Aug 10th, 12:00 AM

Towards a Novel Business Process Model for Food Delivery Services Using Blockchain Technology

Rodrigo Folha
Federal University of Pernambuco, vinigo2@gmail.com

Valéria Cesario Times
Federal University of Pernambuco, vct@cin.ufpe.br

Arthur Carvalho
Miami University, arthur.carvalho@miamioh.edu

André Araújo
Federal University of Alagoas, andre.araujo@penedo.ufal.br

Flaviano Viana
Federal University of Pernambuco, fjlv@cin.ufpe.br

See next page for additional authors

Towards a Novel Business Process Model for Food Delivery Services Using Blockchain Technology

Completed Research

Rodrigo Folha
Federal University of Pernambuco, Brazil
rbf2@cin.ufpe.br

Valéria Times
Federal University of Pernambuco, Brazil
vct@cin.ufpe.br

Arthur Carvalho
Miami University, USA
arthur.carvalho@miamioh.edu

André Araújo
Federal University of Alagoas, Brazil
andre.araujo@ic.ufal.br

Flaviano Viana
Federal University of Pernambuco, Brazil
fjlv@cin.ufpe.br

Henrique Couto
Federal University of Alagoas, Brazil
henrique.couto@ic.ufal.br

Abstract

Demand for food delivery services has grown significantly in recent years and, in particular, during the COVID-19 pandemic and the resulting stay-at-home orders. From a business perspective, food delivery platforms work as intermediaries by mediating interactions among customers, restaurants, and delivery workers. This paper suggests that blockchain technology coupled with smart contracts can help remove centralized food delivery platforms by enabling peer-to-peer interactions among stakeholders. Following the design science research framework, we first conducted interviews with some relevant stakeholders. We use the obtained responses to design requirements related to cost reduction, transparency, and privacy. We thereafter suggest three design principles to tackle the design requirements. Next, we design and instantiate our artifact based on a decentralized business process model and blockchain technology. Finally, we evaluate our solution in terms of its potential for cost reduction and handling conflicts.

Keywords

Blockchain, dispute resolution, food delivery services, smart contracts, privacy.

Introduction

Recent advancements in information and communication technology have enabled customers to easily connect with and remotely place orders at local restaurants. These restaurants, in turn, may have in-house or outsourced workers on call ready to deliver the ordered food. Several platforms are now intermediating the interactions between customers, restaurants, and delivery workers. For example, iFood dominates the Brazilian market, and Uber — the American ride-hailing company — created Uber Eats, a spin-off application focusing on food delivery. A 2018 report by Forbes estimates the gross revenue of online delivery companies to be around 82 billion U.S. dollars, and this value is expected to double by 2025 (Singh, 2019). Moreover, about 11% of the world's population have now access to food delivery platforms (Singh, 2019).

Food delivery companies, such as Grubhub, Uber Eats, and DoorDash, have business models centered around *delivery fees* charged to the customers for the delivery services as well as *commission fees* charged to the restaurants on a per-order basis, which can range from 10% to 30% of an order. Delivery and commission fees place an *extra* burden on both customers and restaurants, many of which currently face a decline in disposable income and sales due to lockdowns throughout the world (Kim et. al., 2021). Food delivery companies also have access and single-handed control over precious data — including food preferences and optimal prices — that are the byproduct of the purchases placed by customers.

From a business perspective, food delivery companies work as intermediaries intermediating interactions among customers, restaurants, and delivery workers. Removing intermediaries is one of the main motives behind the use of *blockchain* technology. In this paper, we propose a blockchain-based solution to remove centralized food delivery platforms. Specifically, we discuss how blockchain coupled with *smart contracts* can help customers place orders straight with restaurants instead of using centralized platforms. Delivery workers can then monitor orders stored on the blockchain and accept the desirable ones. By removing intermediaries, we argue that our solution may bring purchase costs down due to reduced commission fees. Second, our system creates a common and reliable record of transactions, which can help to prevent costly disputes. Finally, our solution enables privacy because stakeholders' true identities are only disclosed to the appropriate parties. In terms of methodology, we follow the *design science research framework* (DSRF) by Peffers et al. (2007). That is illustrated by the organization of the rest of this paper.

Problem Identification and Motivation

The first step in the design science research framework is to both "*define the specific research problem and justify the value of a solution*" (Peffers et al., 2007). To accomplish this step, we first conducted preliminary interviews with 8 restaurant owners and 34 customers to better understand the problems faced by these stakeholders when using food delivery services. All the interviews happened in Recife, Brazil, in 2021. Although there are some nuances in regulations and consumer behavior across different countries, we nonetheless believe the studied business process is rather general. We summarize next the most relevant responses before deriving *design requirements* (DR) to be addressed by an ideal food delivery solution.

