dAirBnB: experiments on decentralised applications for the sharing economy

Andrea Bracciali *[1], Daniel Broby †[2], and Siham Lamssaoui ‡[3]

[1]University of Stirling, UK
[2]Strathclyde University, UK
[3]Ibn Tofail University, Morocco

Abstract

Blockchain technologies enable trusted interaction between untrusted parties. Traditionally, untrusted parties have a preference for a third party acting as a centralised source of trust, such as an insurer. These men-in-the-middle often have a dominant position, are able to impose fees, exercise control and may engage in censorship. One advantage of blockchain technology is its support of *decentralised trusted interaction*, where the technology alone guarantees trust. Such trust originates from the distributed consensus on information replicated over a network of peers, the majority of which are honest. *Decentralised applications (dApps),* run on blockchains and enforce the "correct" interaction. Travellers, for instance, would trust a dApp in charge of automatically paying proper compensation whenever a flight is delayed. Middle-men, their fees and control are eliminated, markets made more efficient and secure, and new fair interaction modalities enabled. However, middle-men also assume risks, while the innovative blockchain technology is still to be properly understood, used and validated, limiting its large-scale adoption.

We present *dAirBnB*, a proof of concept design of a *decentralised* online accommodation rental marketplace. *dAirBnB* allows two parties to negotiate a rental agreement, and enforces its correct execution by means of suitable economic incentive and punishments. *dAirBnB* runs on the *Tezos* blockchain, which provides effective dApp support and the guarantee that the agreement cannot be tampered with.

Long-term, our contribution is in extending the possibilities that emanate from decentralised applications. Our goal is to explore opportunities, difficulties and technical aspects of dApp development in markets, like the accommodation rental one, that are currently highly centralised.

1 Overview

Whilst technology supports the *sharing economy*, enabling the sharing of resources and services, it does not directly constraint, in itself, the behavioural and economic attitude of participants. This has to be addressed in order to extend access to the market and the sharing economy. A clearly relevant example is the *accommodation rental market*, where ICT technologies allow private individuals to share their available dwellings. As in other similar examples, a limited number of agile tech companies, e.g. AirBnB, Bookings.com and Expedia, support and control the vast majority of the market. Such "intermediaries" support the market by providing low-cost advertising and secure transactions, and assuming part of the risks. However, they also have dominant positions in the market, for instance in terms of fees that they can impose and advertising policies that they can prioritise without needing to disclosure them, making such brilliant example of the gig economy an actually highly centralised market.

*abracciali@gmail.com
†daniel.broby@strath.ac.uk
‡sihamlamssaoui@gmail.com

In order to improve the trust between online participants and promote a fairer market, we explored the advantages of *decentralisation* as supported by blockchain technology. We designed and developed *dAirBnB*, a proof-of-concept decentralised application (dApp) for the highly centralised online accommodation rental market, based on deposit protocols. Under certain assumptions, *dAirBnB* allows for a decentralised negotiation and execution of lodging agreements amongst untrusted parties. Trust is provided by deposit-engineered incentives, designed to promote honest behaviour amongst parties and implemented through smart contracts over the Tezos blockchain and accessory technologies.

Our results allow two untrusted parties to agree on the terms of their rental agreement, encode them into an executable smart contract (a step that can be automated to a certain extent), and then be driven by the smart contract through their interaction, being assured that the stated terms cannot be altered.

The relative simplicity of the relevant terms for the chosen example, led to a rather convincing use case, with several possible suggestions to fruitfully extend the approach. Caveat, constraints and risks are also highlighted and discussed.

Section 2 discusses motivations, advantages and risks of a sharing economy as embodied by centralised platforms, like AirBnB[1]. Section 3 presents the design of the decentralised application *dAirBnB* in the current context of the accommodation rental market. The implementation of *dAirBnB* is described in Section 4. Section 5 discusses the role and relevance of rewards and reputations in decentralised applications. Concluding remarks and future directions are presented in Section 6.

Minimal background. A general understanding of blockchain technologies is assumed. The interested reader can find a general introduction in the SOK paper [BMC+15].

Blockchain technologies enable trusted interaction between untrusted parties. Relevant (digital) information is subject to the decentralised consensus of a network of peers, where, ideally, no dominant position is admitted. Blockchains have been introduced by the Bitcoin cryptocurrency [Nak09]: a large number of algorithmic nodes agree on the current state of digital transactions, by means of distributed consensus on replicated copies of the whole set of transactions, one copy for each node. The system is robust to a minority of dishonest nodes. No node can exercise a dominant position, the system is *decentralised.*

Smart contracts can be understood as *self-enforcing agreements*, encoded in a computer program, according to the original idea in [Sza96]. The execution of such a program on a blockchain makes the agreement untamperable. Further work in this direction include [Gri], and recent applications like [TMP].

A first introduction to *Tezos* can be quickly obtained from the Tezos website and documentation therein[2].

SmartPy[3] is a framework for developing smart contracts for the Tezos blockchain.

2 The centralised market of accommodation rentals

The growth in the online lodging market over the last two decades is hailed as a huge success. Projections[4] forecast that by 2022 half of hotel booking revenues, and an even higher share of short-term rentals, will be online.

The online lodging market is dominated by a few big players which share the vast majority of the market and often plays the role of intermediaries. Dominance of a few big players is not a novelty in online and technology-supported markets, and easily companies like Apple, Google, Amazon, and Facebook come to mind. These are examples of *centralised* markets, featuring a few dominant positions that can exercise a disproportionate control on the overall value of the whole market. Control must not necessarily be monetary, but can take different forms, like for instance

[1]www.airbnb.com
[2]tezos.com
[3]smartpy.io
[4]`https://www.businesswire.com/news/home/20181030005780/en/Global-Online-Accommodation-Booking-Market-2018-2022-One-Half`

censorship, or information and access control, e.g. influencing who can offer their sofa online, worldwide.

In 2017, Booking (launched 1996) and Expedia together, the two main online travel agencies, accounted for more than 1 trillion bookings, while the peer to peer booking platform AirBnB, launched 2006, generated a $2.6 billion revenue[5]. As in the other similar examples, a limited number of agile tech companies, like the ones above, enable and control the vast majority of the accommodation rental transactions. These "intermediaries" support the market by providing low-cost advertising and secure transactions, and assuming part of the risks. As a result of network benefits, platforms may be able to scale up to dominant positions in the global market. The risk, therefore, is that this could give rise to a monopolistic and inefficient regime, with asymmetry of information, inefficient fees and non-transparent policies. Other serious risks regard the management of crisis and unexpected situations, which, even if sporadic, can have important consequences, as for instance documented in [Tem]. From their dominant positions in the market, platforms can control fees and prioritise advertising policies, making such brilliant example of the gig economy an actually highly centralised market in several respects.

