0;
path [0] = origin;
min = origin;
all_visited = 1;
while all_visited < tam do
  for d = 1; d < tam; d++ do
    if exists_invoice(minimum, d) and
    invoice_qty(minimum, d) <
    current_debt(minimum, d) and
    visited(d) = false then
      debt(minimum, d) =
        invoice(minimum, d)
      if invoice_qty(minimum, d) <
        minimum_debt then
          minimum_debt =
            invoice_qty(minimum, d)
       end
      end
    end
    new_minimum = 1;
    for d = 2; d < tam; d++ do
    if invoice_qty(minimum, d) <
      invoice_qty(new_minimum, d) then
        if d == origin then
            cycle exists = true;
            visited [origin] = 1;
            all_visited = tam;
            d = tam;
        end
        new_minimum = d;
        end
    end
end
```

Once this information is provided, after clicking "proceed", an instance of a Solidity smart contract named "company" is created, and the corresponding transaction is deployed in the chain (see Figure 8).

Figure 8. Transaction for company creation in Ganache.

Moreover, a new block is added to the blockchain (see Figure 9). This logic is internally provided using the Web3 library.

Figure 9. Added block after company creation in Ganache.

Each company has a Solidity smart contract with three variables: the name of the company, its CIF, and a mapping from the Ethereum address and debts, i.e., an array where the indices are the debtor's address, and the value is the amount of debt in EUR. This mapping can be seen as a sort of accounting for the debts of a specific company. Let us note that we internally calculate the correspondence between Ethers and EUR.

On the other hand, if the user (normally, the consultant) asks for a cycle compensation, the tool automatically calls the corresponding method of the creditor smart contract, called "cancelInvoice", in order to reduce the amount of debt in the previously presented mapping. This method is invoked for each invoice in the cycle. As previously mentioned, these calls are made using the Web3 library. Figure 10 shows the automatic transactions run on the system, which compensates for the cycle of Figure 11.

Figure 10. Transactions for cycle cancelation.

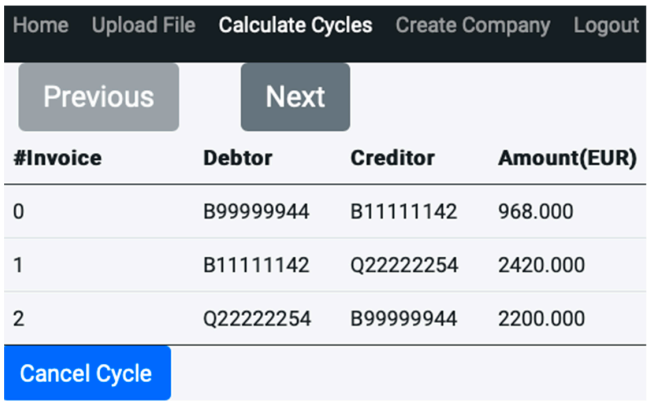

Figure 11. Cycle cancelation example.

6. Conclusions and Future Work

In this paper, a tool for the semiautomatic cancelation of debt cycles among companies was presented. To this end, an algorithm for cycle calculation was implemented, and Solidity smart contracts were established for company models. The Web3 library was used to model the logic of the application with respect to these smart contracts. It is easy to observe the benefits of using blockchain in this type of problem, where conflict resolution and traceability are challenging requirements in such systems. The software developed in this work can be downloaded in the link provided as Supplementary Materials.

As future work, we plan to extend the tool with well-known platforms such as Metamask, and we have deployed an on-premises Ethereum network for deeper testing purposes. In addition, we will deploy the application in other currently available Ethereum test nets. Lastly, let us note that the system was conceived for transferring debt or other kinds of assets among participants since, for instance, one company could be interested in buying another company's debt due to its high reputation. This feature will be included in the next versions of the tool.

In general, the problem dealt in this work is an NP problem from the point of view of computational complexity. However, future work will consist of the implementation of a parallel algorithm to alleviate the limitations of companies that can be treated in a reasonable amount of time. In fact, the authors are also working on a quantum algorithm which will allow us to deal with this problem in the future. In this case, the algorithm to be developed will be executed in a DWave platform. However, other alternatives will be tested.

Supplementary Materials: The software developed for this work has been registered. The registration, software, and documentation can be downloaded from https://www.safecreative.org/work/2211302757747-desarrollo-de-software-de-compensacion-de-deuda. Last accessed on 23 March 2023.