We first note that all interviewed restaurant owners said they rely on a third-party food delivery service. Moreover, 62.5% of them considered the platforms underlying the food delivery services to be the main channel to sell products to customers. The restaurant owners reported paying commission fees ranging from 10% to 30% of the cost of each purchase. Consequently, all restaurant owners reported raising the prices of their products on food delivery platforms from somewhere between 10% to 50% to pass on the costs to their customers. On the customer side, 58% of the interviewees reported that passing the costs concerning commission fees to customers is an unfair practice. From the above discussion, one can conclude that costs must be at the forefront of the discussion when considering alternatives to current food delivery platforms. That discussion leads us to formulate the following design requirement:

Design Requirement (DR) #1: *food delivery services must be supported by economically-feasible systems*.

Another focus of the interviews was on the problems (conflicts) faced by different stakeholders during deliveries. For example, some of the mentioned problems include wrong and/or missing items, late deliveries, missing addresses, etc. The restaurant owners informed that about 1% to 5% of the orders face problems and that the delivery worker is responsible for about 75% of them. Nonetheless, in most cases, the restaurants took responsibility for the costs to avoid unfavorable online reviews. It was widely reported by the restaurant owners that the food delivery platforms do not support the restaurants when conflicts arise. In one specific case, a restaurant owner mentioned that a customer disliked the purchased food and reported that feeling to the food delivery platform, which in turn immediately reimbursed the customer and canceled the restaurant's payment. The restaurant owner disagreed with the decision and has since started a legal process to reclaim the money. On the other hand, over 95% of the customers informed that they had already faced problems using food delivery services, and over 80% of them believed the restaurants were responsible for the problem. The above discussion highlights the need for unambiguous and transparent transactions and dispute-resolution rules, which leads us to the following design requirement:

Design Requirement (DR) #2: *food delivery services must be supported by systems that have transparent and standardized transaction and dispute-resolution rules*.

Finally, although this topic was not part of our preliminary interviews, it is nonetheless essential to consider stakeholders' privacy when developing food delivery solutions. For example, customers might not want data about their food preferences or order time to be available to the public. Likewise, restaurants may not feel comfortable having sales data available to potential competitors. This leads us to formulate the DR #3:

Design Requirement (DR) #3: *stakeholders' identifiable data should not be publicly available*.

Defining the Objectives of a Solution

The second step in the design science research framework is to "*infer the objectives of a solution from the problem definition and knowledge of what is possible and feasible*" (Peffers et al., 2007). In this step, we define *design principles* (DP) to overcome the existing challenges brought by the design requirements. As intermediaries, food delivery platforms can charge hefty fees to process food orders. A potential solution to make food delivery services cheaper is to replace those intermediaries with peer-to-peer (P2P) transactions. Based on this observation, we propose the following design principle to address DR #1:

Design Principle (DP) #1: *stakeholders transact with each other in a P2P way to reduce transaction fees.*

Concerning DR #2, it is crucial for any food delivery solution to transparently manage the interactions among the underlying stakeholders, including handling payments and eventual charges. To avoid potential misunderstandings, the underlying transaction rules should be formalized as much as possible. That includes implicitly codifying answers to conflict-related questions such as "what should happen if a certain purchase is never delivered or when the delivered items are wrong?" To remove any ambiguity in the rules, we formulate the following design principle to handle DR #2:

Design Principle (DP) #2: *transaction and dispute-resolution rules are coded as transparent algorithms.*

Finally, DR #3 states that stakeholders might not be comfortable having their private data publicly available. Nonetheless, some private data must be disclosed to get a transaction processed. For example, delivery workers need customers' addresses and names when delivering food. Several cryptographic tools can be used to control access to private data, which leads to our third design principle:

Design Principle (DP) #3: *cryptographic primitives must be used to control access to private data.*

In short, an ideal solution supporting food delivery services is based on three fundamental design principles: *cost reduction* (DP #1), *transparency* (DP #2), and *privacy* (DP #3). It turns out that *blockchain technology* coupled with *smart contracts* can effectively handle all the design principles. In particular, a blockchain-based solution can act as a decentralized authority that manages transactions to eliminate or reduce commission fees. Moreover, smart contracts can explicitly encode all the transaction and dispute-resolution rules. Lastly, cryptographic techniques, such as asymmetric encryption, are already at the core of blockchain. Before elaborating on our blockchain-based solution for food delivery services, we first explain the concept of blockchain and smart contracts next.

Blockchain and Smart Contracts

For our purposes, blockchain is a *distributed*, *decentralized*, and *append-only* database. *Distribution* regards the redundancy created by replicating the same database across several computational devices, also called *nodes*. *Decentralization* means that the nodes are not necessarily controlled by the same entity. Finally, "*append-only*" means that transactions stored on the database cannot be changed and, consequently, the database is tamper-proof. *Nodes* create a P2P network that defines the information technology infrastructure accessed by potential blockchain users.

In terms of applicability, besides financial services, supply chain management has been one of the most promising application areas since blockchain's decentralized and append-only nature allows one to establish the provenance of an asset (Hastig and Sodhi, 2020). Other than supply chain, different blockchain models have been applied to solve a variety of problems, including how to make certain business practices in the video game industry more transparent (Carvalho 2021), during know-your-customer verification processes, in prediction markets (Carvalho 2020), energy management, among other domains.