Online platforms require a *trust framework* to operate. Big players often enable the market by providing trust amongst participants that have no reason to trust each other. People would typically not allow strangers to occupy their home for a short stay, but if this happens through a platform successfully used by a large number of clients, and the platform provides some identity and reputation framework, and may be also an insurance on the home, than the attitude might change, as the market currently demonstrates. Studies like [FP] have for instance investigated how AirBnB induced trust may affect hosts' attitude to rent their properties. Besides, platforms can also provide forms of support for disputes that have escalated to legal jurisdictions.

Often, platforms provide community-based reputation systems relying on user feedback. One major way of currently achieving trust is through reputation-based feedback between hosts and guests. Bolton et al observe that also such feedback can be biased due to positive reciprocity [BO13]. This reduces its value and therefore harms market efficiency.

Herbert (H), a *host* who rents his flat whenever available, benefits from the tech platform, AirBnB say, and the trust it provides.

However, being dependent on a single dominant player may be a reason of concern for Herbert, since he cannot much negotiate fees and conditions and may have no control on advertising and rental policies.

Gaia (G) is a *guest* looking for accommodation. She also enjoys the availability of an efficient and accessible global market, with a trust and reputation framework and forms of risk protection.

However, Gaia also suffers from potentially high mediation costs, asymmetry of information, and limited negotiation power. One example, for instance, are cancellation policies: Gaia cannot agree on mutually fair cancellation policies, but just accept those proposed by the platform or host. Often, Gaia must guarantee her booking by means of a binding, for her, credit card reservation, that Herbert can cash anytime within a grace period. In that period, Herbert can still withdraw from the agreement, or keep advertising the flat and wait for more convenient deals.

To address these problems, we propose a redesign of the online lodging market trust system, exploring whether and how deposit based reputation models, implemented by smart contracts running on blockchains, can represent a viable, and desirable, alternative to the status quo. The rental accommodation market, indeed, seems to represent an interesting and general testbed for validating the decentralisation promises by blockchain technologies and smart contracts.

Is a decentralised approach possible, and can it make such a market more efficient and fair, still providing a suitable trust framework and risk protection, fostering the participation of players of the sharing economy like Herbert and Gaia?

In order to address the above question, we explored the feasibility of *dAirBnB*, a proof of concept for a *decentralised accommodation rental market*, which we design and implemented as a smart contract on the Tezos blockchain.

[5]`https://en.wikipedia.org/wiki/Airbnb`

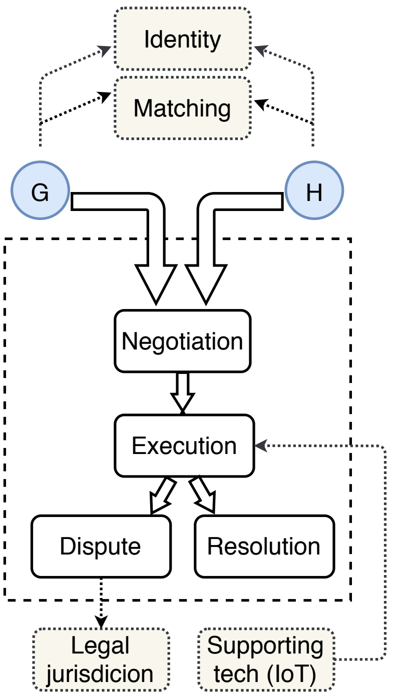


Figure 1: *dAirBnB*: decentralised accommodation rentals. Gaia (G) and Herbert (H) have undergone some form of digital identification, and their offer and demand matched. They have negotiated the terms of a rental contract in the form of a protocol that will be executed as a Tezos smart contract. Disputes may be escalated to a judiciary system (outside of the smart contract, which is represented by the dashed square). Supporting technology interacts with the contract.

3 Decentralising accommodation rentals: opportunities and risks

We consider a basic scenario about the formalisation and *decentralised* execution of a rental agreement between Herbert, the host, and Gaia, the guest. To start with, we abstract away how they have met and formalised their agreement, and assume that the rental terms have been pre-agreed somehow. It is worth noting that the agreement is produced by the free negotiation of Herbert and Gaia, and offers endless possibilities of customisation, according to party's needs.

We discuss how their agreement can be implemented in a smart contract, written in the high-level language *SmartPy* and run on the *Tezos* blockchain. Such smart contract is a building block of *dAirBnB*, and refers to a single agreement between two parties. The agreement considered here features a classical deposit-based incentive system that makes counterproductive for the parties not to adhere to the agreed protocol.

More generally, *dAirBnB* will be a more complex system, integrating for instance a reputation framework and directly supporting the dynamic definition of contracts amongst parties.

The basic scenario is illustrated in Figure 1. Herbert (H) and Gaia (G) have been identified by means of a suitable digital identity provided, and their demand and offer suitably matched. The smart contract, represented by the dashed box, encodes the agreement resulting from negotiation as a protocol. Actions of the protocol may include payments, fill a complaint, or cancelling the agreement itself. The smart contract also includes suitable incentives that encourage "honest" behaviour, i.e. correct execution of the protocol and fulfilment of the implicit rental promises (see Section 5).

It is important to remark that the execution of the smart contract on the blockchain cannot be altered or tampered with, unless an attacker, or one of the two parties, is able to control the majority of the blockchain, which is unfeasible. In this way, Tezos provides the trust framework in place of the centralised platforms currently dominating the market.

3.1 Assumptions

Most relevant assumptions on which *dAirBnB* relies are discussed next. Most of them are reasonably comparable to standard assumptions in the current market, can be reasonably fulfilled by the blockchain or other currently available technology, and are generally compliant with regulations.[6] Here we focus on the dashed square in the figure, the core interaction between G and H, as supported by a *dAirBnB* smart contract.

1. It is possible to associate a real identity to G and H, e.g. by means of a suitable KYC procedure analogous to the one performed by AirBnB, say (box Identity in Figure 1). For the purposes of *dAirBnB*, G and H are identified by their associated address (or public key).

2. G intends to rent the flat offered by H on the market. The matching of G and H is not covered here (box Matching in Figure1).