Author Contributions: J.A.M.-C., conceptualization, software, methodology, investigation, validation, and writing—original draft; E.A.-A., conceptualization, methodology, writing—original draft, and supervision; D.C.-L., writing—original draft, validation, and supervision. All authors have read and agreed to the published version of the manuscript.

Funding: This work was carried out with the support of IPI Conocimiento y Flexibilidad S.L. company through the project "Estudio e implementación de algoritmos para la compensación de deuda y su ejecución en una plataforma paralela".

Data Availability Statement: No data used in this paper.

Acknowledgments: The authors would like to thank Raúl Mata Jiménez, IPI Conocimiento y Flexibilidad's for his valuable contribution to this paper. He was crucial in the conceptualization of the problem and supervision of the study.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Estudio de Riesgo de Crédito en España. Available online: https://agentes.creditoycaucion.es/ca/gguijarro/noticias/analisis/detalle/1020-estudio-riesgo3 (accessed on 22 January 2023).
2. Behún, M.; Knežo, D.; Cehlár, M.; Knapčíková, L.; Behúnová, A. Recent Application of Dijkstra's Algorithm in the Process of Production Planning. *Appl. Sci.* **2022**, *12*, 7088. [CrossRef]
3. Odoo Module for Account Payment Netting. Available online: https://apps.odoo.com/apps/modules/12.0/account_payment_netting/ (accessed on 23 March 2023).
4. Technosis Software. Available online: https://technosis.de/ (accessed on 23 March 2023).
5. Nakamoto, S. Bitcoin: A Peer-to-Peer Electronic Cash System. March 2009. Available online: https://bitcoin.org/bitcoin.pdf (accessed on 23 March 2023).
6. Casino, F.; Dasaklis, T.K.; Patsakis, C. A Systematic Literature Review of Blockchain-Based Applications: Current Status, Classification and Open Issues. *Telemat. Inform.* **2019**, *36*, 55–81. [CrossRef]
7. Kumar, R.; Kumar, P.; Jolfaei, A.; Islam, A.K.M.N. An Integrated Framework for Enhancing Security and Privacy in IoT-Based Business Intelligence Applications. In Proceedings of the 2023 IEEE International Conference on Consumer Electronics (ICCE), Las Vegas, NV, USA, 6–8 January 2023; pp. 1–6. [CrossRef]

8. Kumar, R.; Kumar, P.; Tripathi, R.; Gupta, G.P.; Islam, A.K.M.N.; Shorfuzzaman, M. Permissioned Blockchain and Deep Learning for Secure and Efficient Data Sharing in Industrial Healthcare Systems. *IEEE Trans. Ind. Inform.* **2022**, *18*, 8065–8073. [CrossRef]
9. Kumar, R.; Kumar, P.; Tripathi, R.; Gupta, G.P.; Garg, S.; Hassan, M.M. BDTwin: An Integrated Framework for Enhancing Security and Privacy in Cybertwin-Driven Automotive Industrial Internet of Things. *IEEE Internet Things J.* **2022**, *9*, 17110–17119. [CrossRef]
10. Greenspan, G. Ending the Bitcoin vs. Blockchain Debate. Available online: http://www.multichain.com/blog/2015/07/bitcoin-vs-blockchain-debate (accessed on 22 January 2023).
11. Christidis, K.; Devetsikiotis, M. Blockchains and Smart Contracts for the Internet of Things. *IEEE Access* **2016**, *4*, 2292–2303. [CrossRef]
12. Haferkorn, C.M.; Quintana Diaz, J.M. *Seasonality and Interconnectivity within Cryptocurrencies—An Analysis on the Basis of Bitcoin, Litecoin and Namecoin*; Springer International Publishing: Berlin/Heidelberg, Germany, 2015; pp. 106–120.
13. Notheisen, B.; Cholewa, J.B.; Shanmugam, A.P. Trading Real-World Assets on Blockchain. *Bus Inf. Syst. Eng.* **2017**, *59*, 425–440. [CrossRef]
14. Sedlmeir, J.; Buhl, H.U.; Fridgen, G.; Keller, R. The Energy Consumption of Blockchain Technology: Beyond Myth. *Bus. Inf. Syst. Eng.* **2020**, *62*, 599–608. [CrossRef]
15. Ante, L.A. A place next to Satoshi: Foundations of blockchain and cryptocurrency research in business and economics. *Scientometrics* **2020**, *124*, 1305–1333. [CrossRef]
16. Risius, M.; Spohrer, K. A Blockchain Research Framework. *Bus. Inf. Syst. Eng.* **2017**, *59*, 385–409. [CrossRef]
17. Nofer, M.; Gomber, P.; Hinz, O.; Schiereck, D. Blockchain. *Bus. Inf. Syst. Eng.* **2017**, *59*, 183–187. [CrossRef]
18. European Commission. Blockchain Now and Tomorrow: Assessing Multidimensional Impacts of Distributed Ledger Technologies. 2019. Available online: https://ec.europa.eu/jrc/en/facts4eufuture/blockchain-now-and-tomorrow (accessed on 22 January 2023).