Beyond transaction data, several blockchain models now allow the nodes to store and execute algorithms — also called *smart contracts* — proposed by users. When that happens, not only the underlying code is immutable, but also is any input and potentially outputs produced after a node executes the code. In other words, smart contracts take the idea of putting data in a secure and decentralized database and extend it to computations. As we discuss in the next section, our proposed solution to support food delivery services consists of users interacting with smart contracts when placing/receiving orders and delivering food. Besides handling the transactions, the smart contracts manage all payments and serve as dispute resolution.

Design and Development

The third step in the DSRF is to design and create the underlying artifact (Peffers et al., 2007). Before explaining our solution, it is essential to understand the business process for traditional food delivery services. Our experience interviewing restaurant owners has enabled us to understand their challenges and routines. Figure 1 shows the studied business process mapped using Business Process Model and Notation (BPMN). For our purposes, the most salient point is that any actor involved in the food purchasing, processing, and delivering operations interacts exclusively with the food delivery platform. Figure 1 shows that all communications are centralized, and a third party (platform) handles all payments and has a monopoly over all generated data. Even when phone calls are necessary, the platform is the entity that forwards calls to the appropriate parties. When that happens, private information, such as phone numbers, is only concealed in specific circumstances after the platform operators evaluate the situation.

Having the three design principles in mind (cost reduction, transparency, and privacy), Figure 2 shows our proposed decentralized business process. Our approach relies on data stored on- and off-chain. For example, data about restaurants' products (*i.e.*, menu items) are stored by the restaurants either using their propriety information technology infrastructure or in public domains/databases. When doing so, restaurants can add constraints regarding delivery, such as the maximum delivery distance and estimated time to prepare each item. Customers can then access a restaurant's menu, select the desired items, and place orders straight with restaurants instead of using food delivery platforms. As we elaborate in the next section, customers place orders with restaurants through a smart contract. Restaurants are then informed of new orders. To avoid having sales data available to all, the customer encrypts all the details concerning the order — such as the purchased items and prices — during the purchase process using the restaurant's public key. Consequently, the restaurant, and only the restaurant, can decrypt the order after retrieving it from the blockchain. Naturally, restaurants may then accept or reject any order.

Through a suitable application, delivery workers can monitor jobs accepted by restaurants and accept the desirable ones. Information about accepted jobs is stored on the blockchain, including timestamps and the driver's info. After the initial customer-restaurant interaction, followed by the restaurant-delivery worker interaction, it is now time for the customer to send private information plus order details to the delivery worker. To do so, the customer uses the public key of the delivery worker to encrypt the customer's private information, such as name, phone number, physical address, and the ordered items. The delivery worker, and only the delivery worker, will be able to decrypt the information reported by the customer. Having that information, the delivery worker receives the order and acknowledges that all the purchased items are correct. That acknowledgment plus a receipt timestamp are stored on the blockchain. Finally, the delivery worker delivers the food, and it is now time for the customer to acknowledge that all the items are correct. Again, a delivery timestamp plus the acknowledgment are stored on the blockchain. As we explain in the Demonstration section, the process of acknowledging the accuracy of the items helps when disputes arise.

The most obvious difference between the above business processes is the removal of the food delivery platform in Figure 2. As a result, payments are made directly between customers, restaurants and delivery people as soon as the delivery is carried out, thus dispensing with intermediation fees ranging from 10% to 30%. Another problem with platforms that intermediate food delivery is that they accumulate the balance of restaurants and delivery people, and pay only once a week, giving the false impression of a high cash flow (Chen et al., 2022). This commission rate suffocates restaurants, such that having more food delivery orders actually hurts the restaurant's profitability (Tkacik, 2020). We argue that our new proposal has many advantages over traditional processes: 1) transaction costs might be lowered by decreasing commission fees (see discussion in the Evaluation section); 2) data democratization in the sense that no single central entity is the owner of the system and has control over all generated data; and 3) sensitive data is protected by design. On the other hand, as we discuss in the Evaluation section, there are new challenges to be addressed, *e.g.*, the most appropriate data governance and blockchain model.