3. Disputes will be escalated to a suitable legal jurisdiction, if necessary, analogously to what would currently happens in case of disputes that cannot be resolved by the procedures of existing platforms.

4. Supporting external technology, interacting with *dAirBnB* as appropriate, may be used to facilitate the various phases of *dAirBnB*. One example are IoT technologies managing accommodation access, such as the smart lock system Nuki.[7]

3.2 Agreement negotiation and core incentives

The agreement negotiation phase enable G and H to tailor their rental to their needs, agreeing on price, payments, cancellation and conciliation policies, deadlines, and possibly other aspects of interest. The agreement, encoded in a Tezos smart contract, enforces the correct execution of the agreement. Here we present a pre-defined protocol, but it is natural to imagine a library of tested agreement templates that can drive negotiation, and can be composed in the desired overall rental agreement. Most relevant aspects are discussed in the following.

Escrow. A double escrow is used to incentive good behaviour. Both parties will pay a deposit d to the contract, to be returned at the mutually satisfactory completion of the protocol. This is a quite standard approach, early adopted in blockchain-mediated trade systems, such as the Bitcoin-based decentralised marketplace Bitbay, [Bit]. The value of d is a critical parameter, trade-off between being accessible but useless, and effective but (too) onerous. An effective value for d is the value of the rental itself, but it could become unpractical when the rental value grows. A double escrow would likely not be used for buying an house. A formal analysis of the problem based on game theory can be found in [BBMT15]. Other choices are possible: G and H may pay different ds, which may depend on their reputation. H might pay a larger d, linked to the advertisement for its lifetime (unless lost because of misbehaviour).

Price and information asymmetry. Free negotiation may lead to increased market efficiency and facilitated price discovery processes, which are not anymore affected by possibly onerous middleman fees. Furthermore, the transparent encoding of the agreement in the smart may reduce information asymmetry, preventing for instance H from leveraging on rental proposals that are binding for G only, as mentioned above.

Cancellation policy. Currently, a set of cancellation policies are typically offered by platforms and selected by H. Blockchain-supported negotiation may foster fairer policies. G and H may for instance agree a cancellation fee proportional to the average income of the period, as estimated by the smart contract on historical data. Duplicated payments due to cancellations and re-booking could be addressed. *G1* is incentivised to leave some privacy-preserving note the of the cancellation payment on the blockchain. This information would then be available to *G2*, when booking

[6]Compliance with regulators, and also the evolution of rules caused by the new technologies is an interesting topic, which affects blockchain adoption and feasibility. This is scope for future work.

[7]nuki.io

the same period. *G2* might then partially refund *G1* and pay the difference, with mutual benefit. *H* would still benefit from a fair payment, the opportunity of being in the decentralised market, possibly a share of the refund, and a better marketability of the flat. Such behaviour could be easily enforced by encoding in the smart contract.

Privacy and transparency. As the previous example suggested, a suitably incentivised trade-off between privacy and transparency may open interesting scenarios. Consider the implications on taxation and common goods. The latter is extremely relevant for the ongoing debate on the social effects of the sharing economy, such as the impact of touristic rentals on the social thread of cities. A blockchain-guaranteed community fund reinvested in city's infrastructures, for instance, could be embedded in the design of *dAirBnB*. Similarly, applications to government-backed rental agreement for protected categories with tax incentives for hosts, could also be implemented through smart contracts. This is scope for future work.

Liability and worst case scenario. Missing middle men, also the risk protection that they may guarantee ceases to exists. While centralised platforms can support enough (human) workforce to attempt the initial resolution of disputes, the *code is law* interpretation of smart contracts makes them less effective in dealing with unexpected situations. To start with, *dAirBnB* adopts the approach that in case of disputes that can not be resolved within the agreed protocol, *dAirBnB* responsibility will be limited to provide sufficiently detailed documentation to the legal jurisdiction.

The link between a smart contract and its legal implications in a court is an interesting multidisciplinary area of research. It is worth citing *Ricardian contracts* [Gri], an extension of smart contracts embedding in a secure way the legal terms of an agreement together with the code of the corresponding smart contract. Such terms, together with the smart contract, provide enough information to be used in a court case. The idea is practically developed by the Mattereum project[8] [TMP].

3.3 Rental agreement as a protocol

G and *H* have been connected and undergone suitable identification. The actual mechanics of agreement definition is not detailed here. They are identified in the system as *addresses*, i.e. their accounts on the Tezos blockchain (we assume that real identities can be recovered in case of need from addresses through the identification services - suitable privacy is assumed to apply).

Figure 2 shows an agreed protocol, that we have actually implemented and tested as a smart contract on the Tezos blockchain. The implementation of *C* is discussed in Section 4. *C* can be seen as a state machine which reacts to specific actions participants depending on its current state. For instance, the first - and only - thing that *H* and *G* are initially entitled to perform is a payment. The construction of *C* guarantees that no party of the agreement can fail to respect the agreement without incurring in a loss, which makes not fulfilling obligations not convenient. Being the contract deployed and run on Tezos, the parties trust that it cannot be altered and will run exactly as specified.

H and *G* pay a deposit $\$d$ and the deposit plus the rent $\$r$ to *C*, respectively (1. & 2.). *H* could pay a deposit for the whole duration of its flat offer, which could be different from the one paid by *G* and could contribute to *H*'s reputation, the larger the deposit *H* is liable for, the more trustworthy *H* becomes. To the guarantee of *H*, *G* may be required to pay *C* within an agreed timeout t_{-1} (otherwise the agreement is void).