19. Truffle Suite. Available online: https://trufflesuite.com/docs/truffle/ (accessed on 22 January 2023).
20. Ali, O.; Ally, M.; Dwivedi, C.Y. The state of play of blockchain technology in the financial services sector: A systematic literature review. *Int. J. Inf. Manag.* **2020**, *54*, 102–199. [CrossRef]
21. Varma, J.R. Blockchain in Finance. *Vikalpa J. Decis. Makers* **2019**, *44*, 1–11. [CrossRef]
22. Chiu, J.; Koeppl, T.V. Blockchain-Based Settlement for Asset Trading. *Rev. Financ. Stud.* **2019**, *32*, 1716–1753. [CrossRef]
23. Crepaldi, M. Distributed ledgers, EOP, and debt recovery mechanisms: A new technology for civil procedure. *Civ. Proced. Rev.* **2018**, *9*, 59–69.
24. Monrat, A.A.A.; Schelen, O.; Andersson, K. A Survey of Blockchain from the Perspectives of Applications, Challenges, and Opportunities. *IEEE Access* **2019**, *7*, 117134–117151. [CrossRef]
25. The European Union Blockchain Observatory Forum, Blockchain Innovation in Europe. 2018. Available online: https://dutchblockchaincoalition.org/assets/images/default/Report-innovation-in-europe-light.pdf (accessed on 22 January 2023).
26. Yeoh, P. Regulatory issues in blockchain technology. *J. Financ. Regul. Compliance* **2017**, *25*, 196208. [CrossRef]
27. Schär, F. *Decentralized Finance: On Blockchain- and Smart Contract-Based Financial Markets. Federal ReServe Bank of St. Louis Review, Second Quarter*; The Federal Reserve Bank of St. Louis: St. Louis, MO, USA, 2021; pp. 153–174. [CrossRef]
28. Blechschmidt, B. Blockchain in Europe: Closing the Strategy Gap. Cognizant Reports. 2018. Available online: https://www.cognizant.com/whitepapers/blockchain-in-europe-closing-the-strategy-gap-codex3320.pdf (accessed on 22 January 2023).
29. Escala, A. *Definiendo un Nuevo Futuro Para la Banca Con Blockchain*; IBM Reports; IBM: Armonk, NY, USA, 2018.
30. Grech, A.; Camirelli, A.F. *Blockchain in Education*; European Commission: Brussels, Belgium, 2017.
31. *European Coordination Committee of the Radiological, Electromedical and Healthcare IT Industry*; Blockchain in Healthcare: Brussels, Belgium, 2017.
32. Allende, M. *Blockchain: Cómo Desarrollar Confianza en Entornos Complejos Para Generar Valor de Impacto Social*; Banco Interamericano de Desarrollo: Washington, DC, USA, 2018.
33. Ethereum. Available online: https://ethereum.org/ (accessed on 23 March 2022).
34. Buterin, V. A Next-Generation Smart Contract and Decentralized Application Platform. Available online: https://ethereum.org/en/whitepaper/ (accessed on 22 January 2023).
35. Wood, G. Ethereum: A secure decentralised generalised transaction ledger; EIP-150 REVISION. *Ethereum Proj. Yellow Pap.* **2014**, *151*, 1–32.
36. Cardano. Available online: https://cardano.org (accessed on 22 January 2023).
37. Hyperledger Fabric. Available online: https://www.ibm.com/blockchain/hyperledger (accessed on 22 January 2023).
38. Hyperledger Technology. Available online: https://www.hyperledger.org/ (accessed on 22 January 2023).
39. Szabo, N. The Idea of Smart Contracts. 1997. Available online: http://www.fon.hum.uva.nl/rob/Courses/InformationInSpeech/CDROM/Literature/LOTwinterschool2006/szabo.best.vwh.net/idea.html (accessed on 22 January 2023).
40. Namecoin. Available online: https://www.namecoin.org (accessed on 22 January 2023).
41. Ganache. Available online: https://trufflesuite.com/docs/ganache/ (accessed on 22 January 2023).
42. Metamask. Available online: https://metamask.io (accessed on 22 January 2023).