Transactions and Dispute-resolution Rules

We next elaborate on some rules regarding transactions within the system and dispute resolution. When the customer first interacts with the system, a payment must be issued covering the order cost and any delivery fee. Then, after accepting the order, the restaurant also deposits a value through a smart contract covering only the delivery fee. Finally, upon accepting the delivery task, the delivery worker also deposits a

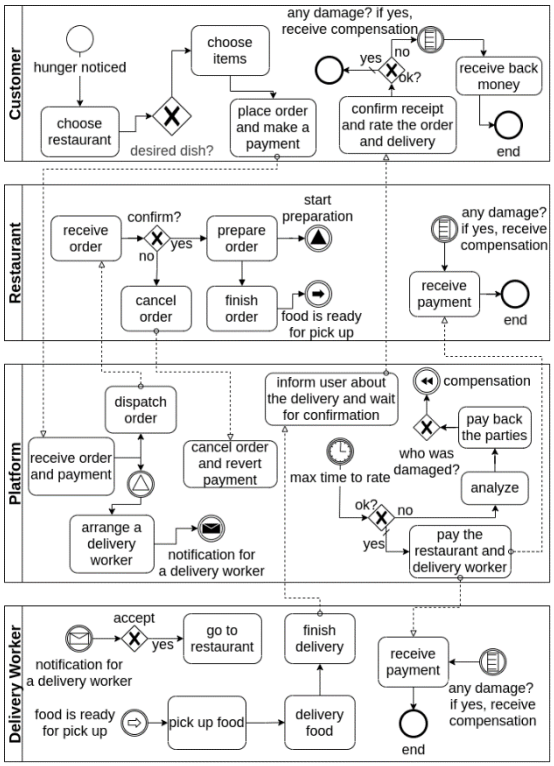

Figure 1. Traditional business process using food delivery platforms.

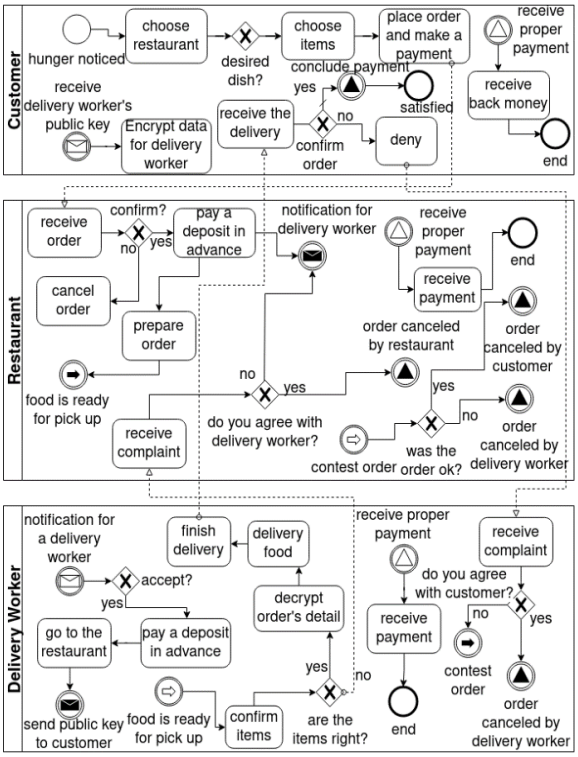

Figure 2. The proposed business process enabled by blockchain technology.

fee to cover the total or a small fraction of the order cost. At this point, the smart contracts behave as escrow accounts waiting for the food delivery process to end to release the deposits.

When the food is delivered successfully, the delivery worker receives his/her deposited amount back plus the delivery fee deposited by the customer. Now, consider the first dispute when the restaurant finishes an order and, while receiving the order, the delivery worker identifies that the ordered items are incorrect. Consequently, the delivery worker does not accept the order. In this case, the restaurant can fix the order, and this scenario returns to the same setting as when the delivery is successful. Alternatively, the restaurant can agree with the delivery worker but cancel the order, which causes the deposited money by the customer to be refunded and the deposited money by the restaurant to be sent to the delivery worker. Finally, if the restaurant disagrees with the delivery worker, then another delivery worker is requested, and the previous delivery worker receives only his/her deposit back.

The next conflict may occur when the customer receives the order. Suppose the customer states the ordered items are inaccurate or damaged, and the delivery worker agrees with the statement. In this case, the delivery worker explicitly accepts responsibility for the problem since the delivery worker did inspect the order with the restaurant before. Subsequently, the customer and the restaurant receive their deposits back, whereas the delivery worker's deposit goes to the restaurant to cover the order costs. Alternatively, when the delivery worker disagrees with the customer, the restaurant then decides whether the order has an issue. If the answer is yes, then the output is similar to what we discussed above, and the blame is on the delivery worker. If the answer is no, then the restaurant disagrees with the customer's assessment, and the resulting outcome is similar to when the delivery is successful. We note that one can always adjust the dispute resolution rule by adding a third option to the business process where the restaurant benevolently takes the loss fully or partially to solve the conflict without over penalizing the delivery worker.

Observe how the chain of acknowledgments alongside the deposits made by the stakeholders cover expenses and eventual sanctions, thus preserving the sustainability of the business, fairness throughout the process, and the autonomy of the proposed solution. Smart contracts, behaving as escrow accounts, completely replace the centralized platforms when objectively and transparently handling each dispute. In

spirit, our solution works as a "*proof-of-guilt*" consensus mechanism where there is always an implicit agreement on the stakeholder exhibiting undesirable behavior, who in turn is penalized for doing so. Such penalties are enforced automatically by smart contracts. Having described our proposed conflict resolution mechanism, it is important to highlight how the solution increases the delivery worker's duties throughout the process. But on the other hand, despite the increased responsibility, we believe the proposed solution may also be advantageous to him/her. For example, the delivery fee can be adjusted to benefit the delivery worker after the savings obtained from eliminating platforms' commission fees.