C instructs *N* about the rental agreement between *G* and *H* from time t_0 and end e and other relevant information (3.) *N* is an IoT smart home system, called after Nuki, a commercial system that we are considering to integrate with *dAirBnB*. Nuki is connected to the internet, can communicate with *C* and operates on the flat, particularly enabling locks and certifying, for instance, the actual start of the rental. *N* plays the part of what is called an *oracle*, i.e. a player in charge of injecting information in the blockchain. Such information is typically not validated by the blockchain distributed consensus, making the oracle a point of centralisation. In principle, the parties need to trust that the oracle is well-behaved and does not, for instance, injects fake

[8] mattereum.com

1.	H	$\rightarrow$	C	:	$\$d$	
2.	G	$\rightarrow$	C	:	$\$d + \$r$	$(< t_{-1})$
3.	C	$\rightarrow$	N	:	$< G\ H\ t_0\ e >$	
4^*.	G	$\rightarrow$	C	:		$(< t_0)$ *cancel*
5^*.	C	$\rightarrow$	H	:	$f_{c\ pol}(\$d)$	*cancellation fee*
6^*.	C	$\rightarrow$	G	:	$\$d - f_{c\ pol}(\$d) + \$r$	*refund*
7^*.	C	$\rightarrow$	N	:		$< G\ H\ end >$
8^*.	END					
4.	N	$\rightarrow$	C	:	$< G\ open >$	
5^+.	G	$\rightarrow$	C	:		$(< t_{4.} + t_g)$ *issue*
6^+.	H	$\rightarrow$	C	:	$f_{comp}(\$r)$	*compensation*
7^+.	G	$\rightarrow$	C	:		*accept*
5.	C	$\rightarrow$	H	:	$\frac{1}{2}\$r$	$(t_{4.} + t_g)$
6.	N	$\rightarrow$	C	:	$< G\ bye >$	
7.	C	$\rightarrow$	G	:	$\$d + f_{comp}(\$r)$	
8.	C	$\rightarrow$	H	:	$\$d + \frac{1}{2}\$r - f_{comp}(\$r)$	
9.	END					

Figure 2: The protocol representing the rental agreement between G and H. A generic protocol step, $G \rightarrow C : \$d\ (< t_0)$ *cancel* say, represents G communicating to C about a payment $\$d$, with additional information $(< t_0)$ *cancel*. Steps * represent G cancelling, and steps $^+$ represent G claiming an issue with the flat. These are alternative branches to the standard execution of the protocol (1.-9.).

information favourable to H. For the time being and considering that Nuki is a third-party independent company liable of its reputation, this does not represent a major trust issue.

Prior to t_0, G may cancel the agreement. H could do the same, but this possibility has not been negotiated in the present protocol. G communicate cancellation to C. C immediately compensates H by paying a share of G's deposit, according to the agreed cancellation policy, i.e. $f_{c\ pol}(\$d)$, and then returns the remaining deposit and rent to G. N is notified. The rental ends here (4^*.-8^*.). It is worth noting that no middle-man nor even H can unduly withhold G's money or postpone H's compensation.

G enters the flat. N records this with C, allowing G a grace period $(t_{4.} + t_g)$ to rise a complaint about the flat. Should this happen, H will be able to offer a compensation, if he acknowledge the complaint, that G may be willing to accept (5^+.-7^+.). This will be reflected in the final payments (7.& 8.). Other outcomes are possible from G's complaint that can be suitably encoded in the protocol. For simplicity, we have only considered such an amicable conciliation. Worth reminding that in case of unresolved disagreement, the parties may decide to revert to the judiciary system, as it would have happened with centralised platforms (sometimes filing cases against the platforms itself, too. See [Tem]).

Importantly, this is an example of how trust can be supported by smart contracts: G needs to trust H about the flat, which cannot be checked beforehand, but the contract guarantees that a conciliation procedure will be surely available in case of problems.

At time $t_{4.} + t_g$, it is assumed that G is happy with the flat, and H is credited with half of the rent (assuming that $f_{comp}(\$r) \leq \frac{1}{2}\$r$). Not releasing the whole amount is a form of further guarantee for G, who might discover further problems later on (although this case is not considered in this protocol).

At the end of the rent, N notifies C that G has left the flat, and C returns deposits (+/- compensations if any) and completes the payment to H. The rental agreement has been successfully completed according to what was agreed, and C can be discarded.

Neither H or G may at any time derail from the agreement, the contract is *safe*. For instance, G could only refuse to pay in 2., but then the agreement is void; cancel the booking in 5^*. paying the negotiated fee; or fill in a complaint in 5^+. that could lead to a compensation or be escalated to court. On the other hand, G has had the opportunity to negotiate a suitable deposit and

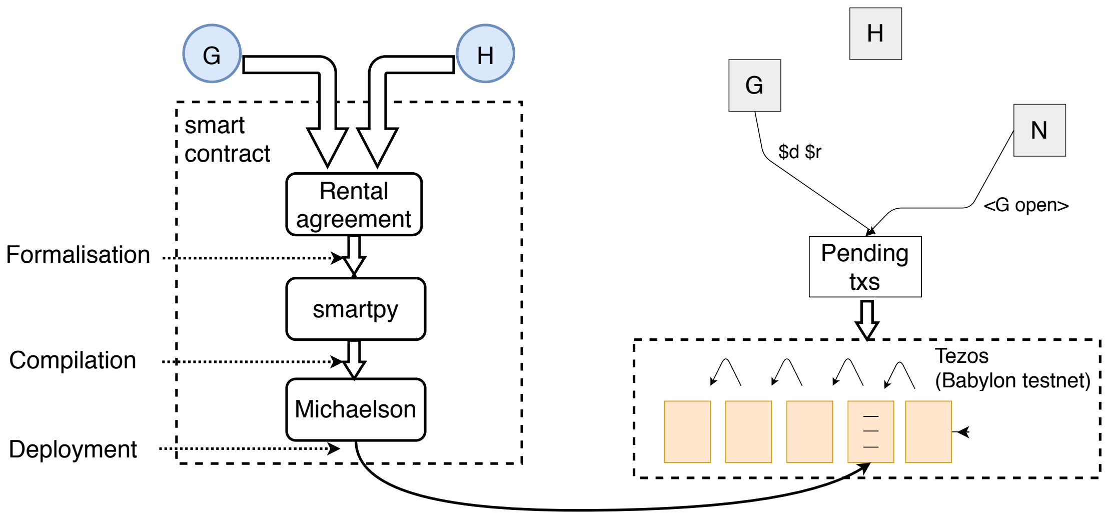


Figure 3: *dAirBnB*'s framework

cancellation policy. For instance, being G a well-known customer, H may decide to waive her the entire deposit. Both G and H reduce (to zero) their fees. The contract if *fair* and *efficient* . The contract is also *decentralised*, in so far there are not centralisation points and dominant positions that might overrule the agreement between G and H (not even G or H can).

As mentioned, the deposit may be hard to negotiate and onerous. However, in the end, it might not be much different from the insurance that customers pay on a rented car, which is comparable to the price of the car itself (see Section 3.2 for deposit quantification). The fundamental difference being that the *dAirBnB* deposit does not require trust in the rental company, since the conditions that trigger its return are immutably encoded in the smart contract.

3.4 Agreement execution

Figure 2 formalises a specific rental agreement negotiated by the untrusted parties G and H, in the form of a protocol.

Such a protocol will be "implemented" by a Tezos smart contract. Such implementation is straightforward, considering the smart contract as a state machine only allowing interaction between parties which is compliant with the protocol.

Each party will be interacting with the contract via a provided verified application with an application, one can think of an extended crypto-wallet. As standard, interaction will be carried out as transactions sent to the contract. Transactions can trigger contract functionalities or carry a payment. Examples are the smart home application N that notifies that G entered the flat, H paying a deposit, and C itself paying H. Transactions are subject to Tezos's latency, i.e. the time needed to produce a new block, and Tezos's fees, although the details of these aspects are not covered here.

The detailed implementation of the contract in Figure 2 is described in the next Section 4.

4 *dAirBnB* as a Tezos smart contract

Figure 3 illustrates the general framework of *dAirBnB*. It focuses on three main components:

1. The formalisation of the rental agreement between G and H as a Tezos contract (left dashed box) and its deployment on the Tezos blockchain.

2. The Tezos blockchain (bottom dashed box), running the contract and interacting with the external world by means of transactions sent by the various actors, such as G, H, N, ...

3. The applications used by the various actors (greyed squares on the top), which send transactions to the blockchain (queued in the pending txs box), and react to the state of the blockchain according to what prescribed by the smart contract.

4.1 Rental agreement as a smart contract

Starting point is a rental protocol agreed by G and H. Such protocol prescribes prices, deposits, cancellation policies, and whatever else G and H have freely agreed as terms of the rental.

We focus here on the possibility of *decentralised* execution of rental agreements between *untrusted parties*, abstracting away from the problem of matching demand and offer. Analogously to the "free" provision of e-mail, services like google maps make advertising easily available to the rental market. These are clearly not decentralised systems, but the currently availability of free rental advertising outwith the channels of dominant platforms, justifies our simplifying assumption that demand and offer have been matched. Another reasonable assumption is that of using an "off-the-shelf" standardised agreement, template agreements, or an agreement freely negotiated by G and H. The rental agreement is then formalised in a Tezos smart contract.

First step is the encoding of the agreement in the smart contract program: *code as law*. The correspondence between agreement and code is hence particularly sensitive and requires, as a general problem, further study. In a first instance, one can imagine to revert to modular libraries of tested agreement components, as also suggested by our running example. The initial encoding has been done in *SmartPy*[9], a python library tailored to Tezos smart contract. Such a high-level language guarantees expressiveness, ease of development and testing.

SmartPy code is compiled in Michelson, the Tezos native stack-based, type-safe, functional language for smart contracts. The Michelson contract can then be deployed at an address on the Tezos blockchain, where its functionalities can be invoked by G and H. We are using the *Carthagenet* test network.

The SmartPy framework also provides a simulation environment, where the contract can be tested locally, and an under-development verification framework.

The *dAirBnB* smart contract imports the SmartPy library and is defined as a python class (Figure 4a). It maintains a state with essential data, including balances in Tez, the Tezos cryptocurrency, and flags that implement an implicit state-machine, specifying which actions are allowed at which stage of the agreement. Participants, like G and H are identified by their addresses, provided by Public Key cryptography, as standard.

The contract consists of a set of functionalities that are invoked by means of transactions sent to the contract's address in the Tezos blockchain. Each transaction specifies a functionality to be invoked (as a function invocation), parameters, the sender and, possibly, the amount of Tez transferred. We do not discuss here the cost of transaction execution, the gas machinery, and possible optimisations. This is scope for future work.

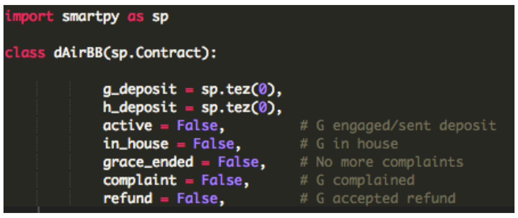
H is in charge of initialising the smart contract according to the agreed parameters, e.g. the deposits and the rent, and pays its deposit.

The agreement comes into effect when G also pays the deposit and the rent. It is worth noting that deposits and rent have been paid to the smart contract and are not anymore available to G and H. The smart contract will redistribute such a value according to what prescribed by the agreement, and facts that have happened, e.g. cancellations or complaints.

The function allowing G to pay deposit and rent (Figure 4c) performs some sanity checks: The sender of the transaction has to be the designated G, as specified by H in the contract initialisation, the amount has to be equal to the agreed deposit and rent. Any other amount would not make the transaction valid and the contract would not activate (and the deposit eventually returned to H). This is part of how the smart contract builds the trust framework: These checks are guaranteed by all nodes in the Tezos network that replicates and validate the execution of the smart contract, on behalf of all the parties, so that H and G must behave as prescribed by the agreed protocol in Figure 2. A distinguishing feature of the blockchain is that the smart contract cannot be tampered with, so that G and H can trust the rules that govern their interaction.

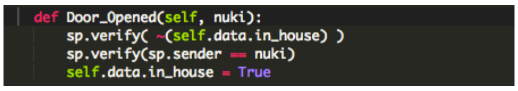
G is allowed to cancel the booking (Figure 4e). This can only be done by G - identities are enforced by PKI cryptography, for an active contract (for which deposits and rent have been paid),

[9]smartpy.io



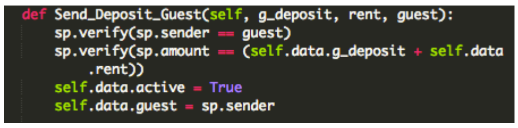
```
import smartpy as sp

class dAirBB(sp.Contract):

            g_deposit = sp.tez(0),
            h_deposit = sp.tez(0),
            active = False,         # G engaged/sent deposit
            in_house = False,       # G in house
            grace_ended = False,    # No more complaints
            complaint = False,      # G complained
            refund = False,         # G accepted refund
```

(a) Contract state



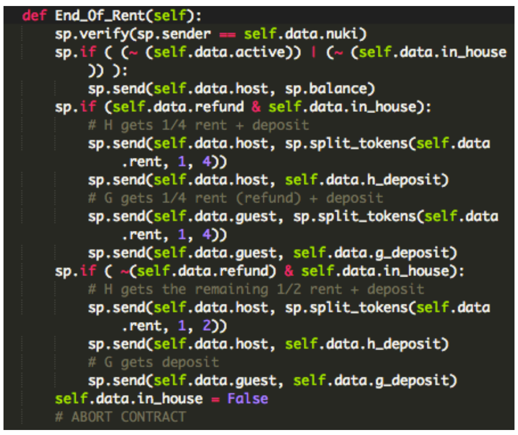
```
    def Door_Opened(self, nuki):
        sp.verify( ~(self.data.in_house) )
        sp.verify(sp.sender == nuki)
        self.data.in_house = True
```