Demonstration

The fourth step in the design science research framework is to demonstrate how to instantiate the proposed design (Peffers et al., 2007). Recall that our solution is a hybrid system in that data are stored on-chain and off-chain. In particular, restaurants store menu-related data on a public or their own information-technology infrastructure (off-chain storage). On the other hand, orders (transactions) are stored directly on the adopted blockchain (on-chain storage) alongside customers' and restaurants' digital addresses. We use Ethereum as the blockchain model since it enables fast prototyping and evaluation of the proposed solution. Moreover, Ethereum allows for the deployment of and interaction with smart contracts. Using the Solidity programming language, we developed five smart contracts as the core of our solution. We call them *Chain*, *Order*, *View*, *Interact*, and *Storage*.

The contract *Chain* is a factory of contracts of type *Order*. In the contract *Chain*, the orders are created and stored on the blockchain. *Chain* is also responsible for storing the addresses (identifiers) of the contracts of type *Order* as well as the total quantity of orders that have been submitted. Second, each created contract *Order* contains all the information about a single order — including all the dispute-resolution rules — since it represents an order made by a customer. Specifically, the contract *Order* stores the digital addresses of all involved parties, a timestamp for each transaction, the current step along the process, the ordered items together with prices, the delivery fee, and information about all deposited values connected with the order. To manage the order workflow updates, a state machine pattern is used to implement the contract *Order*. Through the state machine, one can, for example, determine the current order status and find out who will be charged if any problem arises. Third, the contract *View* is responsible for retrieving information about: 1) a specific order; 2) all the orders associated with a given stakeholder; or 3) the open orders that are available to delivery workers. As the contract *Order* only stores information related to one order, it is then necessary to send a request to the contract *View* to retrieve all contracts *Order* linked to the requesting user. Fourth, the contract *Interact* is the channel used by the parties to interact with a given order, *e.g.*, to accept, update, or cancel an order. The state machine described in the previous paragraph makes it possible to verify whether the user who requested an interaction through the contract *Interact* is authorized and whether the interaction is valid. Finally, the contract *Storage* is responsible for storing encrypted data, such as the customer's private information and details about orders. The encrypted data can only be decrypted by the stakeholder whose public key was used during the encryption process. For example, a new order appears to the delivery worker after a restaurant accepts an order. If the delivery worker decides to accept the delivery, then s/he must confirm it by sending his/her public key to the contract *Storage*. Subsequently, the customer retrieves the delivery worker's public key from the blockchain, encrypts his/her personal information and ordered items using the delivery worker's public key, and sends the output to the contract *Storage*. Then, when the delivery worker receives the order information from the contract *Order*, s/he accesses the contract *Storage* to retrieve the customer's encrypted data and order info. Finally, the delivery worker decrypts the customer's information and orders items using his/her private key.

For testing purposes, we compiled and deployed the smart contracts to one of the Ethereum test networks called *Ropsten*. The contracts are now available for external calls by *decentralized applications* (DApps), *i.e.*, applications that use decentralized technology, such as blockchain, as the backend. Moreover, one can access all the information stored by the system through blockchain explorer services such as Etherscan, which means that any user can verify the data stored on the blockchain. This feature is important to satisfy design principle #2 (transparency). Regarding the system's frontend, we developed a specific DApp for each stakeholder, *i.e.*, customers, restaurants, and delivery workers.[1] The three versions of the application connect to an Ethereum blockchain using a *cryptocurrency wallet* called Metamask. Cryptocurrency

[1] The source code of all DApps and contracts are available at https://github.com/rodrigofolha/foodchain.

wallets work as a gateway between DApps and blockchain. They manage digital identity and all cryptographic keys, thus having no need for an application to store and manage those keys.

Evaluation

The fifth step in the design science research framework concerns evaluating how well the proposed artifact fulfills the design requirements. We first perform a *static analysis* (Hevner et al., 2004) to examine the artifact's structure for one quality, namely cost. Next, we follow a *scenarios* approach where we construct detailed scenarios to demonstrate the transparency property of our solution. Then, we follow an *informed argument* approach, where we build arguments in favor of the artifact's utility concerning privacy. Finally, we explain how our work relates to previous blockchain-based solutions for food delivery services.