(b) Nuki, the smart home oracle



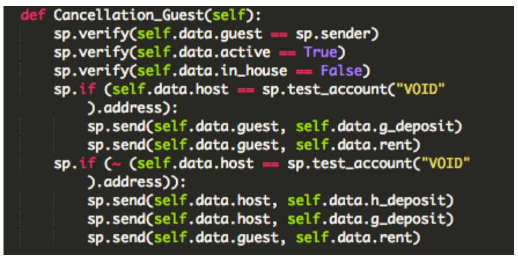
```
    def Send_Deposit_Guest(self, g_deposit, rent, guest):
        sp.verify(sp.sender == guest)
        sp.verify(sp.amount == (self.data.g_deposit + self.data
            .rent))
        self.data.active = True
        self.data.guest = sp.sender
```

(c) G deposit



```
    def End_Of_Rent(self):
        sp.verify(sp.sender == self.data.nuki)
        sp.if ( (~ (self.data.active)) | (~ (self.data.in_house
            )) ):
            sp.send(self.data.host, sp.balance)
        sp.if (self.data.refund & self.data.in_house):
            # H gets 1/4 rent + deposit
            sp.send(self.data.host, sp.split_tokens(self.data
                .rent, 1, 4))
            sp.send(self.data.host, self.data.h_deposit)
            # G gets 1/4 rent (refund) + deposit
            sp.send(self.data.guest, sp.split_tokens(self.data
                .rent, 1, 4))
            sp.send(self.data.guest, self.data.g_deposit)
        sp.if ( ~(self.data.refund) & self.data.in_house):
            # H gets the remaining 1/2 rent + deposit
            sp.send(self.data.host, sp.split_tokens(self.data
                .rent, 1, 2))
            sp.send(self.data.host, self.data.h_deposit)
            # G gets deposit
            sp.send(self.data.guest, self.data.g_deposit)
        self.data.in_house = False
        # ABORT CONTRACT
```

(d) Rental agreement successful termination



```
    def Cancellation_Guest(self):
        sp.verify(self.data.guest == sp.sender)
        sp.verify(self.data.active == True)
        sp.verify(self.data.in_house == False)
        sp.if (self.data.host == sp.test_account("VOID"
            ).address):
            sp.send(self.data.guest, self.data.g_deposit)
            sp.send(self.data.guest, self.data.rent)
        sp.if (~ (self.data.host == sp.test_account("VOID"
            ).address)):
            sp.send(self.data.host, self.data.h_deposit)
            sp.send(self.data.host, self.data.g_deposit)
            sp.send(self.data.guest, self.data.rent)
```

(e) G cancellation

Figure 4: The *dAirBnB* smart contract implementation in SmartPy

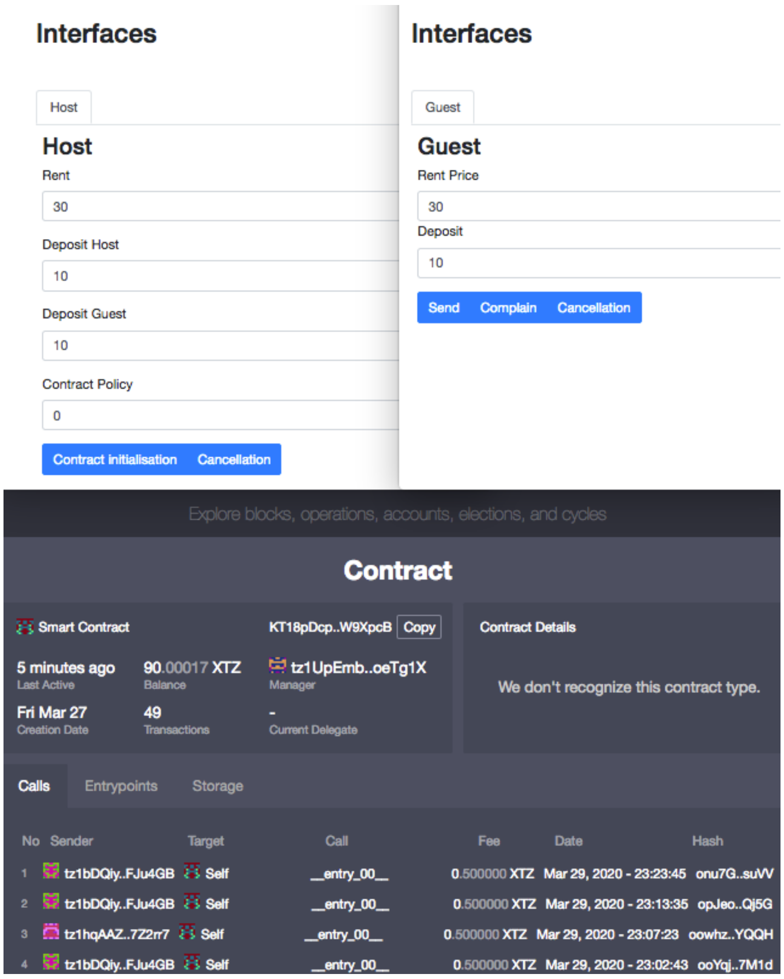
and before entering the flat. If H had not yet paid the deposit - the host address is VOID, G gets back the whole rent and deposit, otherwise the agreed cancellation policy applies. In this case H gets both deposits and G the rent. It is worth noting that this policy has been agreed beforehand by G and H. More sophisticated and fair policies are possible. For instance once could imagine that H is reimbursed proportionally to the actual average expected return for the flat for the period, if the flat is not re-rented. Fairer cancellation policies and mutual benefits enabled by *dAirBnB* are scope for further work.