Cost Reduction

Design requirement #1 states that food delivery services must be supported by systems that ensure economic feasibility. Before analyzing the transaction costs when using our blockchain-based solution, it is important to first explain key concepts behind our chosen blockchain model, namely Ethereum. We note that the primary Ethereum network (*mainnet*) is public, meaning that anyone can join the network as a node or a user. Moreover, Ethereum's source code is also public, which implies that anyone can download the code and start a new *public* or *permissioned* blockchain network. The latter means that different users may have different permissions to, for example, join the network as a node, read/write data from/to the blockchain, *etc*. Focusing on the cost aspect, the question that arises is then: which network type (public or permissioned) should one deploy our solution to? We argue next that it is not feasible, cost-wise, to deploy our application to Ethereum's mainnet. Instead, a permissioned network should be created to reduce costs.

To back up the above argument, we note that whenever a user or a smart contract creates a transaction that stores data on Ethereum's mainnet, that user/smart contract must pay a transaction fee. Such a fee is dependent on the performed computations measured in terms of a metric called *gas*. The gas price in *Ether* — the currency used within the Ethereum blockchain ecosystem — is dynamic and defined by the user proposing the transaction. In particular, the higher the suggested gas fees in Ether, the quicker the transaction will be processed (added to a block) by a node. Now, the price of Ether varies according to the demand for that cryptocurrency. Thus, the underlying price volatility brings uncertainty to users and applications relying on Ethereum's mainnet since the transaction fees can change drastically in a matter of hours. As a consequence, using Ethereum's mainnet brings potentially unbounded and highly volatile costs to the blockchain-based food delivery solution. To illustrate the above point, we create all the transactions required, from placing an order all the way to its delivery. The gas cost for all the transactions summed to 5,253,369 gas units. At the time of writing, it is necessary to pay 12 gwei per gas to process transactions in approximately 30 seconds. The gwei price in U.S. dollars is $0.00000253. Thus, the total cost to process a single order using Ethereum mainnet is the staggering value of 159 U.S. dollars.

Instead of relying on public blockchain networks, our suggestion for a cost-effective deployment is to create specialized permissioned blockchain networks. Specifically, we envision the creation of consortia led by restaurant associations, consumer protection groups, and delivery worker unions, who will jointly define the governance of the blockchain network. We simulate and evaluate the IT-related cost of operating such a network using the DLPS framework (Sedlmeir *et al.,* 2021). Specifically, we run the *writeMuchData* benchmark that simulates heavy input/output workload. Our solution can achieve 309 transactions per second as maximum throughput when Ethereum runs on a permissioned network on Amazon Web Services consisting of ten nodes of type m5.large, each having the storage capacity of 13 GB in SSD. The cost to perform the experiments was 1.4847 USD for 1.4773 hours, which resulted in a monthly cost of approximately 30*24*1.4847/1.4773 ≈ 723 USD. For comparison's sake, Grubhub receives approximately 745,000 daily orders (Grubhub, 2021). That order volume means a total cost of 0.0000323 USD per transaction using our proposed solution in a permissioned blockchain.

To incentivize decentralization, we suggest that any stakeholder with the appropriate resources join the network to participate in the consensus mechanism and provide processing power to execute smart contracts. In this network, onerous consensus mechanisms, such as proof of work, can be replaced by faster and less computationally intensive practical byzantine fault tolerance techniques. Moreover, highly volatile cryptocurrencies, such as Bitcoin or Ether, can be replaced by stablecoins, *i.e.,* cryptocurrencies whose value

is pegged to fiat currencies or assets like gold. Using stablecoins, the price per transaction can be determined by the governance model, *e.g.*, it can be fixed and low enough just to keep the infrastructure alive.

We note that several important questions are to be addressed regarding the governance of the proposed blockchain network. For example, how are decision management rights and control rights allocated (Beck *et al.,* 2018)? Or, what happens when erroneous or malicious data are added to the blockchain, an immutable database (Carvalho *et al.,* 2021)? Although our focus in this paper has been on the feasibility and technical aspects of the solution, we do plan in the future to thoroughly investigate governance-related questions and their implications for transaction costs.

Transparency

The second design requirement concerns transparency in terms of transactions and dispute resolution. Focusing on the latter aspect, we have created two scenarios based on the data collected from our interviews with stakeholders (see Problem Identification and Motivation). These scenarios simulate conflicts and resolutions under our proposed solution and under a traditional food delivery platform mimicking the biggest of such platforms in Brazil. For each scenario, we examine the transparency aspect with respect to two questions: 1) *who pays for the damages?* And 2) *who is to blame?*

Scenario 1: Who pays for the damages?

Consider the scenario where a customer places an order with a restaurant via a centralized platform. Then, the restaurant prepares the order and dispatches it using the services of a delivery worker arranged by the platform. Next, say that the delivery worker informs through the platform that the order has been delivered. However, after the order status is updated, the customer complains through the platform that s/he has not received the order.

In this scenario, it is unclear who is to blame: either the delivery worker correctly delivered the order and the customer behaved maliciously, or the delivery worker misbehaved and the customer is at a loss. One of the customers who took part in our interviews indicated he had experienced a similar situation. After the event, that customer registered a complaint with the food delivery platform, a dispute was initiated, and the customer was eventually reimbursed fully. The customer said during the interview that he still does not know who paid for the reimbursement and that he believes the delivery worker is to be blamed.