Injecting external information in a smart contract is a centralisation point because the source needs to be trusted. Trusted sources are typically called *oracles*. *dAirBnB* uses an oracle to get information about the development of the rental agreement. We are thinking to a smart home system, like the commercial Nuki[10], an Internet of Things framework for the management of locks and access to a flat through Bluetooth and wireless technology. A Nuki component N, properly identified by its own address, interacts with the smart contract, and for instance, set the flag stating that G has accessed the flat (Figure 4b). This flag has implications on the continuation of the rental and determination of costs. Specifically, G and H have agreed a grace period for a complaint about the state of the flat. In case of complaint, we have here adopted the policy that H will offer 1/4 of the rent as compensation. G is free to accept or leave. N will notify the end of the grace period, the flat is ok and H immediately gets half of the rent. Delaying part of the payment can be used to build more accountability on H.

Finally, the rental will be completed after a notification by N that G has left the flat (or the rent period has expired and Nuki has locked the flat) (Figure 4d). At this point,after the due checks, the smart contract will refund rent and deposits according to what had been agreed in origin. Specifically, if the contract has not been activated or G left, H gets the whole contract balance, including its deposit at the minimum. If G has completed the rent, the balance will be suitably split according to whether there was a complaint/refund or not. All rights are guaranteed.

All the smart contract functions are fully deterministic and can be invoked on testnet blockchain.

[10]nuki.io



No	Sender	Target	Call	Fee	Date	Hash
1	tz1bDQiy..FJu4GB	Self	__entry_00__	0.500000 XTZ	Mar 29, 2020 - 23:23:45	onu7G..suVV
2	tz1bDQiy..FJu4GB	Self	__entry_00__	0.500000 XTZ	Mar 29, 2020 - 23:13:35	opJeo..Qj5G
3	tz1hqAAZ..7Z2rr7	Self	__entry_00__	0.500000 XTZ	Mar 29, 2020 - 23:07:23	oowhz..YQQH
4	tz1bDQiy..FJu4GB	Self	__entry_00__	0.500000 XTZ	Mar 29, 2020 - 23:02:43	ooYqj..7M1d

Figure 5: Interfaces of the Host and Guest web application, and an explorer showing the *dAirBnB* contract deployed on the Carthagenet Tezos testnet, and the list of its transactions.

4.2 Untrusted parties's applications

Untrusted parties, including G, H and N, interact with the smart contract through suitable interfaces. We have implemented prototype web applications for the client side, i.e. the participants, of *dAirBnB*. These web applications are portable and allow for fast development. We have exploited the ConseilJs framework on top of JavaScript and HTML5, for interfacing web applications to the *dAirBnB* contract deployed on the Carthagenet Tezos test net.

In order to interact with the contract, each participants has assigned a testnet account, with associated public and private keys. This is easily supported by SmaryPy.io.

There are two modalities for contract invocation: the Michelson or Micheline format. Michelson contracts are a single-entry points, with only one function that can be called externally. Such function acts as a router, selecting the actual function to be called from one of its parameters `params`.

We assume that an an external identity provider guarantees a suitably recognisable identity associated to the keys. Analogously to existing centralised platform, escalating a dispute to a court is always a possibility, and *dAirBnB* provides certified tracking of all the happened transactions and agreed contract. Link to identities is part of such certification.

Web interfaces of G and H are shown in Figure 5, together with an explorer showing the deployed contract. In our context, explorer are web applications that render in a easily accessible and suitably structured the information encoded in a blockchain.

The interfaces show the parameters associated to the transactions, e.g. the amount of deposit and rent for G, and the transactions that are currently enabled, according to the state of play. Since in a trustless decentralised context security can not be demanded to the participants, whose behaviour cannot be constrained, further security checks are performed on the blockchain-side, rejecting transactions that are not legal.

The general format of the ConseilJs method used to interact with the contract is

```
conseiljs.TezosNodeWriter.sendContractInvocationOperation(
   tezosNode, keystore, contractAddress, 10000, 100000, '', 1000, 100000,
   undefined, params, conseiljs.TezosParameterFormat.Michelson).
```

Such calls are associated to a given address and public/private keys, i.e. a participant.

The ConseilJs call remotely triggers a function of the smart contract (subject to a type check of the parameters).

Acceptance of the transaction is subject of the minting of a new block, which will register the transaction, and is constrained by the minting speed of Tezos. In our tests, transactions on the Carthagenet requires a few tens of seconds.

In our tests, the cost of the transactions was averaged in between 1 and 2 Tez, currently about 2USD. A limited number of transactions, 1-4 say, are required to each participant, making it extremely economically convenient when compared to existing centralised platforms.

The explorer in Figure 5 shows reference information about the deployed smart contract, its current balance and originator, and a list of transactions.

5 Rewards and reputation

The literature on reputation in "trusted community marketplaces" has largely focused on ratings and their distributions. In this respect, it is interesting that Hu et al observe a J shaped distribution in negative reviews [HP09]. Reviewers prefer to leave positive comments. Fradkin et al show this is the case even with two-sided review protocols [FP]. We argue that this supports our deposit protocol as a superior builder of trust.

Akerlof introduced economic models in which trust is important [Ake78]. He argued that guarantees are preconditions for trade and production. Where these are ambiguous, Graham's Law suggests that the the transaction suffers. Clear incentives, such as the deposit protocol we propose, should differentiate between good and bad quality hosts and guests, thereby enhancing price outcomes.

The decentralised application we propose is a trust model. The deposit is the guarantee that there is a penalty in leaving the rental or not following the protocol. There are alternative hybrid systems that can act as incentives. One could, for example, build in open banking permissions to each parties bank account. In this way, the *dAirBnB* can verify ability to pay. Combined with a direct debit smart contract, the deposit would only be drawn down when the contract was cancelled by either party.

5.1 Deposit based incentives

The deposit based approach we take is as follows. In agreement, *Gaia (G)*, the guest, and *Herbert (H)*, the host pay each other a contract rental sum (R). Should they disagree, the negotiation is void. Table 1 shows the decentralised scenario where if either guest or host disagree after entering into the contract they default a deposit.

In the consensus phase, *Gaia (G)*, the guest, and *Herbert (H)*, the host have probabilistic outcomes on their fulfilling the contract as depicted in equation 1. The deposit held back from both parties depends on the expectations εD.

$$\varepsilon D = \frac{Hp - G}{R} = \frac{R(H+1) - 1}{R} \tag{1}$$

Where:

		Host, H	
Guest, G		*Agree*	*Disagree*
	Agree	$(H, -G)$	$(-H)$
	Disagree	$(-G)$	$(Void)$

Table 1: dAirBnB: A scenario between guest *Gaia (G)* and host *Herbet (H)* where both parties have to decide whether or not they should honor the deal made or forgo a deposit.