We note that, under our proposed solution, the delivery worker would not be able to finish the delivery without the customer's acknowledgment of receipt. Moreover, if the delivery worker does not finish the order within a certain time specified in the *Order* contract, the customer may cancel the order at any time after that. If that happens, the smart contracts will automatically consider the delivery worker responsible for the problem and refund the customer and the restaurant for having paid for and prepared the food. Therefore, the stakeholder charged due to non-compliance is known in advance under our proposed solution. That naturally results in a more effective and efficient conflict resolution mechanism.

Scenario 2: Who is to blame?

Consider a second scenario where a customer orders a main course and a side dish. Next, the restaurant prepares the order, and a delivery worker picks the food up and delivers it to the customer's provided address. Subsequently, the customer contacts the restaurant to complain about the forgotten side dish. In a traditional food delivery platform, there are three potential explanations for the dispute: 1) the restaurant could have made a mistake by forgetting to include the side dish; 2) the delivery worker could have intentionally or not modified the order; or 3) the customer could have lied about not receiving the side dish in order to be refunded or receive extra food. One restaurant owner who took part in our interviews reported that he had experienced a similar situation before. He also indicated that his customers try to solve similar problems in one of two ways: 1) by complaining directly to the restaurant; or 2) by registering the problem with the centralized food delivery platform. In the first case, the restaurants often solve the problem by sending a new side dish to the customer for free, and the restaurants bear the costs. In the second case, the platform fully reimburses the customer, and the restaurant does not receive any payment for the order.

Under our decentralized solution, the delivery worker must check the order before delivering it to prevent the application of a smart contract rule from being enforced against him/her. If the order is correct, s/he

then delivers it to the customer, who must check its content before acknowledging receipt. When the above scenario happens, only two outcomes can occur: First, the delivery worker accepts the error, cancels the order, and refunds both the restaurant and the customer either fully or partially, depending on the rule in the *Order* contract. Second, the delivery worker does not acknowledge his/her error, and the restaurant must decide whether or not the customer is right. Therefore, all the involved stakeholders are aware of any decision and who is to be blamed.

Discussion

We conclude that myopic decisions are made when centralized food delivery platforms intervene during disputes. That happens because such platforms do not implement the steps required to collect data relevant to fair dispute resolution. Our solution shows that when those required steps are implemented, a centralized food delivery platform is no longer needed as the stakeholders can transact directly in a P2P fashion; trust among the parties is achieved via blockchain technology instead of an intermediary. Another relevant aspect of our solution concerns its legal validity. In particular, smart contracts are already considered legal contracts in many jurisdictions (Gilcrest and Carvalho, 2018). That implies that malicious stakeholders may not be able to legally challenge the validity of a smart contract-based dispute resolution mechanism.

Privacy

DR #3 concerns privacy. In particular, customers might not be willing to have their food preferences and sensitive information, such as physical addresses, publicly available to all. Likewise, restaurants might not be willing to disclose information about sales numbers. Our solution accomplishes the above by heavily relying on asymmetric encryption. In particular, by using a restaurant's public key to encrypt details about an order, we ensure that specific details about all the orders received by a restaurant are only available to that restaurant since it is the only entity to have the necessary private key to decrypt the data. Similarly, data related to a customer is only available to the delivery worker who needs that data for delivery. DR #3 also concerns data ownership. Besides profiting from commission fees, centralized food delivery platforms also have a monopoly over the data produced by all stakeholders. The platform can make a profit from that by, for example, selling data about consumption habits to restaurants. Thus, it is crucial to preserve data ownership and make sure that only the stakeholders involved in the transactions have access to the generated data. Overall, our solution illustrates how blockchain and cryptography can be effectively used to ensure privacy and data access control and, thus, avoid exposing stakeholders' sensitive information.

Related Blockchain-based Solutions

We are now able to evaluate our solution by comparing its features against other existing blockchain-based solutions for food delivery. Although there are many articles on the related area of supply chain and logistics (Hastig and Sodhi, 2020), the literature on blockchain-based food delivery solutions is really sparse. One notable exception is the Lisk Restaurant Sidechain (Alves, 2021), which offers an infrastructure based on the Lisk blockchain network to receive requests from restaurants, store encrypted orders, and receive payments from customers using a unique cryptocurrency. To a certain degree, this solution addresses some privacy concerns (DR #3) using blockchain technology since a centralized entity does not have a monopoly over the generated data. However, this solution is limited because it does not consider delivery workers.

Bistroo.io (Roos et al., 2020) offers an idea slightly similar to what we propose in this paper in that it tries to connect stakeholders in a P2P fashion. However, similar to the Lisk Restaurant Sidechain and unlike our work, Bistroo.io does not consider any interaction with independent delivery workers. Instead, it offers its own transportation service. Consequently, it does not transparently and automatically address conflicts. In other words, Bistroo.io is a marketplace environment where goods can be purchased and reviewed. To use its service, Bistroo.io charges a hefty commission fee of 5% of the order cost. An interesting aspect of Bistroo.io is that it rewards users willing to share their data by offering the created BIST token.