εD = Expectation of the fraction of the rent to be taken as deposit, R= Rent agreed, H = Host outcome, G = Guest outcome, p = Probability.

The consensus process is illustrated in Figure 6. H has a deposit loss history depicted by one star. G has a deposit loss history depicted by three stars. The consensus is formed around a mean of two stars.

★☆☆☆☆ H, deposit loss history
★★★☆☆ G, deposit loss history
★★☆☆☆ H+G consensus on εD

Figure 6

The stars depict dAirBnB's probability of deposit forfeiture. Gaia (G) and Herbert's (H) deposit defaults are compared. The consensus determines εD. The general framework supposes that there a two asymmetric outcomes from this comparison. These are depicted in the following formula:

$$nh_i^{dA} = \phi^i(H_1^i, H + G_2^i) \tag{2}$$

Where: nh_i^{dA}= the competing hosts on dAirBnB, the function ϕ is decreasing as H+G is increasing.

or,

$$ng_i^{dA} = \phi^i(G_1^i, H + G_2^i) \tag{3}$$

Where: ng_i^{dA}= the competing hguests on dAirBnB, the function ϕ is decreasing as H+G is increasing.

The deposits act as incentives to establish trust. Incentives are economical and make good behaviour Pareto optimal. In the example above, they are based on a traditional escrow scheme, where both host and guest behaviour is constrained by a deposit that will be returned only at the mutually satisfactorily completion of the agreed protocol. The deposit is determined by consensus negotiation which reduces the issue of rental heterogeneity.

5.2 Reputation systems

Although the definition of a reputation system is out of scope here, it is worth noting some possible extensions of the presented contract toward reputations.

As a first step, the contract should incorporate permanent identities to which reputation could be attached. Differently from the one-shot approach here discussed.

Then different strategies could be adopted. Reputation could be ascertained *by referral*, as a transitive closure of a trusted relationships between individuals, which could be easily deduced from the blockchain. Alternatively, it could be more traditionally ascertained *by endorsement*, for instance exploiting micro-transactions to record appreciation.

6 Concluding remarks and future directions

We have designed and developed *dAirBnB*, a proof of concept smart contract for a decentralised accommodation rental marketplace.

One of our goal was to clarify the opportunities that smart contracts running on blockchains offer to untrusted parties willing to negotiate and execute a rental agreement in a decentralised way. The choice of the application domain was particularly interesting, since the digital market of accommodation rentals is a particularly centralised one.

Other analogous proposal are currently being developed, advocating the power of blockchain disintermediated markets, witnessing the interest for the technology and the market opportunity, also given the huge capitalisation of the digital rental market. Amongst early proposals, can be mentioned the Universal Sharing Network[11], the Trips Community[12], Emphy[13], and Lovehomeswap[14]. Often these combine cryptocurrency payments with peer2peer negotiation and standardised agreements. Many are pre-trade or under-development.

Our experiment has highlighted a number of opportunities and risks for further development. Particularly, we have started to investigate how the technology can allow custom agreements tailored to maximise participants' interest, while maintaining strong security.

Another specific focus of *dAirBnB* is the actual execution of the agreed and personalised contract, from activation to resolution, supporting the agreed alternatives in an automated way, and depending on the evolving state of play. For instance, the contract can supervise agreed back-up plans in case of problems, like the simple compensation scheme illustrated.

dAirBnB is one of the first running on Tezos, to the best of our knowledge, and integrating smart home systems as trusted oracles injecting information on the actual development of the rental agreement.

It will be interesting to study the potential profitability of a decentralised system like *dAirBnB*, for instance to acquire resources that could even further protect the decentralised and autonomous interaction of parties.

Finally, a formal verification of the properties experimentally illustrated here, will be of strong interest and an challenging open research problem.

Acknowledgement. Authors would like to thank the anonymous yourcodesucks@riot for their support about the ConseilJS framework, and Fadi Barbara for early discussions on the topic.

References

[Ake78] G.A. Akerlof. The market for "lemons": Quality uncertainty and the market mechanism. *Uncertainty in economics*, 1978.

[BBMT15] Giancarlo Bigi, Andrea Bracciali, Giovanni Meacci, and Emilio Tuosto. Validation of decentralised smart contracts through game theory and formal methods. In Chiara Bodei, Gian Luigi Ferrari, and Corrado Priami, editors, *Programming Languages with Applications to Biology and Security - Essays Dedicated to Pierpaolo Degano on the Occasion of His 65th Birthday*, volume 9465 of *Lecture Notes in Computer Science*, pages 142–161. Springer, 2015.

[Bit] BitBay. Double deposit escrow. `https://bitbay.market/double-deposit-escrow/`.

[BMC$^+$15] Joseph Bonneau, Andrew Miller, Jeremy Clark, Arvind Narayanan, Joshua A Kroll, and Edward W Felten. Sok: Research perspectives and challenges for bitcoin and cryptocurrencies. In *Security and Privacy (SP), 2015 IEEE Symposium on*, pages 104–121. IEEE, 2015.

[BO13] Greiner B. Bolton, G. and A. Ockenfels. Engineering trust: reciprocity in the production of reputation information. *Management science*, 59(2):265–285, 2013.

[11]https://slock.it/usn
[12]https://tripscommunity.com/en/
[13]https://emphy.io/
[14]https://www.lovehomeswap.com/how-it-works

[FP] Grewal E. Holtz D. Fradkin, A. and M. Pearson. Reporting bias and reciprocity in online reviews: Evidence from field experiments on airbnb. 2015.

[Gri] Ian Grigg. The ricardian contract. `http://iang.org/papers/ricardian_contract.html`.

[HP09] Zhang J. Hu, N. and P.A. Pavlou. Overcoming the j-shaped distribution of product reviews. *Communications of the ACM*, 52(10):144–147, 2009.

[Nak09] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. *Bitcoin project white paper*, 2009.

[Sza96] Nick Szabo. Smart contracts: Building blocks for digital markets. `http://www.fon.hum.uva.nl/rob/Courses/InformationInSpeech/CDROM/Literature/LOTwinterschool2006/szabo.best.vwh.net/smart_contracts_2.html`, 1996.

[Tem] James Temperton. Threats, fear and chaos: the messy fall of an Airbnb scam empire. `https://www.wired.co.uk/article/airbnb-scam-london-suspended`. Wired, 2020.

[TMP] The Mattereum Project. Smart contracts. Real property - working paper. Technical report, available at mattereum.com.