Conclusion and Future Work

Partially fueled by the COVID-19 pandemic, food delivery platforms have experienced tremendous growth in recent years (Kim et al., 2021). Unfortunately, the hefty commission fees charged by some of these

platforms increase financial stress for both customers and restaurants. That said, we proposed a peer-to-peer system enabled by blockchain and smart contracts to disintermediate food delivery services. Following the DSRF (Peffers et al., 2007), we first defined design requirements based on challenges faced by stakeholders using centralized food delivery platforms. Then, we suggested design principles to handle those design requirements. We finally proposed, instantiated, and evaluated a fully-functional prototype.

Several exciting research directions result from this work. For example, besides the *ex-ante* interviews we already had with key stakeholders, we plan to further evaluate our solution via *ex-post* interviews and compare the proposed solution with the current approaches. That might create extra requirements and principles, thus leading to another design cycle. From a theoretical perspective, it is essential to understand the behavior and interactions among the stakeholders in the presence of deposits. We plan to perform such an analysis by developing and validating game-theoretic models. Overall, our ultimate goal is to generalize our work to a class of logistics problems beyond food delivery, thus creating a *design theory* (Jones and Gregor, 2007) for the disintermediation of specific business processes to be characterized in the future.

REFERENCES

Alves, D. 2021. "Proof-of-Concept (POC) of Restaurant's Food Requests in the Lisk Blockchain/Sidechain," *Journal of Physics: Conference Series* (1828), pp. 1828.

Beck, R., Müller-Bloch, C., and King, J. L. 2018. "Governance in the Blockchain Economy: A Framework and Research Agenda," *Journal of the Association for Information Systems* (10:19), pp. 1020-1034.

Carvalho, A. 2020. "A Permissioned Blockchain-Based Implementation of LMSR Prediction Markets," *Decision Support Systems* (130), pp. 113228.

Carvalho, A. 2021. "Bringing Transparency and Trustworthiness to Loot Boxes with Blockchain and Smart Contracts," Decision Support Systems (144), pp. 113508.

Carvalho, A., Merhout, J. W., Kadiyala, Y., and Bentley, J. 2021. "When Good Blocks Go Bad: Managing Unwanted Blockchain Data," *International Journal of Information Management* (57), pp. 102263.

Chen, M., Hu, M., and Wang, J. 2022. "Food delivery service and restaurant: Friend or foe?," Management Science.

Gilcrest, J. and Carvalho, A. 2018. "Smart Contracts: Legal Considerations," in *Proceedings of the 2018 IEEE International Conference on Big Data*, pp. 3277-3281.

GrubHub. 2021. "About Us," URL: https://about.grubhub.com/about-us/what-is-grubhub/default.aspx (visited on March 1, 2022)

Hastig, G. M. and Sodhi, M. S. 2020. "Blockchain for Supply Chain Traceability: Business Requirements and Critical Success Factors," *Production and Operations Management* (29:4), pp. 935-954.

Hevner, A. R., March, S. T., Park, J., and Ram, S. 2004. "Design Science in Information Systems Research," *MIS Quarterly* (28:1), pp. 75-105.

Jones, D., and Gregor, S. 2007. "The Anatomy of a Design Theory," *Journal of the Association for Information Systems* (8:5), pp. 312-335.

Kim, J., Kim, J., and Wang, Y. 2021. "Uncertainty Risks and Strategic Reaction of Restaurant Firms Amid COVID-19: Evidence from China," *International Journal of Hospitality Management* (92), pp. 102752.

Peffers, K., Tuunanen, T., Rothenberger, M. A., and Chatterjee, S. 2007. "A Design Science Research Methodology for Information Systems Research," *Management Information Systems* (24:3), pp 45-77.

Roos, B., Dohmen, B., and Geelen, B. 2020. "The Peer-2-Peer Food Marketplace," URL: https://bistroo.io/whitepaper (visited on October 20, 2021)

Sedlmeir, J., Ross, P., Luckow, A., Lockl, J., Miehle, D., and Fridgen, G. 2021. "The DLPS: A New Framework for Benchmarking Blockchains," in *Proceedings of the 54th Hawaii International Conference on System Sciences*, pp. 6855-6864.

Singh, S. 2019. "The Soon to be $200B Online Food Delivery is Rapidly Changing the Global Food Industry". URL: www.forbes.com/sites/sarwantsingh/2019/09/09/the-soon-to-be-200b-online-food-delivery-is-rapidly-changing-the-global-food-industry/ (visited on March 1, 2022).

Tkacik, M. 2020. "Restaurants are barely surviving. Delivery apps will kill them," The Washington Post. URL: https://www.washingtonpost.com/outlook/2020/05/29/delivery-apps-restaurants-coronavirus (visited on April 21, 2022).